AppBuilder

By Magic Software Enterprises

Magic Software AppBuilder

Version 3.2

Development Tools Reference Guide

Corporate Headquarters:

Magic Software Enterprises
5 Haplada Street,
Or Yehuda 60218, Israel
Tel +972 3 5389213
Fax +972 3 5389333

Development Tools Reference Guide

This guide describes AppBuilder Construction Workbench. AppBuilder Construction Workbench is an integrated development environment that you use to build your application on a workstation. The Workbench provides easy access to a full set of tools to design, prepare, execute, and debug your application.

The following subjects are covered:

Introduction to Construction Workbench

The AppBuilder Construction Workbench is an integrated development environment that you use to build your application on a workstation. The Workbench provides easy access to a full set of tools to design, prepare, execute, and debug your application.
From the AppBuilder Construction Workbench, you can perform the following tasks:

- Display, access, and modify application objects from a repository
- Use design tools to create high-level representations of the design of an application
- Use display tools to create the user interface and the high-level code
- Create the pseudo-code and any components
- Build the application on target platforms
- See the results of operations from analysis to verification to preparation
- Run the application and debugging tools
- Use scripting tools to speed application development

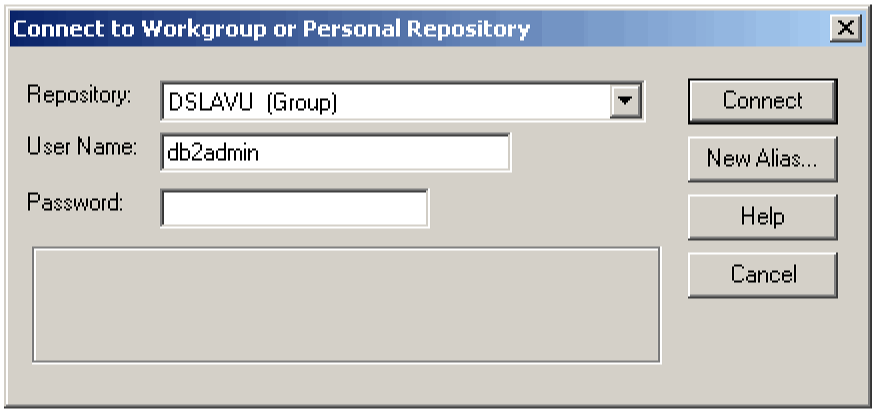
The following topics are discussed in this guide:

Opening the Construction Workbench

To open the Construction Workbench, complete the following steps:

1. From the Start menu, select **AppBuilder > Construction Workbench**. The Connect to Freeway Group or Personal Repository a dialog appears and prompts you for your password (see Connect to Workgroup or Personal Repository dialog sample).

Connect to Workgroup or Personal Repository dialog sample



Repository: DSLAVU (Group)

User Name: db2admin

Password:

Note
You must first create the Personal Repository or create an Alias connection to a Workgroup Repository. Refer to the *Repository Administration Guide for Workgroup and Personal Repositories* for more information.

2. Type your password and click **Connect**.

3. Select the version and click **Continue**. The Construction Workbench opens.

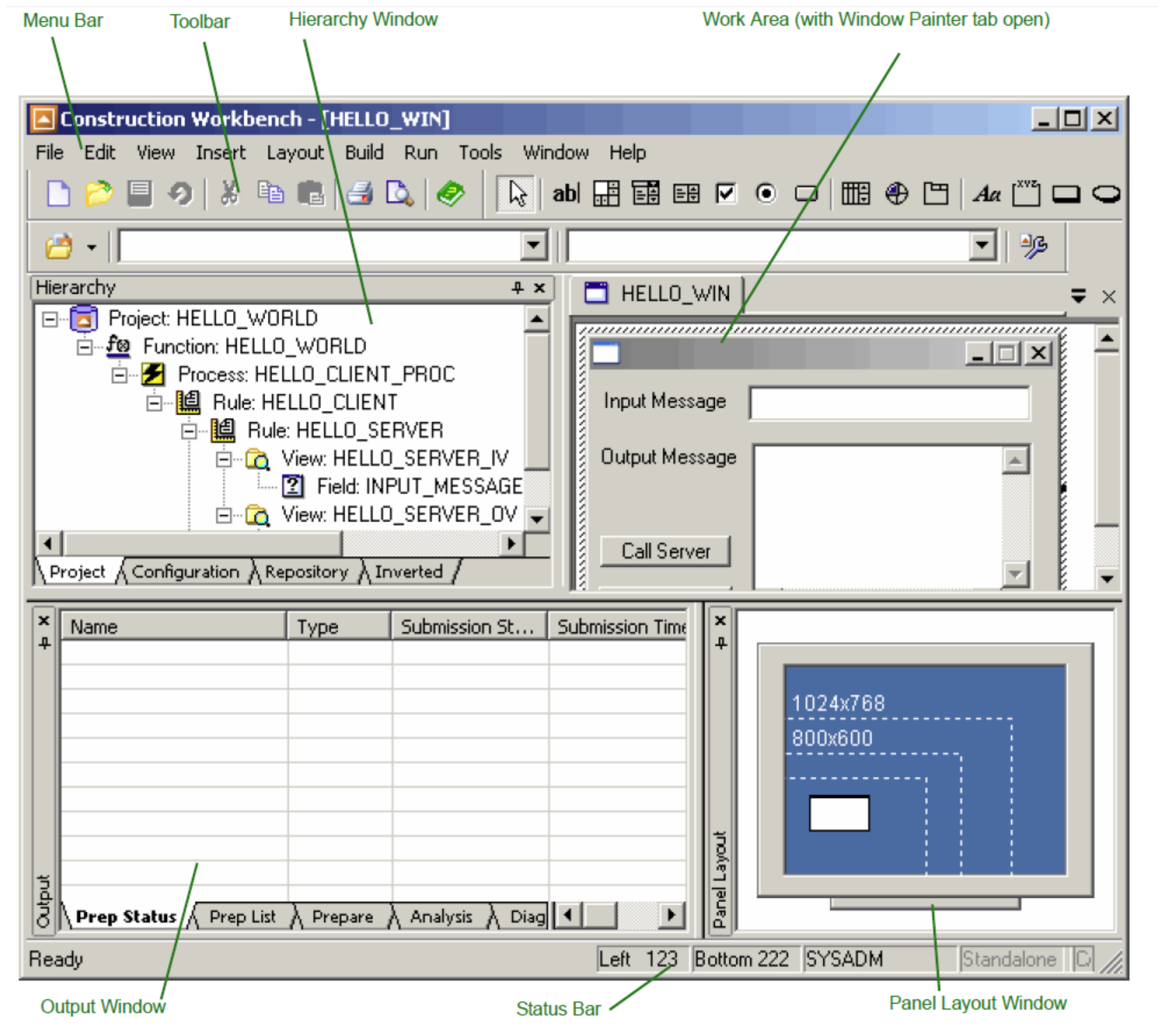
Understanding the Construction Workbench Interface

The Construction Workbench interface (as shown in AppBuilder Construction Workbench) is divided into the following areas:

Construction Workbench Interface

Area	Function
Menu Bars	Provide access to AppBuilder commands through the menu and the keyboard shortcuts.
Dockable Control Bars	Include a collection of buttons that provide quick access to tools, commands, and objects, and change when you change focus from one tool to another.
Hierarchy Window	Contains tabs to build a project hierarchy and deployment configuration, and provides an area to store and query objects from the repository and display an object's parents.
Work Area	Provides an area for the specific AppBuilder tools (for example, Rule Painter, drawing tools, etc.) to open a window in which you can work.
Output Window	Contains tabs to monitor the outputs of preparation, analysis, verification, etc.
Status Bar	Displays information about the current preparation mode, help for each tool, and other information.
Object Property Window	Displays the properties of the currently selected repository object.

AppBuilder Construction Workbench


Menu Bar
Toolbar
Hierarchy Window
Work Area (with Window Painter tab open)
Construction Workbench - [HELLO_WIN]
File Edit View Insert Layout Build Run Tools Window Help
Hierarchy
Project: HELLO_WORLD
Function: HELLO_WORLD
Process: HELLO_CLIENT_PROC
Rule: HELLO_CLIENT
Rule: HELLO_SERVER
View: HELLO_SERVER_IV
Field: INPUT_MESSAGE
View: HELLO_SERVER_OV
Project Configuration Repository Inverted
HELLO_WIN
Input Message
Output Message
Call Server
Name Type Submission St... Submission Time
Output
Prep Status Prep List Prepare Analysis Diag
Panel Layout
1024x768
800x600
Ready
Left 123 Bottom 222 SYSADM Standalone
Output Window
Status Bar
Panel Layout Window

Hierarchy Window

The Hierarchy Window is a graphical representation of your application hierarchy. Typically, the Hierarchy Window is on the left side of the Construction Workbench. Use this window to display and interact with the applications objects in the repository. For detailed information about the Hierarchy window, see Using the Hierarchy Window.

Work Area

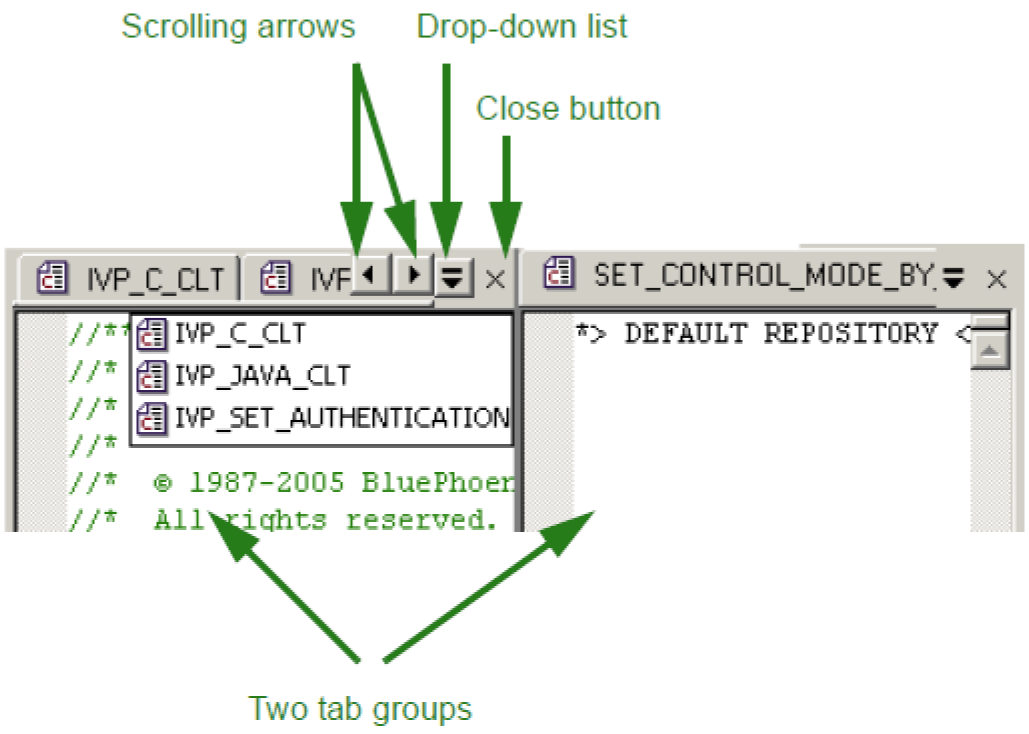
The Construction Workbench tools (Window Painter, Rule Painter, Entity Relationship Diagram, etc.) open in the work area. Typically, the work area is on the right side of the Construction Workbench interface. Each tool opens a separate tab in the work area. A tool tab must be active to activate the options listed on the *Windows* menu.
The following sections cover arranging and managing windows in the Construction Workbench:

- Managing the Tool Tabs
- Docking and Floating Windows
- Navigating Among Tabs
- Managing Multiple Tabs

Managing the Tool Tabs

You can open multiple tool tabs on the Construction Workbench. You can create a new tab, close all the open tabs, and you can group the tool tabs from the **Window** menu, **New Window**, **Close All Documents**, and **New Vertical Tab Group**, respectively. To bring a specific tab to the front of the others, just click its header in the work area, or select its name from the **Window** menu.
Each tab group has a left-arrow, a right-arrow, and a drop-down list with the currently open tabs in the group (see Tab grouping sample).

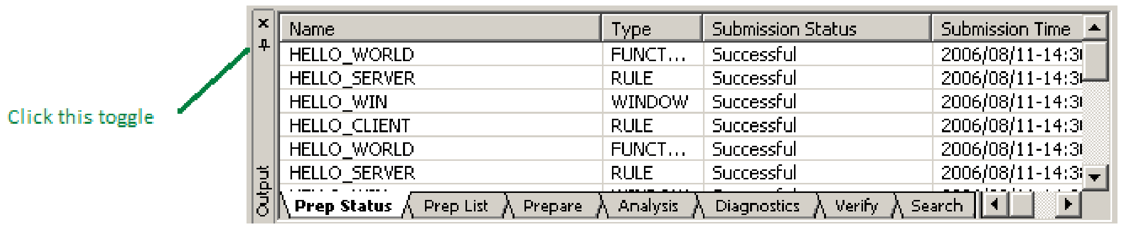
Tab grouping sample



Docking and Floating Windows

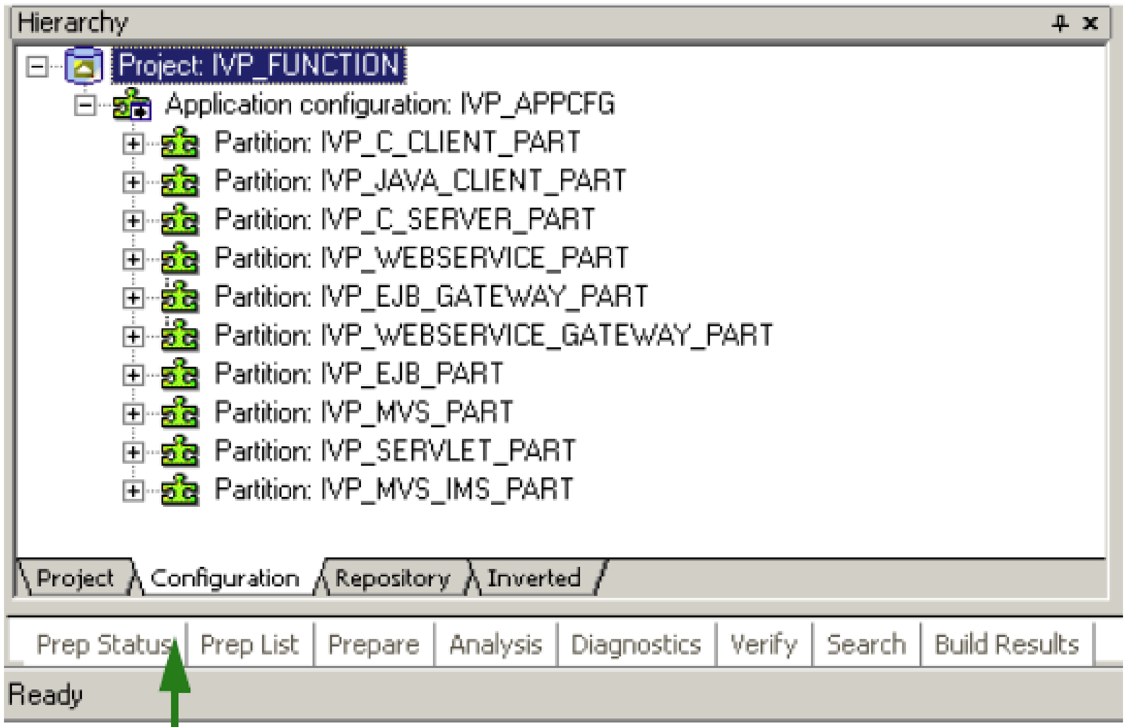
The Hierarchy window, the Output window, and the Object Property window in Window Painter can be docked in the Construction Workbench interface or it can float. You can resize and move them around in the Construction Workbench.
Also you can collapse the windows when they are not necessary. You can see an example in Expanded window sample, Collapsed window sample, and The tab content is displayed when the mouse is over the tab.

Expanded window sample



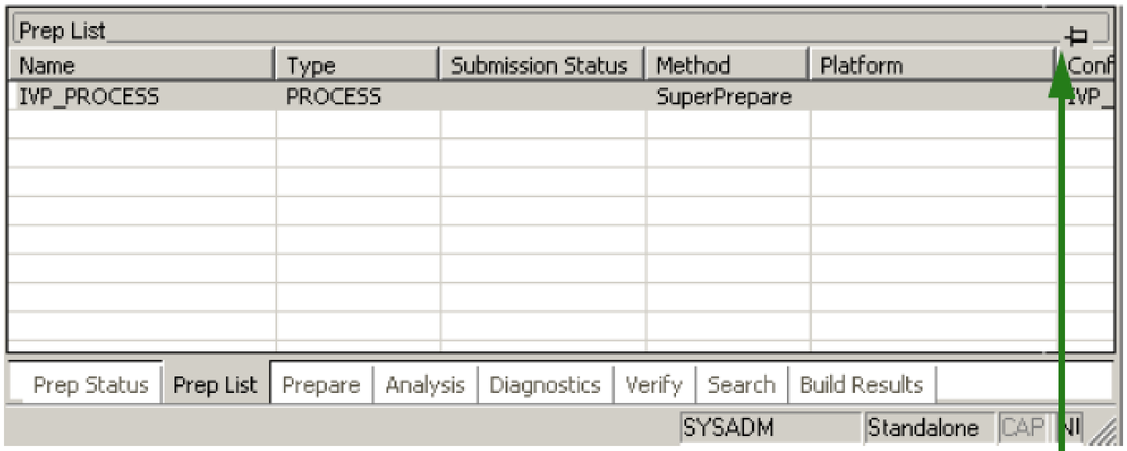
Name	Type	Submission Status	Submission Time
HELLO_WORLD	FUNCT...	Successful	2006/08/11-14:3
HELLO_SERVER	RULE	Successful	2006/08/11-14:3
HELLO_WIN	WINDOW	Successful	2006/08/11-14:3
HELLO_CLIENT	RULE	Successful	2006/08/11-14:3
HELLO_WORLD	FUNCT...	Successful	2006/08/11-14:3
HELLO_SERVER	RULE	Successful	2006/08/11-14:3

Collapsed window sample


Hierarchy
Project: IVP_FUNCTION
Application configuration: IVP_APPCFG
Partition: IVP_C_CLIENT_PART
Partition: IVP_JAVA_CLIENT_PART
Partition: IVP_C_SERVER_PART
Partition: IVP_WEBSERVICE_PART
Partition: IVP_EJB_GATEWAY_PART
Partition: IVP_WEBSERVICE_GATEWAY_PART
Partition: IVP_EJB_PART
Partition: IVP_MVS_PART
Partition: IVP_SERVLET_PART
Partition: IVP_MVS_IMS_PART
Project
Configuration
Repository
Inverted
Prep Status
Prep List
Prepare
Analysis
Diagnostics
Verify
Search
Build Results
Ready

Output window is now collapsed.

The tab content is displayed when the mouse is over the tab



Prep List

Name	Type	Submission Status	Method	Platform	Conf
IVP_PROCESS	PROCESS		SuperPrepare		IVP_

Click the toggle to dock the window

There are three modes for these windows:

- Docked, in which the window is attached to one of the side of WorkBench
- Floating, in which the window appears as a regular window with a thin title bar on the top of the main WorkBench window. In this mode, you can move it out of the main window to anywhere on the desktop.
- Collapsed (Auto hide), in which the window is collapsed, and the tab will automatically display when you hover the mouse over it.

To set the mode for one of these windows:
Right-click the gray area at the top of these windows for the pop-up menu with the following choices:

Pop-Up Menu for Docking and Floating Windows

Menu Choice	Action
Allow Docking	Check the **Allow Docking** option to lock the floating window in a docking position. This option does not dock the floating window. It only allows you to dock it. You still must drag the window to dock. To dock a floating window, drag the window to the desired edge of the Construction Workbench work area. If you uncheck the Allow Docking option when the window is docked, the window will change to floating mode. If you check the Allow Docking option when the window is floating, the window will still be in floating mode; you still must drag it to the side of the main window to dock it. You can resize a window whether it is docked or not.
Hide	Check this option to hide a window from view. To display a hidden window, from the menu bar, click **View** and select the window name you want to display.

Navigating Among Tabs

You can have multiple tabs open in the work area, such as Rules, Windows, diagrams, etc. The last open tab is the active one. To move and activate a different tab and display it in front, click the tab's icon. If the tab is not visible, click the down-arrow to display the drop-down list of the available tabs in that group, then click the tab to activate it (see Tab grouping sample).

Managing Multiple Tabs

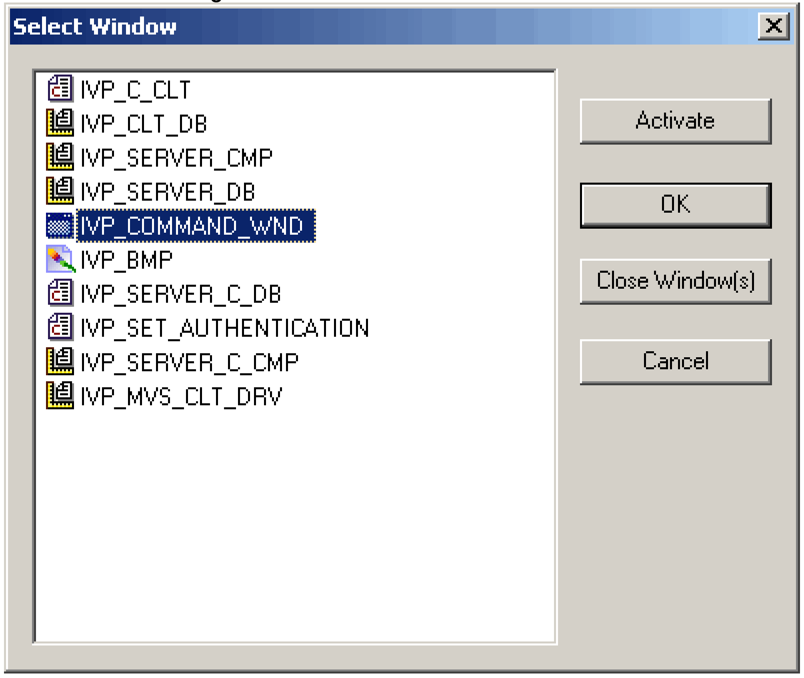
When you have multiple tabs open in the Construction Workbench, icons identify the types of tabs you have open, such as Windows, Rules, and Bitmaps. If you have more than nine tabs open and select **Window > More Windows** on the menu, new controls in the Select Window dialog box for activating or closing selected windows become available.

To activate a window without closing the Select Windows dialog box, select a window on the list and click **Activate**.

To close more than one window at a time, complete the following steps:

1. Press and hold **Ctrl** and select the windows you want to close in the list.\\Or
   Select the first window, then with the **Shift** key pressed, click the last window you want to select. This will select all windows between the first and last window.
2. Select **Close Window(s)**.

Select Window dialog box



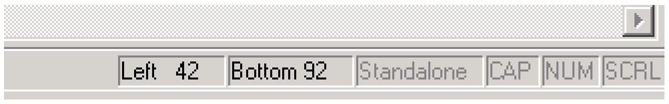
Output Window

AppBuilder prepares objects in the background, so that you can continue working in the Construction Workbench. See AppBuilder Construction Workbench for the location of the Output window. Use the Output window to monitor the preparation status. Use the tabs of the Output window to monitor the preparation process. For detailed information about the Output window, see Output Window.

Status Bar

The Status Bar displays the current preparation mode, basic status information relevant for each tool, and other information. See AppBuilder Construction Workbench for the location of the Status bar. Double-click the preparation mode to quickly toggle the **By default prepare as** an option from **Standalone** to **Distributed**.

AppBuilder Construction Workbench status bar

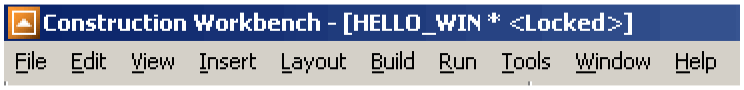


The Status Bar displays the bottom and left coordinates that correspond with the Bottom and Left properties of the selected window control. To display left and top coordinates, select the *Show objects in (Left, Top) coordinates in status bar* check box on the Window Painter section of the Workbench Options window (See Window Painter Options).

Menu Bars

AppBuilder has two ways to access menus in its interface. You can access menus on the menu bar, or you can right-click an object, item, or window in the interface to bring up a context-sensitive menu, also known as a right-click menu.
The Construction Workbench menu bar provides access to commands and functions. The menu bar is dynamic. The menu options change depending on which tool or window is active.

AppBuilder Construction Workbench menu bar



You can also right-click an object or window to access many of the menu options. For example, right-click in a Window Painter tab to display a pop-up menu that contains many of the same options that are available from the Construction Workbench menu.

File Menu

File Menu briefly describes the menu choices on the File menu.

File Menu

Option	Description	Availability
New	Creates a new object or file.	All tools
Open	Opens an object or file.	All tools
Close	Closes the active document.	All tools
Connect	Connects to a repository.	All tools
Disconnect	Disconnects from the active repository.	All tools
New Project	Creates a new project.	All tools
Open Project	Opens an existing project.	All tools
Close Project	Closes an open project.	All tools
Recent Projects	Opens recent projects.	All tools
Commit	Commits session changes to the repository.	All tools
Rollback	Rollback session changes since last commit.	All tools
Import	Imports a bitmap image in Bitmap Viewer.	Bitmap Viewer
Export	Exports a bitmap image in Bitmap Viewer.	Bitmap Viewer
Clear	Removes a bitmap image from Bitmap Viewer.	Bitmap Viewer
Print Setup	Changes the printer and printing options.	All tools
Print Preview	Offers a preview of the print output.	All tools

Print	Prints a document.	All tools
Save As Template	Saves the currently open window as template.	Window Painter
Project Options	Displays a window from where you can set the project options.	All tools
Session Properties	Displays session properties.	All tools
Exit	Exits the Construction Workbench.	All tools

Edit Menu

The following table briefly describes the menu choices on the Edit menu:

Edit menu

Menu Choice	**Description**	**Availability**
Cut	Cuts the selection and moves it to the clipboard.	All tools except for Matrix Builder and Bitmap Viewer
Copy	Copies the selection to the clipboard.	All tools except for Matrix Builder and Bitmap Viewer
Paste	Pastes the selection from the clipboard.	All tools except for Matrix Builder and Bitmap Viewer
Clear	Clears the selected object.	All tools except for Bitmap Viewer
Delete Object	Removes the object from the repository.	Hierarchy window
Delete Relationship	Deletes the relationship from the repository.	Hierarchy window
Move Right	Moves selected items to the right.	Hierarchy window
Move Left	Moves selected items to the left.	Hierarchy window
Select All	Selects all objects.	All tools except for Bitmap Viewer, Set Builder, Matrix Builder.
Deselect All	Deselects selected objects.	All tools except for Rule Painter, Window Painter, Bitmap Viewer, Set Builder, Matrix Builder.
Deselect System Object	Deselects selected system objects.	Hierarchy window
Expand	Expands selected items by the number of levels indicated in the submenu.	Hierarchy window
Collapse	Collapses selected objects.	Hierarchy window
Change Owner	Changes the owner of selected objects.	Hierarchy window
Change Project	Changes the project of selected objects.	Hierarchy window
Search in Files	Search a string in files.	Hierarchy window
Find Object	Finds an object in the hierarchy by name.	Hierarchy window
Find Next	Finds the next occurrence for the previous search.	Hierarchy window
Find Again	Repeats the search operation when working with a matrix.	Matrix Builder
Find Previous	Find the previous object in the hierarchy.	Hierarchy window
Copy menu	Copies the window menu for an open window in Window Painter, so that it can be pasted into another window.	Window Painter

Copy case	Creates case statements for menu items created in the Menu Editor.	Window Painter
Copy function keys	Duplicates function key assignments from one 3270 window to another.	3270 Window Painter
Language	This submenu manages MLUI for a window. It has the following options: Show Default Panel, Show Current Panel, Show Language Panel, Create Language Panel, Delete Language Panel, Set Language, Verify Synchronization, Synchronize Language Panels	Window Painter
Undo	Rolls back last change you made.	All tools except for 3270 Window Painter, Window Painter, and Report Builder, and Hierarchy Window.
Redo	Re-executes the last undone change.	All tools except for 3270 Window Painter, Window Painter, and Report Builder, and Hierarchy Window.
Explode	Explodes the object in the diagram.	Diagram tools
Explode Relations	Explodes the relationships between objects.	Diagram tools
Show Details	Shows the hierarchy for each entity in the diagram and check each entity to ensure it has the attribute structure as described in Building Attribute Hierarchies.	Diagram tools
Hide Details	Hides the entity details.	Diagram tools
Align	Aligns the diagram's objects.	Diagram tools
Navigate	Navigates to another diagram.	Diagram tools
Snap To Grid	Places objects only at grid points.	Diagram tools
Reformat	Redraw the diagram's objects.	Diagram tools
Open as Hierarchy	Opens a scoped version of the entity's hierarchy.	Diagram tools
Delete Object	Deletes an object from the repository.	Hierarchy Window
Find	Searches for an object or a regular expression.	Hierarchy Window
Replace	Replaces an expression with another one.	Hierarchy Window
Go to line	Displays the Go to line dialog.	Rule Painter
Selected	This submenu provides the following options: Uppercase, Lowercase, Move Right, Move Left, Shift Selection Left, Shift Selection Right.	Rule Painter
Line properties	Displays the Selection Line Font dialog for a report.	Report Builder
Insert Line	Inserts a new line at the current cursor position of a report.	Report Builder
Delete Line	Deletes the current line in a report.	Report Builder
Append Line	Adds a line at the end of the report.	Report Builder
Snap	Available when working with reports.	Report Builder
Verify	Verifies the report.	Report Builder

View Menu

The following table briefly describes the menu choices on the View menu.

View Menu

Menu Choice	Description	Availability

Hierarchy	Displays (via toggling on/off) and changes focus to the hierarchy window.	All tools
Status Window	Toggles on/off the display of status window.	All tools
Toolbars	Enables you to select the toolbars to display in the application window.	All tools
Panel Layout	Toggles on/off the display of Panel Layout window.	Window Painter
Parents	Shows parent objects in the Inverted tab.	Hierarchy window
Children	Shows child objects in the Repository tab.	Hierarchy window
Configuration	Edits the .ini configuration file.	Hierarchy window
Format	Changes format specifications for edit fields, list boxes, or MCLBs.	Window Painter
Accelerator	Creates an accelerator for a push button.	Window Painter
Templates	Displays the Workstation or 3270 templates dialog.	3270 Window Painter, Window Painter
Navigate to Hierarchy	Displays the hierarchy of selected object from a tool in Work Area in Hierarchy window.	Diagram Tools, Bitmap Viewer
Zoom	Changes the magnification factor for a bitmap image.	Bitmap Viewer
Workstation/3270	Toggles between the workstation and the 3270 view of a window.	Window Painter, 3270 Window Painter
Properties	Displays the Properties window for a window object.	All tools
Bookmarks	Offers a set of bookmark-related options: Toggle Bookmark, Next Bookmark, Previous Bookmark, Clear All Bookmarks.	Rule Painter
Ruler	Toggles on/off the ruler display for a report.	Report Painter
Alignment Bar	Toggles on/off the alignment bar display for a report.	Report Painter
Diagram Rulers	Toggles on/off the display of diagram rulers.	Diagram tools
Headers/Footers	Displays the Headers/Footers dialog.	Diagram tools
Grid	Toggles on/off the grid display.	Diagram tools
Zoom In	Zooms in the diagram.	Diagram tools
Zoom Out	Zooms out of the diagram.	Diagram tools
Show Normal Size	Shows the diagram at the normal size after a zoom operation.	Diagram tools
Show All	Shows all the diagram in the available tab space.	Diagram tools
Current Template	Displays the Template dialog for the current matrix template.	Matrix Builder
Zoom To Selected Area	Zooms into the selected area.	Diagram tools

Insert Menu

Insert Menu briefly describes the menu choices on the Insert menu:

Insert Menu

Menu Choice	**Description**	**Availability**
Insert	Inserts objects in the hierarchy tree.	Hierarchy window
[Object]	Inserts an available object (item), depending on the currently open tool in the work area.	Diagram tools, Window Painter

Insert Child	Inserts available objects as children of a selected object in the hierarchy tree.	Hierarchy window
Insert Sibling	Inserts available objects as siblings of a selected object in the hierarchy tree.	Hierarchy window
Insert Sibling [Object]	Inserts available objects as siblings of the selected object in the hierarchy tree. The object selected in the hierarchy tree displays in the menu as [Object].	Hierarchy window
Development	Inserts procedural objects. Available options are: Function, Process, Rule, Window, View, Field, Set, Value, Bitmap, Component, File, Report, Section, Component Folder, Physical Event	Hierarchy Window, Repository tab
OO Development	Inserts OO objects. Available options are: Interface, Class, Structure, Exception, Service, Array, Map, List, Set (Container), Typedef, Stereotype, Rule Wrapper, Native Class, Native Exception, Native Name, Native Implements, Native Extends, Native Imports.	Hierarchy Window, Repository tab
Native File	Inserts record formats. Available options are: Data Source, Data Record and Data Field	Hierarchy Window, Repository tab
Database Diagram	Inserts database diagram specific items. Available options are: Table, Key, Column.	Hierarchy Window, Repository tab
Entity Relationship Diagram	Inserts entity relationship diagram specific items. Available options are: Entity, Attribute, Data Type, Identifier, Business Object.	Hierarchy Window, Repository tab
State Transition Diagram	Inserts state transition diagram specific items. Available options are: State and Transition.	Hierarchy Window, Repository tab
Process Dependency Diagram	Inserts process dependency diagram specific items. Available options are: Event, Relationship, Logical Process.	Hierarchy Window, Repository tab
Configuration	Inserts configuration specific items. Available options are: Application Configuration, Partition, Server, Machine, Database, Package.	Hierarchy Window, Repository tab

Analysis Menu

Analysis menu briefly describes the menu choices on the Analysis menu:

Analysis menu

Menu Choice	Description	Availability
Check for Duplicates	Checks for duplicate objects when you are done with a diagram.	Diagram tools
Check for Unnamed Object	Checks for unnamed objects when you are done with a diagram.	Diagram tools
Verify	Verifies a diagram when you are done with it.	Diagram tools
Verify Selected	Verifies the selected items of the diagram when you are done with it.	Diagram tools
Forward Engineering	Forward engineers a diagram.	Diagram tools
Forward Engineering Selected	Forward engineers the selected items of a diagram.	Diagram tools
Trace	Tracks how forward engineering converted entities to tables and produces reports that you can view.	Diagram tools

Trace Selected	Tracks how forward engineering converted entities to tables and produces reports that you can view for the selected items of the diagram.	Diagram tools
Reverse trace	Tracks and reports how the relational entities in the database model correspond to the logical entities in an ERD.	Diagram tools
Turbocycler	Invokes TurboCycler.	Diagram tools
Turbocycler Selected	Invokes turbocycler for the selected objects.	Diagram tools
Stop Analysis	Stops the analysis process.	Diagram tools
Simulate window flow	Simulates the window flow for a window flow diagram.	Window Flow Diagrammer
Terminate simulation	Terminates the simulation for a window flow diagram.	Window Flow Diagrammer

Layout Menu

Layout menu briefly describes the menu choices on the Layout menu:

Layout menu

Menu Choice	Description	Availability
Lock	Toggles on/off the window lock.	Window Painter only
Distribute	Distributes the selected items horizontally or vertically.	Window Painter only
Align	Offers options for aligning the selected items: Left, Right, Top, Bottom, Center Vertically, Center Horizontally.	Window Painter only
Tab order	Sets the tab order in the window object.	Window Painter only
Guide Settings	Displays the Guide Settings dialog.	Window Painter only
Preview in Runtime	Offers a runtime preview of the window object.	Window Painter only

Build Menu

Build Menu briefly describes the menu choices on the Build menu.

Build Menu

Menu Choice	Description	Availability
Prepare	Prepares selected objects.	Hierarchy window
Super Prepare	Superprepares selected objects.	Hierarchy window
Clear Partition	Deletes the package, directories, and files created by preparation (enabled only for Java target).	Hierarchy window, Configuration tab
Create Package	Creates a package (Java environment only).	Hierarchy window, Configuration tab

Deploy Package	Deploys a package (Java environment only).	Hierarchy window, Configuration tab
Create Table	Creates a table in a database.	Hierarchy window
Delete Table	Drops a table in a database.	Hierarchy window
Rebuild	Rebuilds the selected Partition or Application configuration	Hierarchy window, Configuration tab
Rebuild Report	A rebuild report is generated for the selected Partition or Application configuration. It lists all objects that should be rebuilt.	Report Builder
Mark All	Mark All is an option for: • Hierarchy Prepared - Marks the partition or Application configuration as prepared • Rebuild Prepared - Updates the partition or Application configuration as prepared • Unprepared - Marks the partition or Application configuration as unprepared.	Hierarchy window, Configuration tab
Forward Engineering	Forward engineers entities.	Entity Relationship Diagrammer
Preparation Query	Selects the objects at or below the selected object in the Hierarchy Window from a Preparation Query dialog that lists all objects of that type. Available options: Function, Process, Rule, Window, Bitmap, Set, Component, File, Server.	Hierarchy window
Verify	Verifies a rule.	Rule Painter
Verification Language	Selects the verification language for a rule. The available options are: C, Cobol, CServer, CSharp, Java, JavaHTML, JavaServer, and OpenCobol.	Rule Painter

Run Menu

Select this menu to start the execution client for your project. Run Menu briefly describes the menu choices available on the Run menu.

Run Menu

Menu Choice	Description	Availability
Java	Launches the Java Execution Client.	All tools
Windows	Launches the Windows Execution Client.	All tools

Tools Menu

This menu lists the AppBuilder tools that can be accessed in the Construction Workbench. The availability of options listed in AppBuilder Tools Available in Construction Workbench depends on the features you have installed.

AppBuilder Tools Available in Construction Workbench

Option	Description	For More Information, see
Language Editor	Use the Language Editor to create the language objects in the repository for a Multi-Language User Interface (MLUI). When you have created at least one new language and the current language is defined, the AppBuilder MLUI functionality becomes active in the Construction Workbench.	*Multi-Language User Interface Guide*
TurboScripter	TurboScripter increases productivity by enabling you to automate many of the repetitive tasks in the software development lifecycle. It is non-proprietary software that is based on an industry standard. It provides a repository interface that can be accessed from the AppBuilder Construction Workbench or from third-party tools that use the Component Object Model (COM). TurboScripter uses VBScript or JScript to access the properties and methods in AppBuilder.	*Scripting Tools Reference Guide*

TurboCycler	TurboCycler is a proprietary AppBuilder tool that enables you to automate many of the repetitive tasks in the software development lifecycle. TurboCycler Standard Edition, when combined with the TurboCycler Developer's Kit, provides an architecture that you can use to generate objects suitable to AppBuilder. Using templates, you define generation procedures to automate steps in the software development lifecycle. You can write these generation procedures to meet your own requirements.	*Scripting Tools Reference Guide*
Workbench Options	Use the Workbench Options window to specify the various options for each tool in the Construction Workbench.	Workbench Options
Menu Editor	Opens the Menu Editor window.	Using the Menu Editor
MCLB Editor	Opens the MCLB Editor window.	Using the Multicolumn List Box Editor
Chart Editor	Opens the Chart Editor window	Using the Chart Editor
Function Key Editor	Opens the Function Key Editor.	Using the Function Key Editor
Tabbing Editor	Opens the Tabbing Editor.	Using the Tabbing Editor
HTML	Creates (or updates) an HTML page from an AppBuilder window.	Generating HTML from a Window, Updating HTML Attributes
Bitmap Viewer	Opens the Bitmap Viewer tool in a tab.	Bitmap Viewer
Set Builder	Opens the set in Set Builder.	Set Builder
SQL Builder	Opens the SQL Builder dialog.	Using the SQL Builder
Mapping/Setting Wizard	Displays the Mapping and Setting Wizard dialog.	Using the Mapping/Setting Wizard
Reserved words	Opens the Reserved Words window.	Inserting Reserved Words
Record Quick Macro	Creates a quick macro.	Recording a Quick Macro
Play Quick Macro	Plays a quick macro.	Playing a Quick Macro
Edit Quick Macro	Edits a quick macro.	Editing a Quick Macro
Macro	Opens the Macro dialog.	Named Macros

Window Menu

Use this menu to arrange tabs in the Construction Workbench interface. Window Menu briefly describes the menu choices available on the Windows menu.

Window Menu

Menu Choice	Description	Availability
New window	Opens a duplicate of the currently selected tab in the work area.	All tools
Close all documents	Closes all the open tabs in the work area.	All tools
New vertical tab group	Opens a new tab group and moves the currently selected tab in the new group. Only two tab groups can be created.	All tools

Move to previous tab group	Moves the currently selected tab to the previous tab group.	All tools
Move to next tab group	Moves the currently selected tab to the next tab group.	All tools

For more information about arranging and manipulating tabs in AppBuilder and the Construction Workbench, see Managing the Tool Tabs and Navigating Among Tabs.

Help Menu

Select this menu to view the online help system or link to the Customer Support Web site for the latest information. Help Menu items shows the menu items available on this menu.

Help Menu items

Item	Description
Contents	Select this menu choice to access the online documentation.
Search	Select this menu choice to search the online documentation for specific topics.
Index	Select this menu choice to access the index of the online documentation.
Tip of the Day...	Select this choice to set tips to display on Startup.
Magic Software on the Web	Select this choice to link to the Magic Software Web site.
Magic Software Customer Service	Select this choice to link to the Magic Software customer service page.
About Magic Software AppBuilder	Select this choice for information about Magic Software AppBuilder.

From most of the tools and from the Construction Workbench itself, you can view the online help system or link to the Customer Support Web site for the latest information. To access the online help system, from the *Help* pull-down menu, select *Contents* or *Search* or *Index* , depending upon how you want to use the help contents. The Help menu also contains a Tip of the Day command with an option to Show Tips on Startup; these tips can be toggled On/Off. The Help menu also provides access to the Magic Software Web site and customer service links. Finally, the *About* command provides the specifics of the installed version of AppBuilder, including the base version and all fixpacks or other modifications to the application.

Dockable Control Bars

The AppBuilder Construction Workbench user interface has a set of toolbars that can save you time in selecting objects or functions. The first time Construction Workbench is opened, only the Standard and Project toolbars are displayed; but you can display and arrange any or all of the tool-specific toolbars, and the toolbars that are opened when you close the Construction Workbench will open automatically when you open the Construction Workbench again. The list of toolbars that are available is context-sensitive and depends on which windows are active.
To display a list of available toolbars, complete the following steps:

1. Select **View > Toolbars** or right-click in the toolbar area of the Construction Workbench.
2. Select or deselect a toolbar to display it or hide it on the Workbench.

You can anchor toolbars anywhere in the Construction Workbench or float a toolbar to any location on your screen.

The following toolbars can be set to be available in the Construction Workbench interface:

AppBuilder Toolbars

Toolbar	Description
Standard Toolbar	Use the Standard Toolbar to access standard desktop functions, such as cut, paste, and copy, and to commit and rollback changes to and from the repository. This toolbar is open when you open the Construction Workbench.
Project Toolbar	Use the Project toolbar to open projects, specify configurations and partitions, and configure the project settings. This toolbar is open when you open the Construction Workbench.
Hierarchy - Objects Toolbar	Use the Hierarchy - Objects toolbar to add new objects to the application hierarchy and to add existing objects from the repository to your project. This toolbar is only available when you use the Hierarchy window, and specific objects on the toolbar are available only when you are at the correct level of the hierarchy to insert that object.

Hierarchy - Operations Toolbar	Use the Hierarchy - Operations toolbar to manipulate the objects in the hierarchy. This toolbar is only available when you use the Hierarchy window, and specific operations on the toolbar are available only when you are at the correct place in the hierarchy to perform that operation.
Entity Relationship Diagram Toolbar	Use the Entity Relationship (ERD) Diagram toolbar to create objects and relations in an ERD. This toolbar is only available when you are using the ERD tool.
Database Diagram Toolbar	Use the Database Diagram toolbar to create objects and relations in a database diagram. This toolbar is only available when you use the Database Diagram tool.
Matrix Builder Toolbar	Use the Matrix Builder toolbar to handle zooming in and out when using the Matrix Builder. This toolbar is only available when you use the Matrix Builder tool.
Process Dependency Diagram Toolbar	Use the Process Dependency Diagram toolbar when working with Process Dependency Diagrams. This toolbar is only available when you are using the Process Dependency Diagram tool.
Class Diagram Toolbar	Use the Class Diagram toolbar when working with Class Diagrammer tool. This toolbar is only available when you are using Class Diagrammer tool.
Report Toolbar	Use the Report toolbar when the active object in the workspace is a report. This toolbar provides tools for layout and display of a report.
State Transition Diagram Toolbar	Use the State Transition toolbar for working with a State Transition Diagram. This toolbar is only available when you are using the State Transition Diagram tool.
Text Editor – Bookmarks Toolbar	Use the Text Editor - Bookmarks toolbar to find text and use bookmarks to navigate in a text file. This toolbar is only available when you are using the Text Editor tool, which is provided with the Rule Painter and the Component Painter.
Text Editor – Tools Toolbar	Use the Text Editor - Tools toolbar to work with the text editor. This toolbar is only available when you are using the Text Editor tool, which is provided with the Rule Painter and the Component Painter.
Text Editor – Macros Toolbar	Use this toolbar to play, record, pause or delete a macro.
Window – Layout Toolbar	Use this toolbar to align and distribute objects in a window. The Window – Layout toolbar is only available when you are creating or editing a window.
Window – Objects Toolbar	Use this toolbar (shown in Window – Objects toolbar) to quickly add objects to a window when developing the interface of an application. The Window – Objects toolbar is only available when you are creating or editing a window.
Window Flow Diagram Toolbar	Use this toolbar when designing the flow of windows for the application. This toolbar is only available when you are using the Window Flow Diagram tool.
Text Editor - Event Wizard Toolbar	Use this toolbar when applying Events and Event Procedures to Window objects and Rules.
Drawing: properties Window	Use this window to control the properties of a drawing.
Drawing: zoom Toolbar	Use the zoom tools of this toolbar when working on a drawing.
Object Property Window	Use this window to control the properties of the currently selected object.
Properties Window	Use this window to control the properties of the currently selected window object in Window Painter.

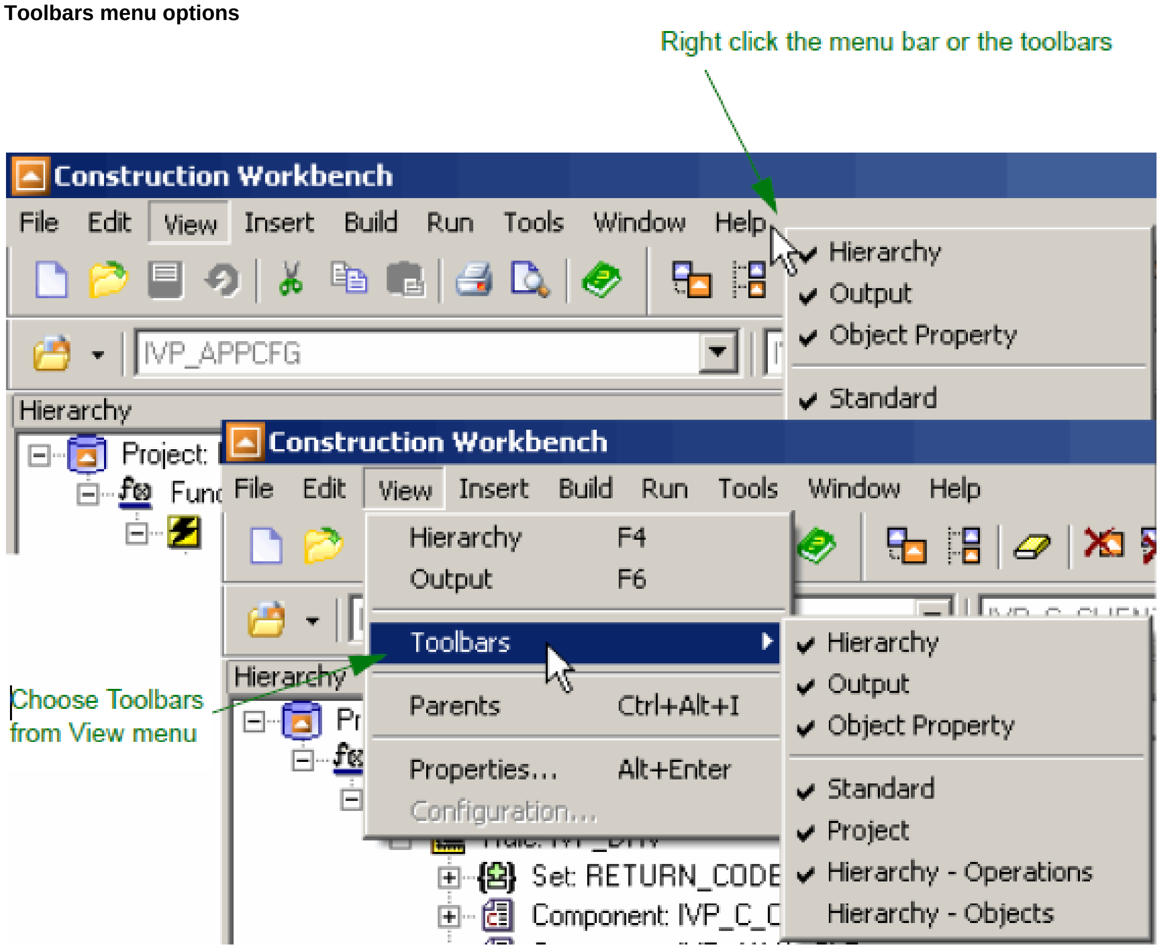
The availability of toolbars depends on the tools being used. For example, when you are working at the project level, the toolbars needed for a project are available. When you are using the text editor to write Rules Language source, the toolbars for Text Editor are available.

To display a list of available toolbars, do one of the following:

- Right-click the menu bar or toolbar area.
- Select **View > Toolbars.**

Available toolbars are displayed as menu options. Check a toolbar to display it.

Toolbars menu options

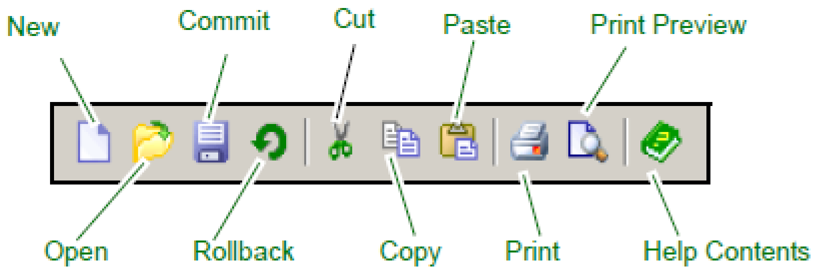


See AppBuilder Toolbars for a complete list of AppBuilder toolbars.

Standard Toolbar

Use the Standard toolbar to quickly use standard desktop functions, such as copy and paste, and to commit and roll back changes to and from the repository. The Standard toolbar is available when you open Construction Workbench.

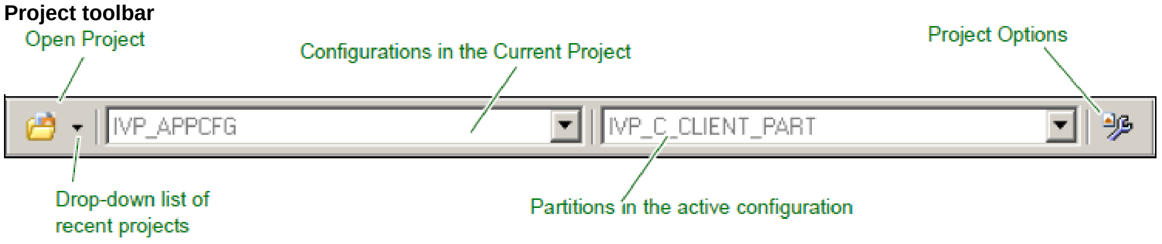
Standard toolbar



Project Toolbar

Use the Project toolbar to open projects, specify configurations and partitions, and configure the project settings. The Project toolbar is displayed when you open the Construction Workbench. The options on this toolbar are grey until you open a specific project.

Project toolbar


Open Project
Configurations in the Current Project
Project Options
IVP_APPCFG
IVP_C_CLIENT_PART
Drop-down list of recent projects
Partitions in the active configuration

Hierarchy Toolbars

Hierarchy toolbars are available only when you use the Hierarchy window. There are two Hierarchy toolbars:

- Hierarchy - Objects Toolbar
- Hierarchy - Operations Toolbar

To display the hierarchy toolbars, complete the following steps:

1. Select **View > Toolbars** from the Construction Workbench menu.
2. From the Toolbars menu, select the toolbar to display (**Hierarchy - Objects** or **Hierarchy - Operations**).

Hierarchy - Objects Toolbar

Use the Hierarchy - Objects toolbar to add new objects to the application hierarchy and to add existing objects from the repository to your project. This toolbar is only available when you use the Hierarchy window, and specific objects on the toolbar are available only when you are at the correct level of the hierarchy to insert that object. For example, to activate the Process icon on the toolbar, you must be at the Function level in the hierarchy.

To access the Hierarchy - Objects toolbar, the Hierarchy window must be displayed. When the Hierarchy window is hidden, this toolbar is also hidden.

Read more about the hierarchy objects in Using the Hierarchy Window.

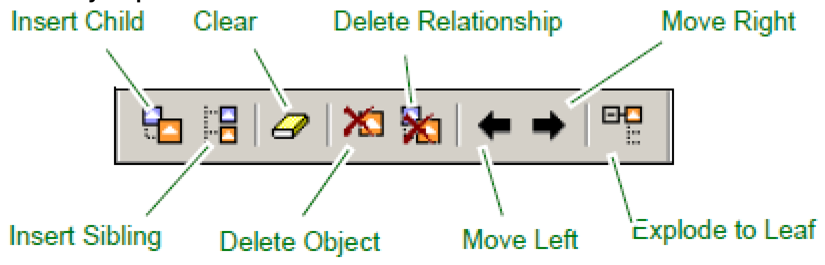
Hierarchy - Operations Toolbar

Use the Hierarchy - Operations toolbar to manipulate the objects in the hierarchy. This toolbar is only available when you use the Hierarchy window, and specific operations on the toolbar are available only when you are at the correct place in the hierarchy to perform that operation. For example, to activate the **Remove From Repository** icon on the toolbar, you must highlight an object that is in the repository.

To access the Hierarchy - Operations toolbar, the Hierarchy window must be displayed. When the Hierarchy window is hidden, this toolbar is also hidden.

Read more about the hierarchy operations in Using the Hierarchy Window.

Hierarchy - Operations toolbar



Entity Relationship Diagram Toolbar

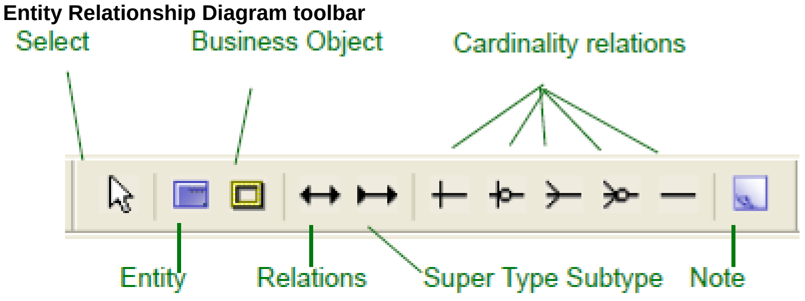
Use the Entity Relationship (ERD) Diagram toolbar to create objects and relations in an ERD. This toolbar is only available when you are using the ERD tool. To access the ERD tool, do the following:

1. Select **File > New.**
   The Create New window displays.
2. Click **Entity Relationship Diagram .**
3. Click **OK .**

The Entity Relationship Diagram tab displays, and the Entity Relationship Diagram toolbar becomes available on the **View > Toolbars** menu.
4. Make sure that the Entity Relationship Diagram toolbar is checked.

Read more about the Entity Relationship Diagram tool in Modeling Tools.

Entity Relationship Diagram toolbar



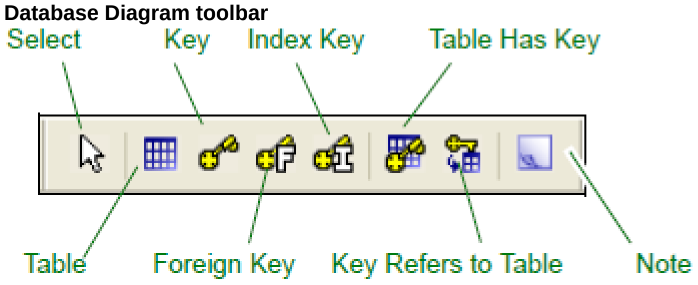
Database Diagram Toolbar

Use the Database Diagram toolbar to create objects and relations in a database diagram. This toolbar is only available when you use the Database Diagram tool. To access the Database Diagram tool, do the following:

1. Select **File > New.**
   The Create New window is displayed.
2. Click **Database Diagram.**
3. Click **OK.**
   The Database Diagram tab displays, and the Database Diagram toolbar becomes available on the **View > Toolbars** menu.
4. Make sure that the Database Diagram toolbar is checked.

Read more about the Database Diagram tool in Modeling Tools.

Database Diagram toolbar

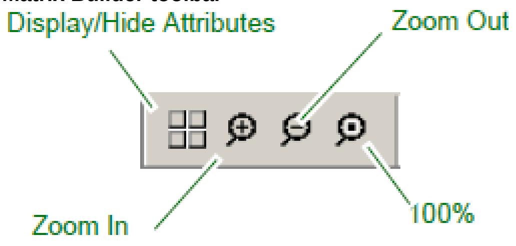


Matrix Builder Toolbar

Use the Matrix Builder toolbar to handle zooming in and out when using the Matrix Builder. This toolbar is only available when you use the Matrix Builder tool. To access the Matrix Builder tool, do the following:

1. Select **File > New.**
   The Create New window is displayed.
2. Click the Matrix icon.
3. Click **OK.**
   The List of Templates tab displays, and the Matrix Builder toolbar becomes available on the **View > Toolbars** menu.
4. Make sure that the Matrix Builder toolbar is checked.

Matrix Builder toolbar



Read more about the Matrix Builder tool in Matrix Builder.

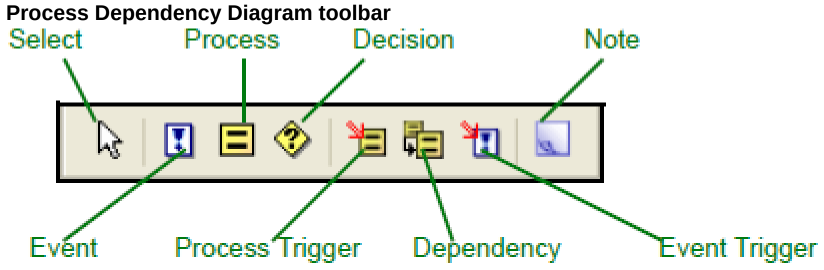
Process Dependency Diagram Toolbar

Use the Process Dependency Diagram toolbar when working with Process Dependency Diagrams. This toolbar is only available when you are using the Process Dependency Diagram tool. To access the Process Dependency Diagram tool, do the following:

1. Select **File > New.**
   The Create New window is displayed.
2. Click **Process Dependency Diagram.**
3. Click **OK.**
   The Process Dependency Diagram tab displays, and the Process Dependency Diagram toolbar is available on the **View > Toolbars** menu.
4. Make sure that the Process Dependency Diagram toolbar is checked.

Read more about the Process Dependency Diagram tool in Modeling Tools.

Process Dependency Diagram toolbar



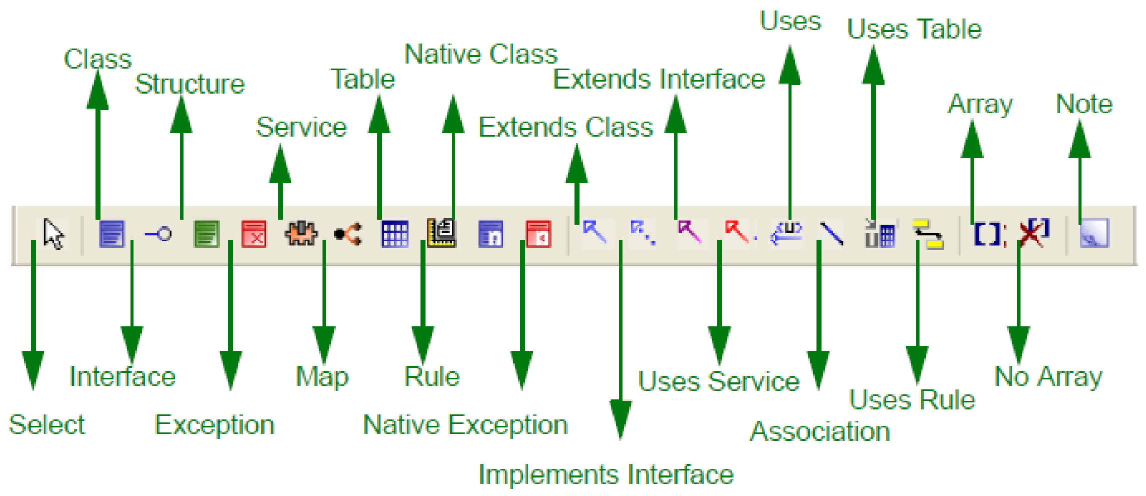
Class Diagram Toolbar

Use the Class Diagram toolbar to create objects and relations in a class diagram. This toolbar is only available when you are using the Class Diagrammer tool. To access the Class Diagrammer tool, follow the steps:

1. Select **File > New.**
   The Create New window displays.
2. Click **Class Diagram.**
3. Click **OK.**
   The Class Diagram tab displays, and the Class Diagram toolbar becomes available.
4. Select **View > Toolbars** menu.
5. Make sure that the Class Diagram toolbar is checked.

See Class Diagram for more information.

Class Diagram toolbar



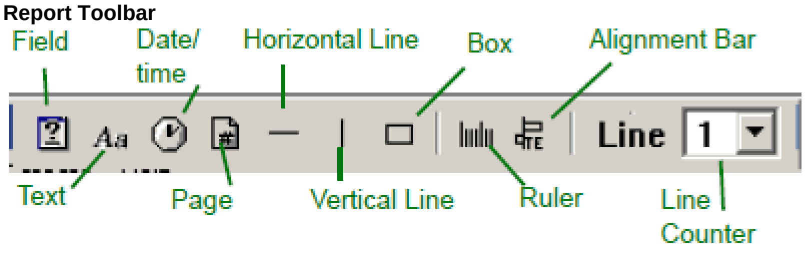
Report Toolbar

Use the Report toolbar when working in the Construction Workbench with reports. This toolbar is only available when you have a report open and active in the Workbench. To access the Report toolbar, do the following:

1. Select **File > New.**
   The Create New window is displayed.

2. Select **Report.**
   A New Report dialog is displayed, requiring that you name the report.
3. Enter a name for the report.
4. Click **OK.**
   The Report opens in Report Painter, and the Report toolbar is available on the **View > Toolbars** menu.

Report Toolbar



For more information on using the Report Toolbar and building or editing reports, see the *Reports Guide* .

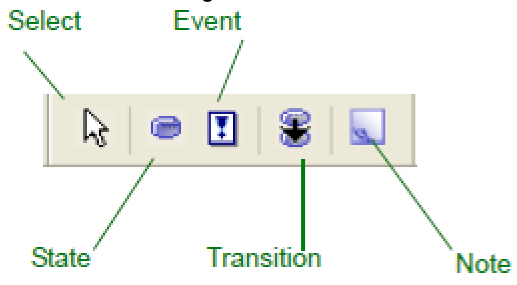
State Transition Diagram Toolbar

Use the State Transition toolbar for working with a State Transition Diagram. This toolbar is only available when you are using the State Transition Diagram tool. To access the State Transition Diagram tool, do the following:

1. Select **File > New.**
   The Create New window is displayed.
2. Click **State Transition Diagram.**
3. Click **OK.**
   The State Transition Diagram tab displays, and the State Transition Diagram toolbar is available on the **View > Toolbars** menu.
4. Make sure that the State Transition Diagram toolbar is checked.

Read more about the State Transition Diagram in Modeling Tools.

State Transition Diagram toolbar

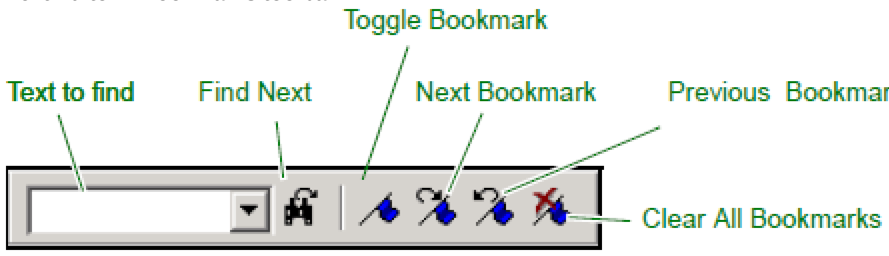


Text Editor – Bookmarks Toolbar

Use the Text Editor - Bookmarks toolbar (shown below) to find text and use bookmarks to navigate in a text file. There is a field for entering a string of text to find. This toolbar is only available when you are using the Text Editor tool, which is provided with the Rule Painter and the Component Painter. To access the Text Editor tool, in the Rule Painter or the Component Painter, open a window that has text and place your cursor in the window. The Text Editor – Bookmarks toolbar is available on the **View > Toolbars** menu, and the Text Editor – Bookmarks toolbar is displayed.

Read more about the Text Editor options in Workbench Options.

Text Editor – Bookmarks toolbar



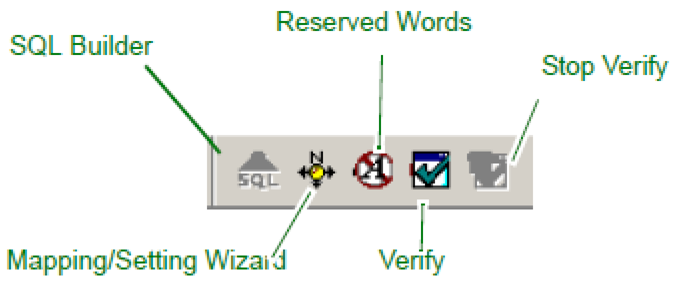
Ctrl+F2, Shift+F2, F2, Ctrl+Shift+F2 are very useful shortcuts to work with bookmarks. These commands are shown in the **View >Bookmarks** menu in the Construction Workbench.

Text Editor - Tools Toolbar

Use the Text Editor - Tools toolbar to work with the text editor. This toolbar is only available when you are using the Text Editor tool, which is provided with the Rule Painter and the Component Painter. To access the Text Editor tool, in the Rule Painter or the Component Painter, open a window that has text and place your cursor in the window. The Text Editor – Tools toolbar is available on the **View > Toolbars** menu, and the Text Editor – Tools toolbar is displayed.

Read more about the Text Editor options in Workbench Options.

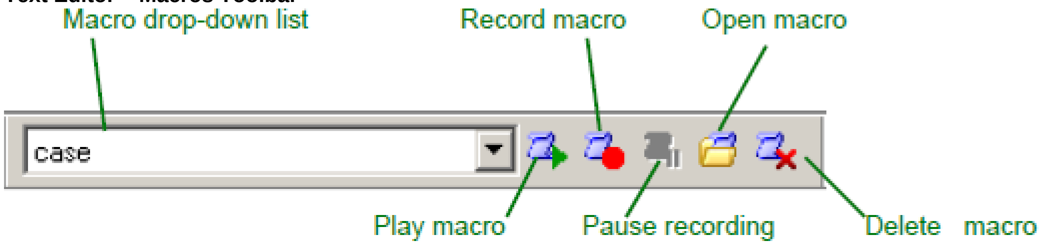
Text Editor – Tools toolbar



Text Editor – Macros Toolbar

Use this toolbar to play, record, pause, and delete a macro. The toolbar is only available when you select it and are working in the Rule Painter.

Text Editor – Macros Toolbar



Window – Layout Toolbar

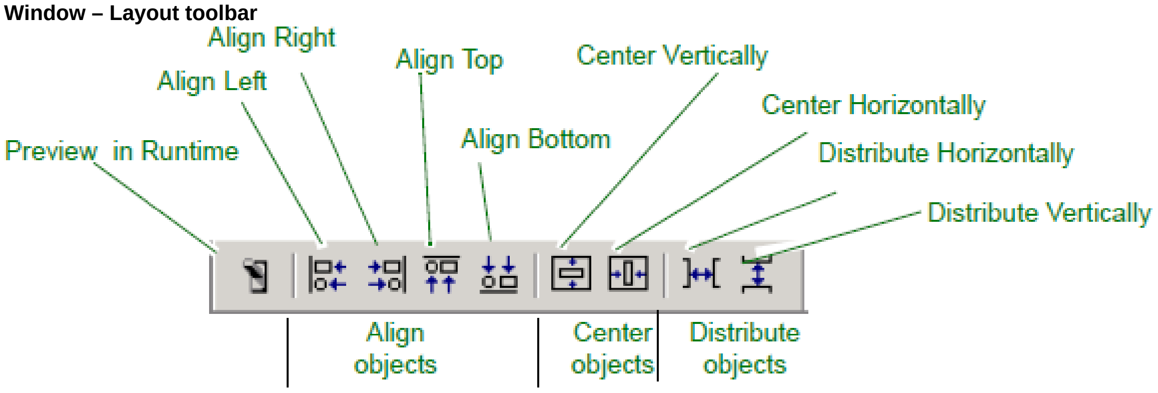
Use this toolbar (shown below) to align and distribute objects in a window. The Window – Layout toolbar is only available when you are creating or editing a window. To access the Window Painter tool, do the following:

1. Click **File > New.**
   The Create New window is displayed.
2. Select **Window** and click **OK.**
   A blank window tab is displayed.

Now the Window – Layout toolbar is available on the **View > Toolbars** menu. When checked, the Window – Layout toolbar displays. Read more about the Window Painter in Window Painter.

To display or hide the toolbar, select it from the Toolbars menu as described in AppBuilder Toolbars.

Window – Layout toolbar



To manipulate an object from the Window – Layout toolbar, select one or more objects on a window, then click a layout button in the Window –

Layout toolbar.

In addition to using the toolbar, you can also manipulate window objects with any of the options on the **Layout** pull-down menu. Select the object from the window in which you are working, click **Layout** and select the layout action you want to perform.

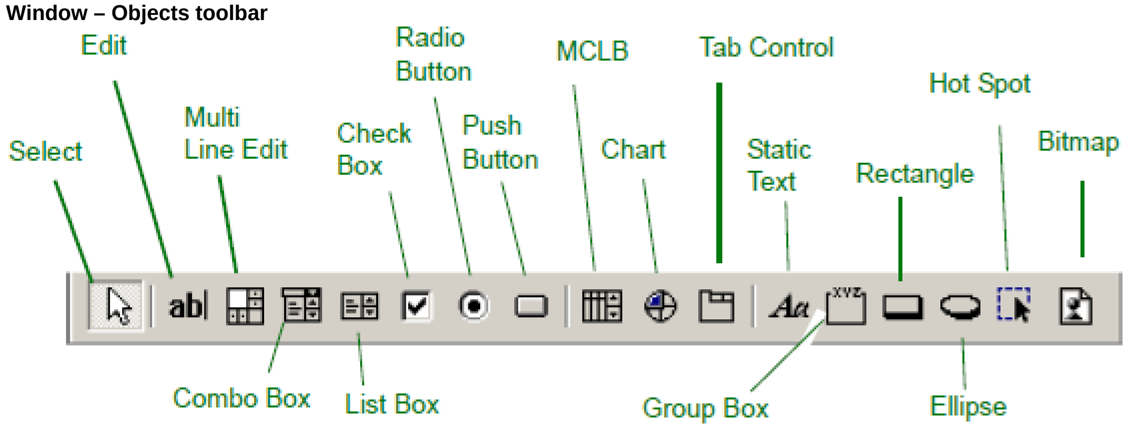
Window – Objects Toolbar

Use the Window - Objects toolbar (shown in Window – Objects toolbar) to quickly add objects to a window when developing the interface of an application. The Window – Objects toolbar is only available when you are creating or editing a window. To access the Window Painter tool, do the following:

1. Click **File > New.**
   The Create New window is displayed.
2. Select **Window** and click **OK**.
   A blank window tab displays, and the Window – Objects toolbar becomes available on the **View > Toolbars** menu.
3. Check Window – Objects to display the Window – Objects toolbar. Read more about Window Painter in Window Painter.

To display or hide the toolbar, select it from the Toolbars menu as described in AppBuilder Toolbars.

Window – Objects toolbar



To add an object from the Window – Objects toolbar, click an object button, then click in the Window Painter tab. In addition to using the toolbar, you can also add window objects by selecting the window to give it focus; then, select from the **Insert** menu the item you want to insert.

Window Flow Diagram Toolbar

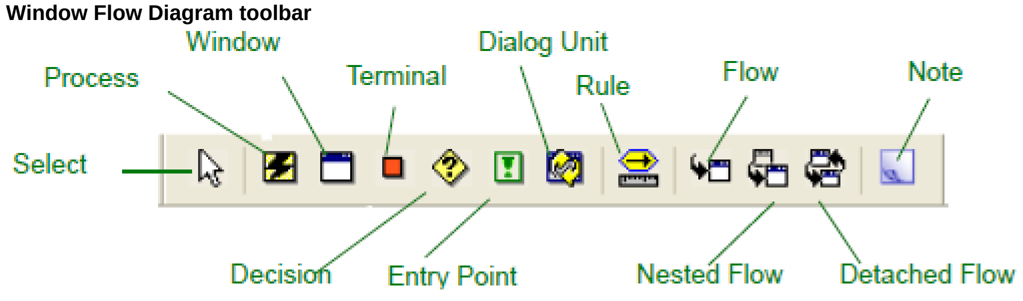
Use the Window Flow Diagram toolbar when designing the flow of windows for the application. This toolbar is only available when you are using the Window Flow Diagram tool. To access the Window Flow Diagram tool, do the following:

1. Select **File > New**.
   The Create New window is displayed.
2. Click **Window Flow Diagram**.
3. Click **OK**.
   The Window Flow Diagram tab displays.

The Window Flow Diagram toolbar is available on the **View > Toolbars** menu. When checked, the Window Flow Diagram toolbar is displayed. Read more about the Window Flow Diagram in Modeling Tools.

To display or hide the toolbar, select it from the Toolbars menu as described in AppBuilder Toolbars.

Window Flow Diagram toolbar



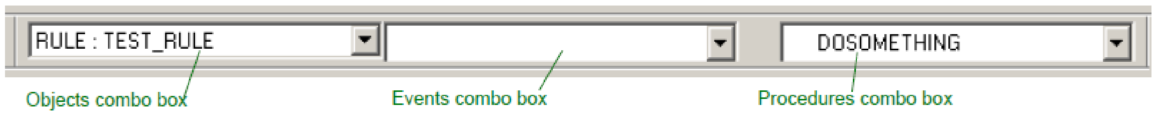
Text Editor - Event Wizard Toolbar

Use the Text Editor - Event Wizard toolbar to apply events and event procedures to objects in Windows and Rules. This toolbar is only available when you are using the Rule Painter tool. To access the Event Wizard toolbar, do the following:

1. Open Rule Painter.
2. Right-click the main toolbar at the top of the interface and select **Event Wizard Toolbar**.
   The Event Wizard toolbar displays.

The Event Wizard toolbar is also available on the **View > Toolbars** menu. When checked, the Event Wizard toolbar is displayed.

Event Wizard Toolbar



To display or hide the toolbar, select it from the Toolbars menu as described in AppBuilder Toolbars.

The Event Wizard toolbar has three combo boxes: Objects, Events, and Procedures.

Objects Combo Box

This combobox holds all controls in the child window of the current rule, including the child window and the current rule itself. Selecting an object refreshes the event combo box.

Events Combo Box

TheEvents combo box holds all valid events for the selected object. If an event is already defined in the rule, this event is displayed in bold font; otherwise, it is displayed in regular font. If the event is defined in a particular environment block, that environment name is appended to the event name. If you select an event that is already defined in the rule source, Rule Painter jumps to that definition. If you select an undefined event, Rule Painter creates a new event procedure in the rule. You can choose the environment block where the procedure will be defined.

Procedures Combo Box

TheProcedures combo box holds all procedures in the current rule. If a procedure is defined in a certain environment block, that environment name is appended to the procedure name. An icon is used to distinguish event procedures. Invalid event procedures are grayed out. Selecting an item in the combo box brings you to the procedure definition in the rule source.

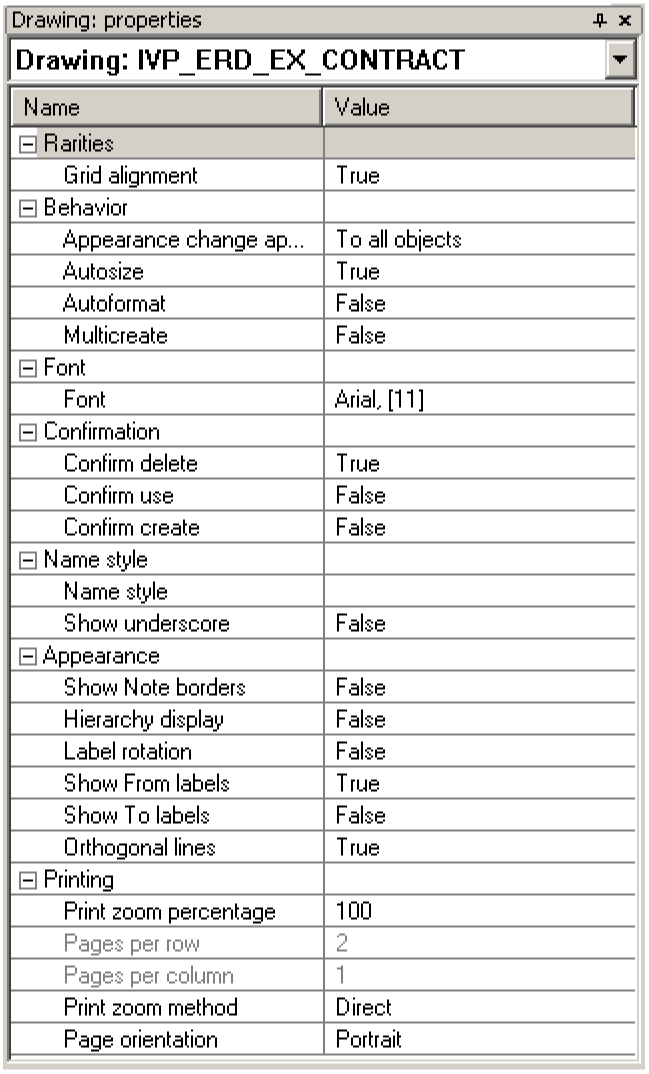
Drawing: properties Window

You can display the Drawing: properties window whenever you have an open drawing in the work area, like a database diagram, a window flow diagram, an ERD, a process dependency diagram (see Drawing: properties window sample).

The Drawing: properties window is available on the **View > Toolbars** menu. When checked, the Drawing: properties window is displayed.

To display or hide the window, select it from the Toolbars menu as described in AppBuilder Toolbars.

Drawing: properties window sample



Drawing: properties

Drawing: IVP_ERD_EX_CONTRACT

Name	Value
⊟ Rarities	
Grid alignment	True
⊟ Behavior	
Appearance change ap...	To all objects
Autosize	True
Autoformat	False
Multicreate	False
⊟ Font	
Font	Arial, [11]
⊟ Confirmation	
Confirm delete	True
Confirm use	False
Confirm create	False
⊟ Name style	
Name style	
Show underscore	False
⊟ Appearance	
Show Note borders	False
Hierarchy display	False
Label rotation	False
Show From labels	True
Show To labels	False
Orthogonal lines	True
⊟ Printing	
Print zoom percentage	100
Pages per row	2
Pages per column	1
Print zoom method	Direct
Page orientation	Portrait

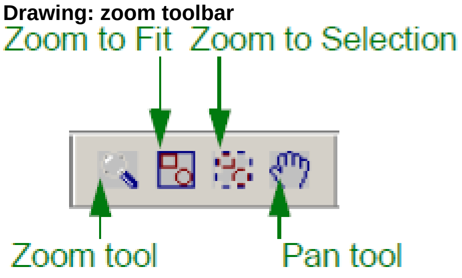
Drawing: zoom Toolbar

Use the Drawing: zoom toolbar to control the properties of a drawing (database diagram, window flow diagram, ERD, process dependency diagram). It is displayed when have open drawing is in the work area (see Drawing: zoom toolbar).

The Drawing: zoom toolbar is available on the **View > Toolbars** menu. When checked, the Drawing: zoom toolbar is displayed.

To display or hide the toolbar, select it from the Toolbars menu as described in AppBuilder Toolbars.

Drawing: zoom toolbar



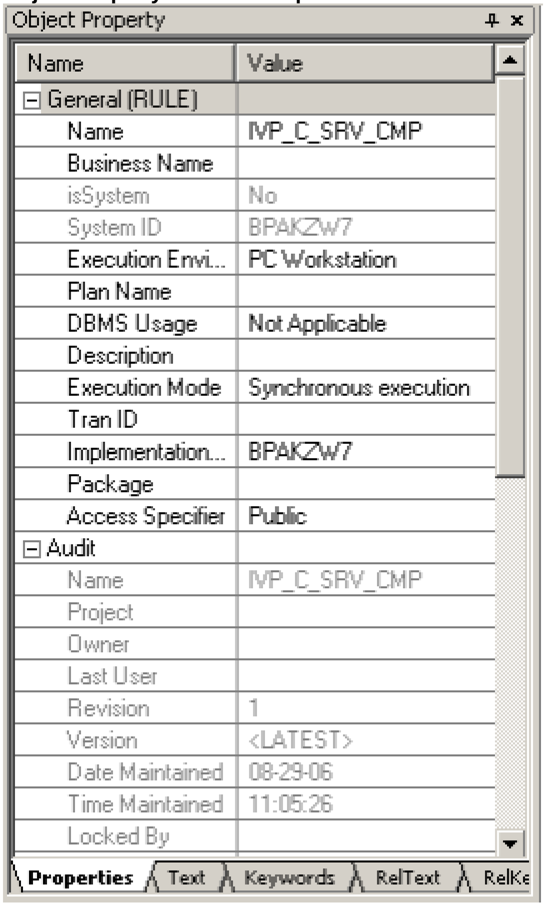
Object Property Window

Use the Object Property window to control the properties of the currently selected object. It is displayed when an open drawing is in the work area (see Object Property window sample).

The Object Property window is available on the **View > Toolbars** menu. When checked, the Object Property window is displayed.

To display or hide the window, select it from the **View** menu.

Object Property window sample



Object Property

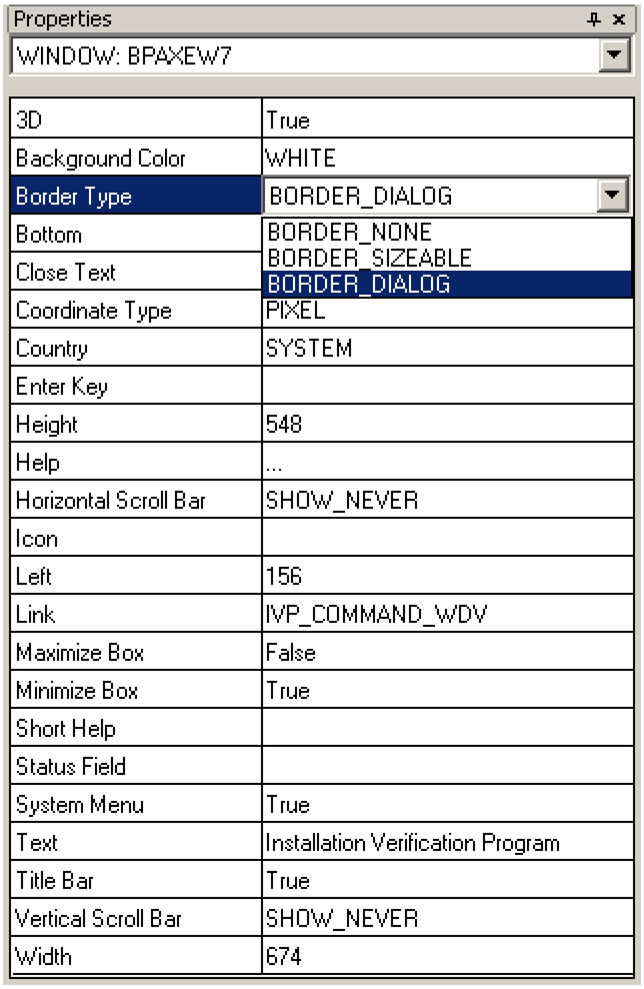
Name	Value
⊟ General (RULE)	
Name	IVP_C_SRV_CMP
Business Name	
isSystem	No
System ID	BPAKZW7
Execution Envi...	PC Workstation
Plan Name	
DBMS Usage	Not Applicable
Description	
Execution Mode	Synchronous execution
Tran ID	
Implementation...	BPAKZW7
Package	
Access Specifier	Public
⊟ Audit	
Name	IVP_C_SRV_CMP
Project	
Owner	
Last User	
Revision	1
Version	<LATEST>
Date Maintained	08-29-06
Time Maintained	11:05:26
Locked By	

Properties Window

Use the Properties window to control the properties of the currently selected window object. It is displayed when you click an object of a Window Painter window (see Properties window sample).

You can view and modify a property of an object in the Properties window, an example of which is shown in Properties window sample. From the Properties window, you can type a value or select a value from the drop-down list. After making necessary changes, click **File > Commit** from the Construction Workbench menu to commit changes to the repository.

Properties window sample



3D	True
Background Color	WHITE
Border Type	BORDER_DIALOG
Bottom	BORDER_NONE
Close Text	BORDER_SIZEABLE
	BORDER_DIALOG
Coordinate Type	PIXEL
Country	SYSTEM
Enter Key	
Height	548
Help	...
Horizontal Scroll Bar	SHOW_NEVER
Icon	
Left	156
Link	IVP_COMMAND_WDV
Maximize Box	False
Minimize Box	True
Short Help	
Status Field	
System Menu	True
Text	Installation Verification Program
Title Bar	True
Vertical Scroll Bar	SHOW_NEVER
Width	674

Do one of the following to open the Properties window, if it is not already displayed:

- Double-click the object in the window
- Right-click the object and select **Properties** from the pop-up menu
- Select **View > Properties** from the Construction Workbench menu
- Press **Alt+Enter**

See Understanding Common Window Object Properties for more details.

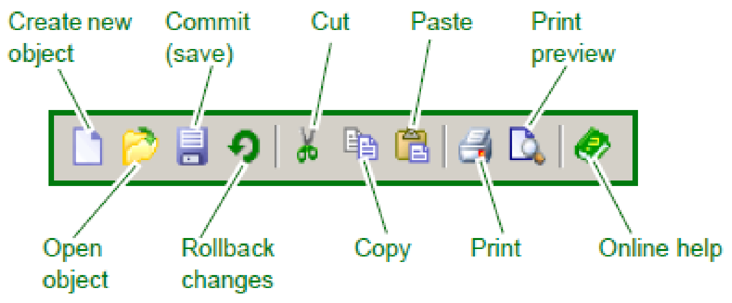
Standard File Operations

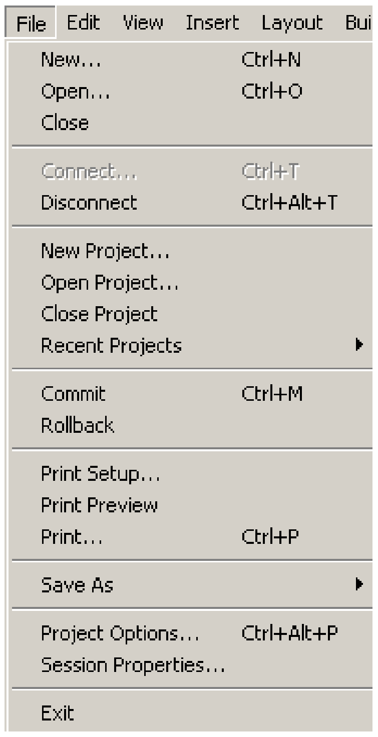
The following operations can be performed from either the standard toolbar or from the File menu, regardless of the Workbench tool you are using:

- Working with Projects and Project Options
- Opening an Existing Project
- Specifying Project Options
- Creating or Opening an Object
- Viewing Session Properties
- Committing or Rolling Back Changes
- Copying, Cutting, or Pasting
- Printing and Previewing
- Dragging and Dropping
- Closing a Project

Use the toolbar or the menus (shown in Standard toolbar and File menu) to perform standard desktop functions quickly.

Standard toolbar and File menu





Working with Projects and Project Options

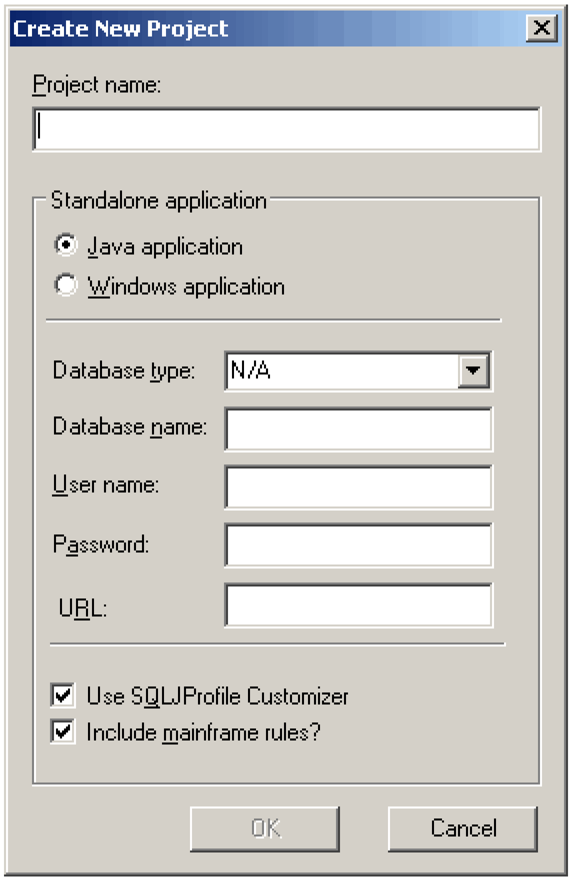
In an AppBuilder application, the *project* contains the business function and associated configuration. The project is the highest level of the application hierarchy. You can create a new project, open an existing project, and specify project options in AppBuilder.

Creating a New Project in AppBuilder

To create a new project, complete the following steps:

1. Select **File > New Project** from the Construction Workbench menu. The Create New Project window displays, as in Create New Project window.

   Create New Project window


Create New Project

Project name:

Standalone application

[✓ Java application] Windows application

Database type: N/A

Database name:

User name:

Password:

URL:

[✓] Use SQLJProfile Customizer

[✓] Include mainframe rules?

2. Type a unique name that is not already in the repository in the Project Name field.
3. Select the Java or Windows radio button for the environment for which you want to prepare the rules and application.
4. Select the database type. The available options are: N/A, DB2/UDB, Oracle, SQL Server.
5. Type the name of the database in the Database name field.
6. Type your username and password in the User name and Password fields.
7. Specify whether or not the application uses a database local to the machine.
8. Check the boxes specifying whether or not the application uses SQLJProfile Customizer if you are developing a Java application and whether or not you want to include mainframe rules to be prepared locally on your PC.
9. Click **OK** to accept the changes.

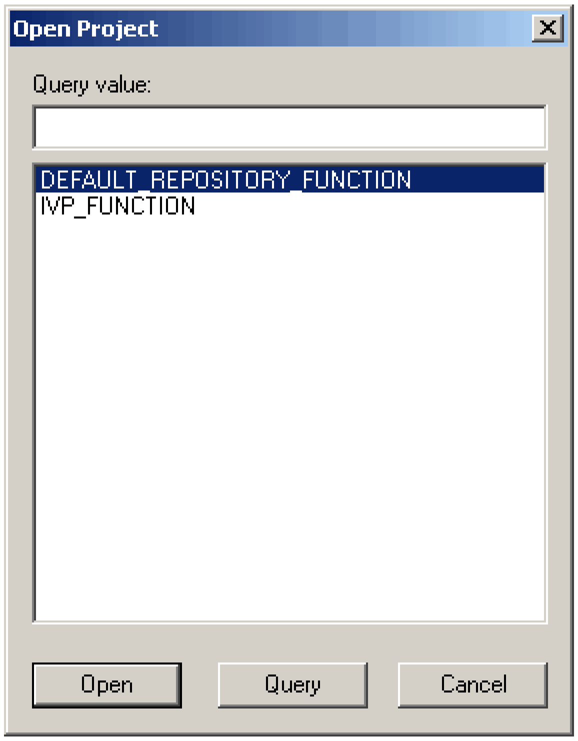
If there is not an open project, the preparation options default to the values specified on the Prepare section of the Workbench Options window. To see these default settings, select **Tools > Workbench Options**, then click **Prepare** in the Workbench Options window. If no project is open, use these options. You can find project options for an open project under **File > Project Options**
.
AppBuilder automatically adds a Function as a child of the project in the hierarchy.

Opening an Existing Project

To open an existing project, complete the following steps:

1. Select **File > Open Project** or click the **Open Project** button in the Project toolbar.
   The Open Project window displays.
2. Click **Query** to display a list of projects in the repository.
3. Select the project to open and click **Open** . See Open Project window after querying the existing projects.

Open Project window after querying the existing projects



DEFAULT_REPOSITORY_FUNCTION
IVP_FUNCTION

If a project has been imported from another repository, the Specifying Project Options project options for more information.

AppBuilder remembers the most recent projects. Select from the list in **File > Recent Projects** to open a recently used project.

Specifying Project Options

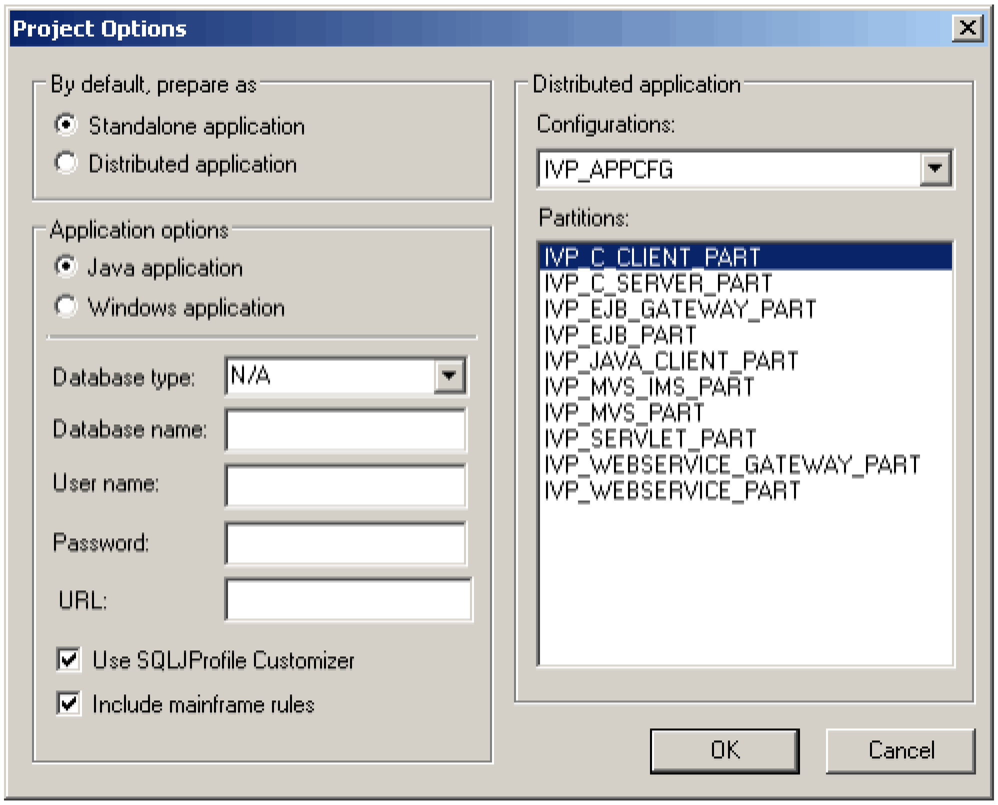
To specify project options for an open file, complete the following steps:

1. Do one of the following to open the Project Options dialog:

   > ⚠ If there is no project open, the Project Options item on the file menu is grayed out, and the preparation options are taken from **Tools > Workbench Options > Prepare** . See Workbench Options.

   - Click the **Project Options** button in the Project toolbar.
   - Select **File > Project Options** from the Construction Workbench menu.
   - From the hierarchy window, right-click the Project object and select **Project Options** from the pop-up menu.
   - From the Configuration Hierarchy window, right-click the Project object and select **Project Options** from the pop-up menu.

   Project Options dialog



Database type: N/A
Database name:
User name:
Password:
URL:
[✓] Use SQLJProfile Customizer
[✓] Include mainframe rules

IVP_APPCFG

IVP_C_CLIENT_PART
IVP_C_SERVER_PART
IVP_EJB_GATEWAY_PART
IVP_EJB_PART
IVP_JAVA_CLIENT_PART
IVP_MVS_IMS_PART
IVP_MVS_PART
IVP_SERVLET_PART
IVP_WEBSERVICE_GATEWAY_PART
IVP_WEBSERVICE_PART

2. Use this table to modify the project options:

Project Options window fields

Option	Description
By default prepare as	Specify if this is **Standalone** or **Distributed** application.
Standalone application	Specify if the application runs in a **Java** or **Windows** environment and provide the necessary database information. These fields apply only if **By default prepare as** = **Standalone** .
Distributed application	Select the default configuration and partitions to use when preparing the application. These fields apply only if **By default prepare as** = **Distributed** .

3. Type the necessary information in the fields and click **OK** to accept the changes.

Creating or Opening an Object

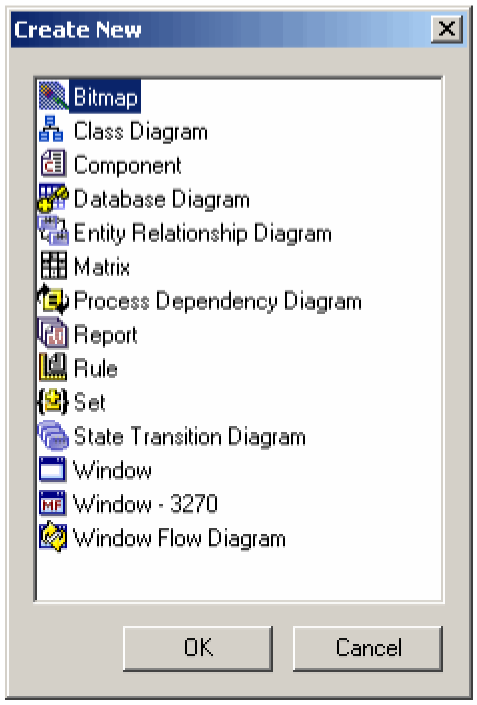
From the Hierarchy window you can either insert a new object or open an existing object.

- Creating a New Object
- Opening an Existing Object

Creating a New Object

To create a new object, select **File > New** . The Create New window is displayed.

Create New window



The following table lists the possible AppBuilder object types and where in this manual you can find information about how to use each object type.

Object tool summary

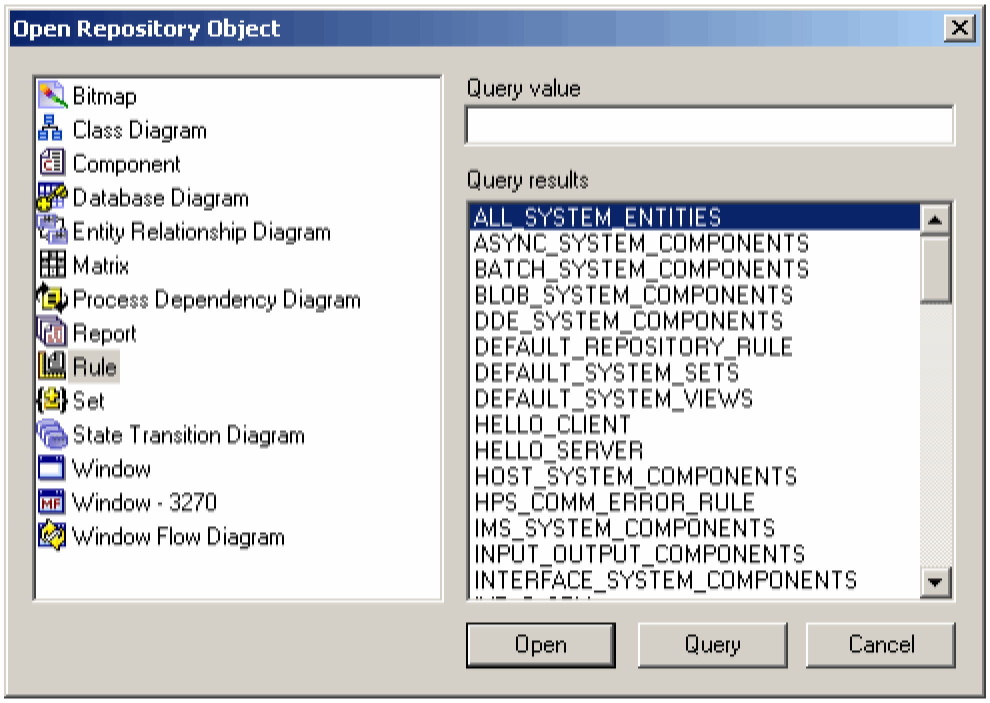
Object Types	**Tool Used**	**Reference**
Bitmap	Bitmap Viewer	See Bitmap Viewer.
Class Diagram	Class Diagram	See *Developing Applications Guide*.
Component	Component Painter	See Component Painter.
Database Diagram	Database Diagram	See Modeling Tools.
Entity Relationship Diagram	Entity Relationship Diagram	See Modeling Tools.
Matrix	Matrix Builder	See Matrix Builder.
Process Dependency Diagram	Process Dependency Diagram	See *Developing Applications Guide*.
Report	Report Painter and Report Writer	See *Reports Guide*.
Rule	Rule Painter	See Rule Painter.
Set	Set Builder	See Set Builder.
State Transition Diagram	State Transition Diagram	See *Developing Applications Guide*.
Window	Window Painter	See Window Painter.
Window - 3270	3270 Window Painter	See Window Painter.
Window Flow Diagram	Window Flow Diagram	See *Developing Applications Guide*.

Opening an Existing Object

To open an existing object, complete the following steps:

1. Select **File > Open**. The Open Repository Object window displays Open Repository Object window.

Open Repository Object window



ALL_SYSTEM_ENTITIES
ASYNC_SYSTEM_COMPONENTS
BATCH_SYSTEM_COMPONENTS
BLOB_SYSTEM_COMPONENTS
DDE_SYSTEM_COMPONENTS
DEFAULT_REPOSITORY_RULE
DEFAULT_SYSTEM_SETS
DEFAULT_SYSTEM_VIEWS
HELLO_CLIENT
HELLO_SERVER
HOST_SYSTEM_COMPONENTS
HPS_COMM_ERROR_RULE
IMS_SYSTEM_COMPONENTS
INPUT_OUTPUT_COMPONENTS
INTERFACE_SYSTEM_COMPONENTS

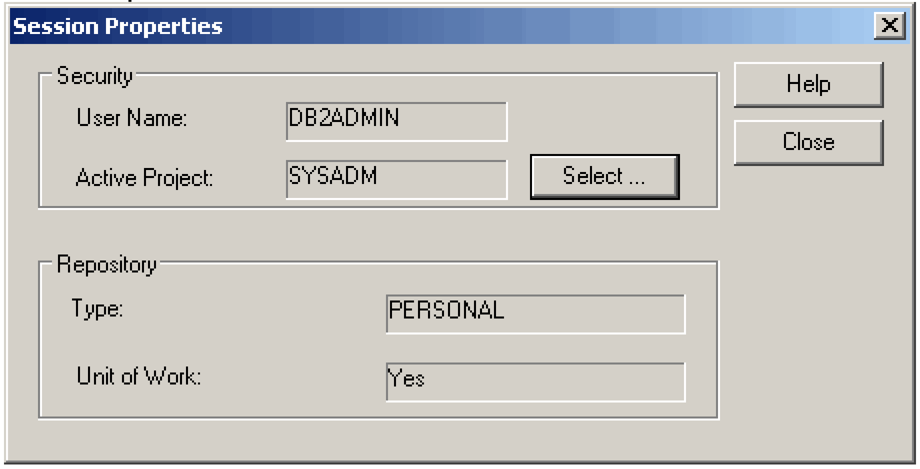
1. Select an object type from the left pane and click **Query** . A list of existing objects for that type in the repository to which AppBuilder is currently linked is displayed in the right pane under Query Results. You can also query a specific object by typing the object's full name, the first character(s) of the name, or you can use the * to enter a search criteria in the Query Value field.
2. From the right pane, select the object that you want to open.
3. Click **Open** .

Viewing Session Properties

To specify the project you want to work on in the current session and view version information, use the Session Properties window.
To view or change your session properties, complete the following steps:

1. From the Construction Workbench, select **File > Session Properties**. (See Session Properties window). You can also access session properties by selecting **Repository > Session Properties** from Repository Administration. Read more about Repository Administration in the *Repository Administration Guide for Workgroup and Personal Repositories*.

   Session Properties window



User Name: DB2ADMIN
Active Project: SYSADM
Type: PERSONAL
Unit of Work: Yes

2. Use the following table as a guide to set the session properties:

   Session Properties window fields

Option	Description
User Name	ID of the current user

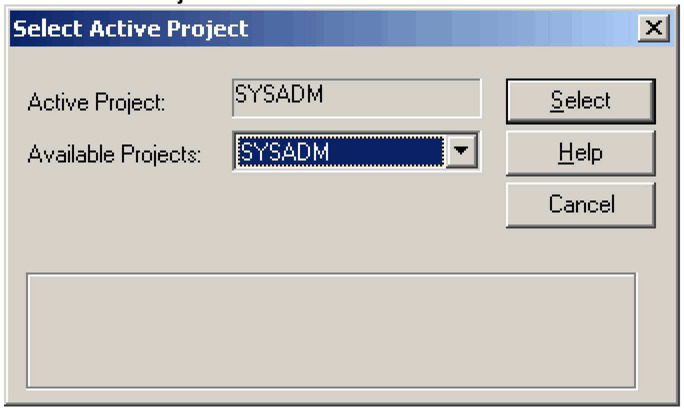
Active Project	Name of the active project in the repository. A project is a set of repository objects accessible only to members of authorized groups. The objects you create during the session belong to the selected project. Click **Select** to choose a project.
Type	The type of repository to which you are connected.
Unit of Work	If you have installed the Unit of Work feature on a Workgroup Repository, the Unit of Work is **WorkGroup** . If you did not install Unit of Work, the value here is **None** . If you have installed the Personal Repository, the value here is **Personal**.

3. Type the necessary information in the appropriate fields and click **Close** to accept the changes for the active project.

From the Session Properties window, you can select which project you want to work on.

1. Next to the Active Project field, click **Select** .
2. In the **Select active project** window, select a project from the drop-down list and click **Select** .

Select Active Project window



If you select a different project to be the active project, a new session must be established to enable the active project. After making this selection, restart the Construction Workbench.

Committing or Rolling Back Changes

While working in Construction Workbench, the changes that you make to your application are not stored in the repository until you either commit those changes or rollback all those changes.
Committing and rolling back apply to all the changes made in that session, regardless of which tool you use.

Committing changes stores all changes to the project in the repository. When you are ready to commit your changes to the repository, do one of the following:

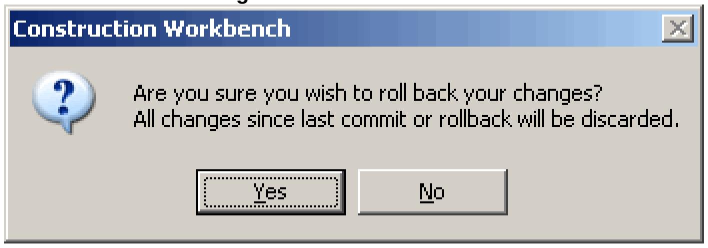
- Select **File > Commit** .
- Press **Ctrl+M** .
- Click the **Commit** button in the Standard toolbar.

Rolling back changes cancels all the changes made since the last commit. To roll back changes, do one of the following:

- Select **File > Rollback** .
- Click the **Rollback** button in the Standard toolbar.

A dialog appears that asks you if you are certain you want to roll back changes.

Rollback Confirm dialog



Are you sure you wish to roll back your changes?
All changes since last commit or rollback will be discarded.

Use the manual rollback command with a Workgroup or a Personal Repository when AppBuilder is not releasing object locks and it is evident that these objects are not locked by any other developer. The manual rollback command unlocks all repository objects based on the Session ID.

To perform a manual rollback, type the following at the command line:
```
gresrvnt -r -i<Session_ID> -u <userid> -p <password>
```
where:

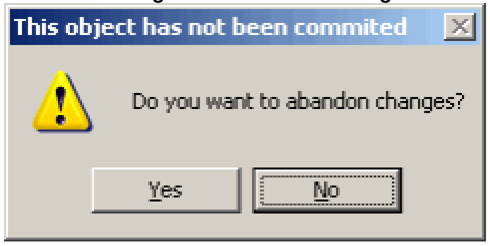
- <Session_ID> is the session containing uncommitted work to be rolled back.
- <userid> and <password> must have administrator level authority to the repository database.

For example:
```
gresrvnt -r -i4 -u SERMKP -p XXXXXXX
```

There is a space between the `-u` and the `<userid>` and a space between the `-p` and `<password>`.

Some editing tools, such as the Rule Painter, or the design tools, such as the Entity Relationship Diagram, also save information in an associated file while you are working. When you close the tool, AppBuilder displays a confirmation dialog that asks whether to abandon changes. The following figure shows the Abandon changes confirmation dialog.

Abandon changes confirmation dialog

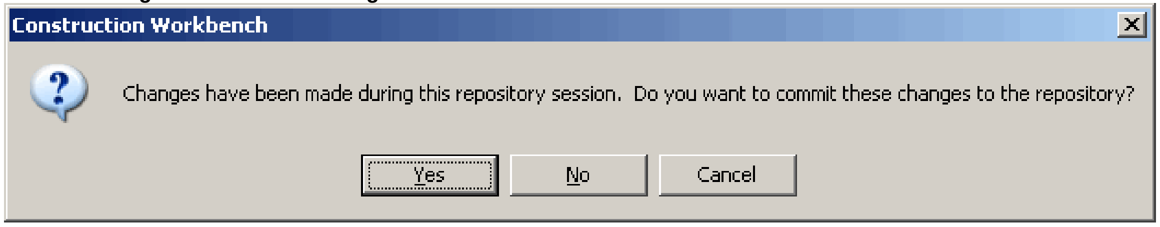

This object has not been commited

Do you want to abandon changes?

Not all tools offer a confirmation. If the changes are automatically placed in the repository as with changes made from the Hierarchy window, on exit, you do not get confirmation on abandoning your changes.

When you exit the Construction Workbench after making changes that have not been committed, AppBuilder displays a confirmation dialog that asks if you want to commit the changes to the repository before exiting. The following figure shows the Commit changes confirmation dialog.

Commit changes confirmation dialog


Construction Workbench

Changes have been made during this repository session. Do you want to commit these changes to the repository?

Copying, Cutting, or Pasting

In many of the tools you can copy or cut objects and string values to the Windows clipboard, either for pasting in the same tool, another Workbench tool, or in other applications. For example, you can copy and paste objects from one Hierarchy tab to another or from one place in a tab to another place in that tab. As another example, from the Rule Painter, you can select source code and copy it for pasting into another Rule Painter window or into an e-mail to a co-worker.

In the Design Tools, like the Entity Relationship Diagram, you can copy objects in the diagram and paste them into other places on the same diagram or paste them to another diagram.

Printing and Previewing

In many of the tools you can print what is displayed in the window of that tool. When printing is possible, previewing what is to be printed is also possible.

- To print, click **File > Print** or click the Print button on the Standard toolbar. The standard Print dialog displays.
- To preview, click **File > Print Preview** or click the Print Preview button on the Standard toolbar. The printout is displayed in a tab on work area with options for you to zoom in or out, print, or close the preview screen.
- To select the printer and the printer settings, click **File > Print Setup**. The standard Print Setup window displays.

In some of the tools, the page header contains the name of the object and the page footer contains the page number. Different tools have different headers and footers. Use **Print Preview** to see what is printed before you send the page to a printer.

Dragging and Dropping

You can drag objects from the Hierarchy window and drop them into the work area or the specific tool window. Use the **Ctrl** key to copy objects rather than move them.

You can drop objects from the Prep Status tab in the Repository or place Inverted tabs under the selected item when dropping. Relationships are created in the same fashion as inserting objects. Within the Hierarchy window, you can also drag and drop from one level in the application hierarchy to another level.

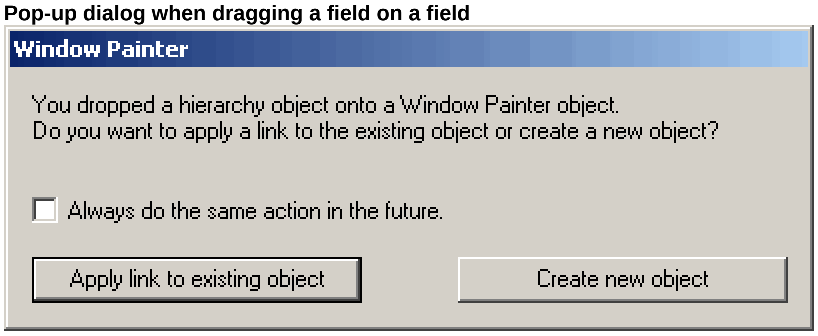
You can drag and drop some objects from the Hierarchy window to an open document as a shortcut for inserting the objects; however, not all windows have the drag and drop functionality. Objects cannot be selected from within the tools in the work area.

You cannot drag and drop a function or rule to Window Painter or a view to an edit control. When you attempt to drag and drop any item that is not allowed, such as a function or rule, the cursor changes to a circle with a slash through it.

Dragging the names of views and fields from the Hierarchy window, Prep Status tab, and Prep List tab, and dropping them into the Rule Painter displays the object name in the Rule Painter using either the dot notation or the standard notation based on the setting in the Text Editor Options of the Workbench Options window. This is not scoped to views that are within the rule's domain. Objects cannot be selected from within the Rule Painter in the work area. Read more about Rule Painter in Rule Painter.

You can drop several types of objects into the Window Painter from the Hierarchy window. You can drag an object from the application hierarchy in the Hierarchy window and drop it into the Window Painter tab. You can also drag window objects from the Prep Status tab and the Prep List tab to the work area. If the object is a window, Window Painter automatically opens. For example, you can drag a Field object that has a View for a parent in the hierarchy into the window, and a message box that provides you with the choice to either create a link to an existing object or to create a new object appears.

Pop-up dialog when dragging a field on a field


Window Painter

You dropped a hierarchy object onto a Window Painter object.
Do you want to apply a link to the existing object or create a new object?

Always do the same action in the future.
Apply link to existing object
Create new object

If you choose to create a link, it becomes an Edit Field and is linked to that view.

Read more about Window Painter in Window Painter.

To submit an object for prepare, drag the object from the Hierarchy window to the Prep Status tab of the Output window. The preparable objects are added to the Prep Status tab and submitted for preparation. The same configuration rules apply as when doing a prepare from the Build menu ? that is, stand-alone versus distributed or Project tab versus Configuration tab. Non-preparable objects cannot be dragged into the Prep Status tab for preparation. For example, you cannot drag an ENTITY object from the Hierarchy window to the Prep Status tab. For more information about the Output window, see Output Window. For more information about preparing an object, see the *Deploying Applications Guide* .

You can select, drag, and drop object names in the Prep Status tab into other tool windows, such as the Rule Painter. When you drop objects from the Prep Status tab in the Repository or Inverted tabs, they are placed under the selected item. Relationships are created in the same manner as with an insert of an object.

When you drag an object from the Hierarchy to the Prep List tab of the Output window, the preparable objects are added to the Prep List window. The same configuration rules apply as when doing a prepare from the Build menu ? that is, stand-alone versus distributed or Project tab versus Configuration tab. Non-preparable objects cannot be dragged into the Prep List tab for prepare. Object names in the Prep Status List tab can be selected, dragged, and dropped into other tool windows, such as the Rule Painter.

Since the Prep List tab does not allow duplicates, a new entry is not created if the same job already exists in Prep List. For more information about the Output window, see Output Window. For more information about preparing objects, see the *Deploying Applications Guide* .

Closing a Project

To close an AppBuilder project, click **File > Close Project**.

Workbench Options

Use Workbench Options window to specify the various options for each tool in the Construction Workbench. To access these options, click **Tools**

> **Workbench Options** from the Construction Workbench menu bar. The Workbench Options dialog displays with the following sections:

- General Section
- Prepare Section
- Script Section
- Tools Section

General Section

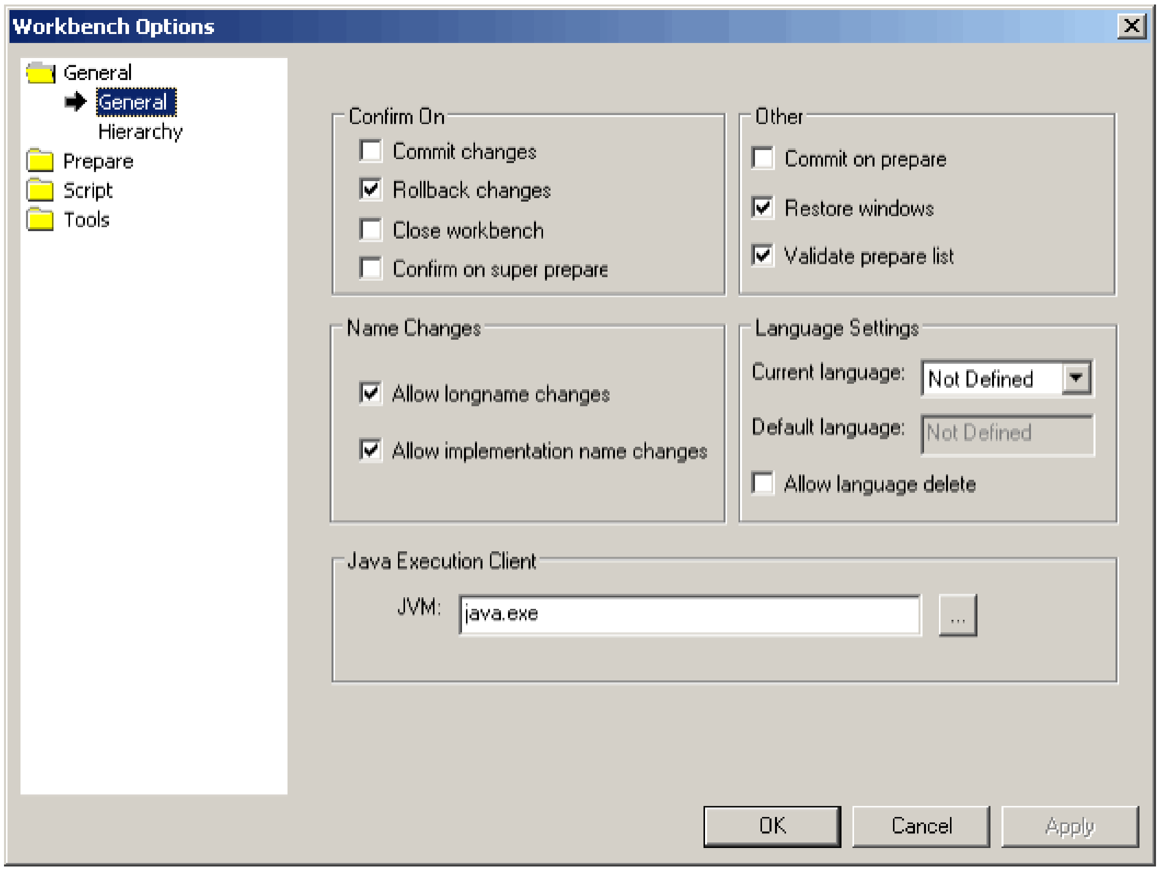
When you open the Workbench Options dialog, the General section displays by default. It contains two sets of options:

- General Options
- Hierarchy Options.

General Options

To get to the General options of the Workbench Options dialog, select **Tools** > **Workbench Options**. The General section is displayed with the General set of options selected as shown in General options. From here you can change your confirmation options, name change options, language settings, Java Execution Client information, and other options.

General options



Confirm On
- [ ] Commit changes
- [✓] Rollback changes
- [ ] Close workbench
- [ ] Confirm on super prepare

Other
- [ ] Commit on prepare
- [✓] Restore windows
- [✓] Validate prepare list

Name Changes
- [✓] Allow longname changes
- [✓] Allow implementation name changes

Language Settings
- Current language: Not Defined
- Default language: Not Defined
- [ ] Allow language delete

Java Execution Client
- JVM: java.exe

General options

Option	Description
Confirm On ...	
Commit changes	Select this option to configure the system to provide confirmation before committing changes to the repository.
Rollback changes	Select this option to configure the system to provide confirmation before rolling back changes to the repository.

Close Workbench	Select this option to configure the system to provide confirmation before closing AppBuilder.
Confirm on super prepare	Select this option to provide confirmation when doing a super prepare.
Other ...	
Commit on prepare	Select this option to configure the system to commit the changes when it runs a prepare.
Restore windows	Select this option to configure the system to remember which windows you had open the last time you used Construction Workbench so you can easily pick up where you left off. If this option is not selected, the Construction Workbench opens with the default (hierarchy and status window) and no additional tools windows are opened.
Validate prepare list	Select this option to validate the list of preparation objects.
Name Changes ...	
Allow longname changes	Select to allow the Implementation field of the Object Property window for any object to be displayed and editable. Otherwise it is grayed out and not editable. (This can be used late in the development cycle if you do not want the name of an entity to change.)
Allow implementation name changes	Select to allow the Implementation field of the Object Property window for any object to be displayed and editable, otherwise it is grayed out and not editable (This is used when, for example, late in the development cycle you do not want the name of an entity to change.)
Language Settings	
Current Language	Specifies the language of preparation and dictates that windows or sets automatically open in this language. When you have created at least one new language and the current language is defined, the AppBuilder MLUI functionality becomes active in the Construction Workbench. For more information, see "Setting the Workbench Options for MLUI" in the *Multi-Language User Interface Guide*.
Default Language	Used when converting an existing application to an MLUI application, the Default Language is the language of your existing application. For more information, see "Setting the Workbench Options for MLUI" in the *Multi-Language User Interface Guide*.
Allow Language Delete	This option is an additional safeguard against inadvertently deleting languages from the repository. For more information, see "Setting the Workbench Options for MLUI" in the *Multi-Language User Interface Guide*.
Java Execution Client...	
JVM:	Specify the Java Virtual Machine for the Java execution client

After you make changes, do one of the following:

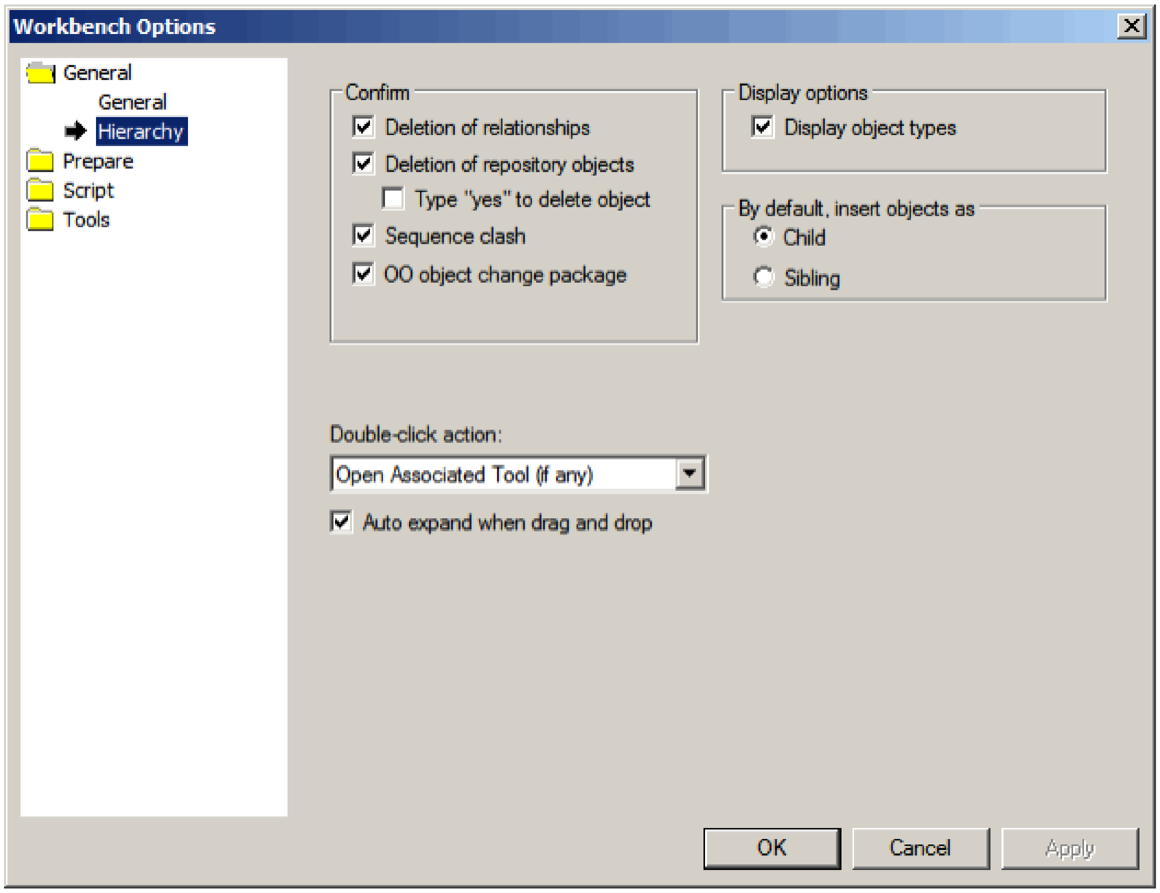
- Click **Apply** to apply the changes for all the options and keep the Options window open.
- Click **OK** to apply the changes for all the options and close the Options window.
- Click **Cancel** to close the Options window without applying any changes.

Hierarchy Options

Use the Hierarchy set of options to control the behavior of the hierarchy tree of your project. The options specified in this set define the look and feel of your hierarchy. To get to the Hierarchy set of the Workbench Options window, select **Tools** > **Workbench Options**, then click **Hierarchy** in General section. The Hierarchy tab is displayed as shown in Hierarchy options.

By default, any time you drag objects in the Hierarchy tab and drop the objects into another location, the hierarchy tree expands. If you do not want objects to expand automatically when you drag them from the hierarchy tree and drop them somewhere else, uncheck the Auto expand when drag and drop choice under the Double-click action field.

Hierarchy options



Confirm
- [✓] Deletion of relationships
- [✓] Deletion of repository objects
  - [ ] Type "yes" to delete object
- [✓] Sequence clash
- [✓] OO object change package

Display options
- [✓] Display object types

By default, insert objects as
- ● Child
- ○ Sibling

Double-click action: Open Associated Tool (if any)

- [✓] Auto expand when drag and drop

Hierarchy options

Option	Description
Confirm	Select this option to have AppBuilder prompt for confirmation when deleting relationships between repository objects, deleting repository objects, or for sequence clashes. The confirm options are: • Deletion of relationship – check this box in order to receive a confirmation message whenever you are deleting a relationship. • Deletion of repository objects – check this box in order to receive a confirmation message whenever you delete repository objects. Since these objects might be important, you have an extra option: Type "yes" to delete object – you can type YES and click OK. • Sequence clash – when the hierarchy sees a sequence clash, you can configure the Construction Workbench to ask if renumbering is OK. • OO Object change package: check this box in order to receive a confirmation message whenever you are changing the package for an OO object. Usually, when you insert a class into a package that previously belonged to another package, AppBuilder automatically moves the class into the new package without informing the user. You can use this option to be notified whenever this happens.
Display Options	Check this box if you want the system to display the appropriate object icon next to the object name.
By default, insert objects as	Select this option to specify the default when objects are inserted: **Child** or **Sibling**.
Double-click action	Specify the default action when you double-click an object in a hierarchy: • Open associated tool (if any) • Show object properties • Show relationship properties • Expand/collapse object

Auto expand when drag and drop	Check this box if you want to automatically expand objects in the hierarchy tree when you drag and drop them from the hierarchy tree to another location.

After you make changes:

- Click **Apply** to apply the changes for all the options and keep the Options window open.
- Click **OK** to apply the changes for all the options and close the Options window.
- Click **Cancel** to close the Options window without applying any changes.

Read more about the Hierarchy window in Using the Hierarchy Window.

Prepare Section

These options are associated with the AppBuilder Prepare function. To get to the Prepare section of the Workbench Options window, select **Tools** > **Workbench Options**, then click on **Prepare** in Workbench Options window. The Prepare section contains three sets of options:

- Preparation Options
- Remote Preparation Options
- HTML Generation Options.

Preparation Options

Select **Tools** > **Workbench Options**, then click on **Prepare** in Workbench Options window. The Preparation set of options is displayed as shown in Preparation options. AppBuilder allows you to specify the following options during preparation:

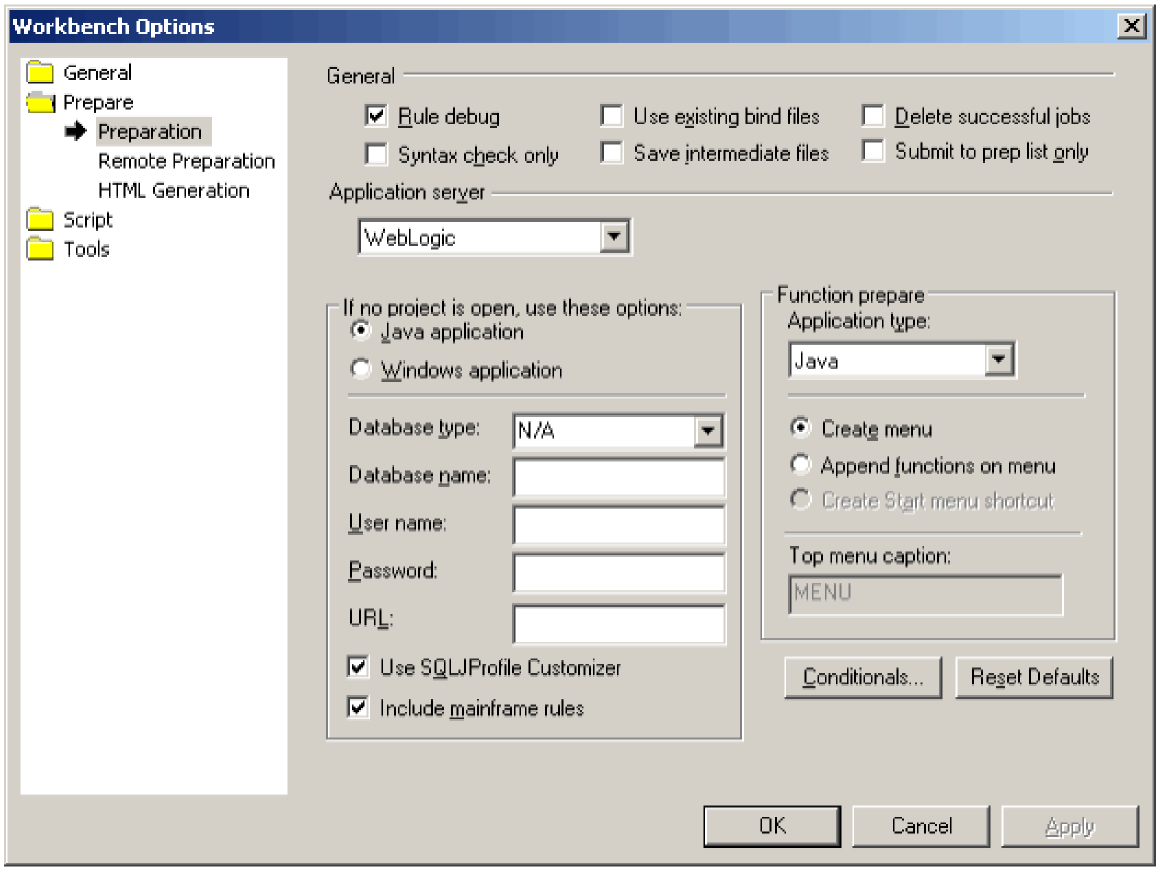
- Extent of debugging
- Type of function object
- Local database
- Default settings for standalone application
- Application server (Web server).

The options you can set in the Preparation section include:

- Preparation: General Settings
- Preparation: Application Server Settings
- Preparation: No Project Open Settings
- Preparation: Function Prepare Settings

Read more about the AppBuilder Preparation function in the *Deploying Applications Guide*.

Preparation options



General

- [✓] Rule debug
- [ ] Syntax check only
- [ ] Use existing bind files
- [ ] Save intermediate files
- [ ] Delete successful jobs
- [ ] Submit to prep list only

Application server: WebLogic

If no project is open, use these options:

- (●) Java application
- ( ) Windows application

Database type: N/A
Database name:
User name:
Password:
URL:

- [✓] Use SQLJProfile Customizer
- [✓] Include mainframe rules

Function prepare

Application type: Java

- (●) Create menu
- ( ) Append functions on menu
- ( ) Create Start menu shortcut

Top menu caption: MENU

After you make changes:

- Click **Apply** to apply the changes for all the options and keep the Options window open.
- Click **OK** to apply the changes for all the options and close the Options window.
- Click **Cancel** to close the Options window without applying any changes.

Preparation: General Settings

The settings that relate to preparation of a rule are check boxes in the General area at the top of the Preparation set of options. These are summarized in Preparation: general options.

Preparation: general options

Option (check box)	Description
Rule debug	For rule preparation, select this option to save the rule's source code so that the tool RuleView can use it. (For more information about debugging, refer to *Debugging Applications Guide*.) For function preparation, select this option to prepare the rules beneath the function (in the hierarchy) for debugging and to add breakpoints as it compiles the rules. Select this option while preparing to a Java thick client as Java execution client loads *.asc file in debug directory.
Use existing bind file	Select this option to suppress the regeneration of bind files. This allows you to save time when preparing if you have not altered a rule's relationships with other objects or views, for example, when you fix a syntax error. If a blind file does not exist, preparing with this option checked generates a new bind file.
Delete successful jobs	Select this option to delete successful preparation jobs automatically. Successful jobs are cleared from the **Prep Status** tab. The advantage of displaying only the unsuccessful jobs is that you know immediately what needs to be fixed.
Syntax check only	Select this option if you want AppBuilder to scan the rule code for syntax errors without performing subsequent steps (See Results of different options on preparing PC and Host rules).
Save intermediate files	Select this option to have AppBuilder save intermediate files for debugging. These files are created for steps internal to preparation.
Submit to prep list only	Select this option to have AppBuilder not prepare the selected job but simply add it to the preparation list (in the Prep List tab of the Output window). Then all the jobs in the prep list can be prepared at once from the Output window.

Preparation: Application Server Settings

Select the application server if you are deploying a Java application to an application server. The following choices can be selected:

- WebLogic (based on BEA WebLogic)
- WebSphere (based on the WebSphere standard)
- Tomcat (based on Apache Tomcat)
- None

Preparation: No Project Open Settings

The settings in this area only apply if a project is not currently open. To access the Project Options settings for an open project, from the Hierarchy, right-click the Project object and select **Project Options**.

Preparation: No Project Open Settings

Option	Description
Java application or Windows application	Select whether the implementation is for Java or Windows. If there is no project open, this setting determines the Application Type field value in the Function Prepare group box. See Preparation: Function Prepare Settings.
Database name, Database type, User name and password	If you are using a database, enter the information here.
URL	Define path to database (for Java application only)
Use SQLJProfile Customizer	Specify whether or not the application uses DB2 profile customizer (for Java application only)
Include mainframe rules	Select this option when using an existing application that contains rules that execute on a mainframe and you want to prepare the entire application locally to test the business logic (See Results of different options on preparing PC and Host rules). This enables you to prepare the application including the rules that usually go to the mainframe. There are exceptions. For example, a rule that references a database on the mainframe cannot be prepared. If you deselect this box, the system only checks the syntax of the rules and provides warnings but does not prepare the rules. To prevent the preparation and verification of the mainframe rules, do one of the following: • If you *are* using a project and do not want to include mainframe rules in your local preparation, from the Project Options window, deselect this option. (Right-click the project in the Hierarchy window and select **Project Options**.) In addition, in your HPS.INI, set the flag `SKIP_MF_RULE_CHECK=Y`. • If you are *not* using a project and do not want to include mainframe rules in your local preparation, on the Workbench Options window, Preparation set of options, deselect this option. (Click **Tools** > **Workbench Options** > **Prepare**.) In addition, in your HPS.INI, set the flag `SKIP_MF_RULE_CHECK=Y`.

Preparation: Function Prepare Settings

You can only select one of these options described in Preparation: function prepare options.

Preparation: function prepare options

Option	Description
Application type	Specifies whether the implementation is for Java or Windows. This value is determined from the application type chosen under the "If no project is open, use these options" group box (when no project is open) or from the Project Options window (when a project is open). The application type is not set by the value in the Application Type field.
Create menu	This option specifies that a pull-down menu item (rather than a shortcut and Start menu item) is created for the latest prepared function. In execution mode, the menu is available. The application executes the same way, regardless of how it is started.
Append functions on menu	This option specifies that a pull-down menu choice is displayed in execution mode not only for the latest prepared function, but for every function previously prepared on this machine that are available. If selected, the generated menu includes all the available prepared functions; it appends all functions previously prepared on this machine to the menu, not just the latest. Menu choices are displayed in the order that their corresponding functions are prepared. If not selected, the generated menu (if the Create Menu option is selected) only includes the most recently prepared function, and the others are not displayed. If selected again, they are added back.
Create Start menu shortcut	For C Windows applications, this option specifies that a Start menu shortcut icon is created for the function being prepared. In execution mode, the Start menu icon is available. The application executes the same way, regardless of how it is started. This option is only supported for Windows applications, not for Java.

Top menu caption	This option specifies a top menu caption. Default is MENU.

Under Function prepare, when you select the Application type as either Java or Windows, the settings under that list box are associated with the chosen application type. The default options for the three radio buttons are the same for Java and Windows. That is why the **Apply** button does not get enabled if you only change the Application type. If you have different settings for Java and Windows, the radio buttons show your choice when you switch between Java and Windows.

These Preparation: Function Prepare Settings are independent of the Preparation: No Project Open Settings, which specify default options in case no project is opened. These Function prepare options determine the result at execution time.

Click **Conditionals** to add variable value conditionals to the preparation process. The Conditional Compilation window (shown in Conditional Compilation window) is displayed. The compilation conditionals can be added, edited, or deleted from the Conditional Compilation window. Macro values are validated against the [MacroDomains] section of HPS.INI. If the MACRO_NAME exists in the [MacroDomains] section, the value used in the rule is validated against the values in the domain, or the list of specified values. If the value is not in the domain, then the rule prepare fails with the following error message: XXXXX-W Value "French" is not listed in the domain for macro TARGET_LANGUAGE. For more information about using conditionals, refer to "Using Conditional Macros" in the *Rules Language Reference Guide*.

To add a condition, complete the following steps:

1. Click **New** and type the name of a variable and its value.
2. Click **OK**.

To change the value or the name of a variable, complete the following steps:

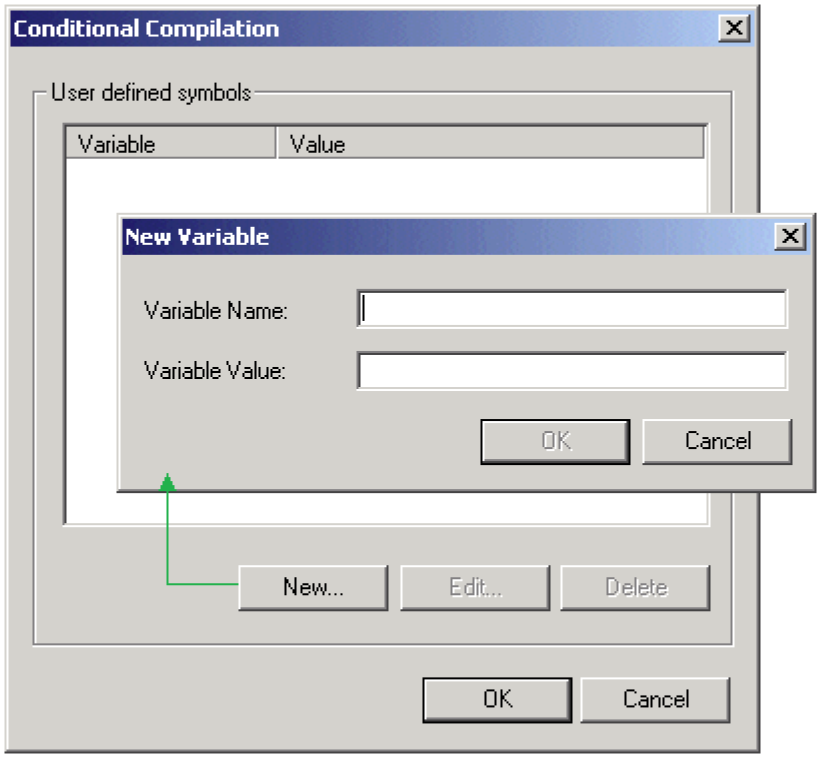
AppBuilder does not validate uninitialized variables before they are used.

1. Select the variable and click **Edit**.
2. In the Edit window, change the variable name or the value or both.
3. Click **OK**.

To remove a variable:

1. Select the variable from the list.
2. Click **Delete**.

Conditional Compilation window



3. Click **OK** to apply the changes and close this window.

In the Preparation section, click **Reset Defaults** to change all the values in the fields back to the original default settings.

Components in C language require header files.

Results of different options on preparing PC and Host rules shows the results of selecting or not selecting the **Syntax check only** and **Include mainframe rule** options on the Workbench Options Preparation section and setting the hps.ini **SKIP_MF_RULE_CHECK** setting to Y or N. The table compares preparing PC and Host Rules on the Project, Repository, and Configuration tabs on the Hierarchy window of the Construction Workbench.

Results of different options on preparing PC and Host rules

Workbench Options, Preparation set of options								
Syntax check only	X	X				X		X
Include Mainframe Rule	X		X				X	X
hps .ini setting								
SKIP_MF_RULE_CHECK	N	N	N	N	Y	Y	Y	Y
Results when Preparing from the Project and Repository tabs on the Hierarchy window								
PC Rule	1/no	1/no	3	3	3	1/no	3	1/no
Host Rule	1/no	1/yes	3	1/yes		2	3	1/no
Results when Preparing from the Configuration tab on the Hierarchy window								
PC Rule	1/no	1/no	3	3	3	1/no	3	1/no
Host Rule	1/no	1/no	3	3	3	1/no	3	1/no

1	=	Syntax check only
no	=	No warning message
yes	=	Warning message
2	=	Skip syntax check
3	=	Prepare

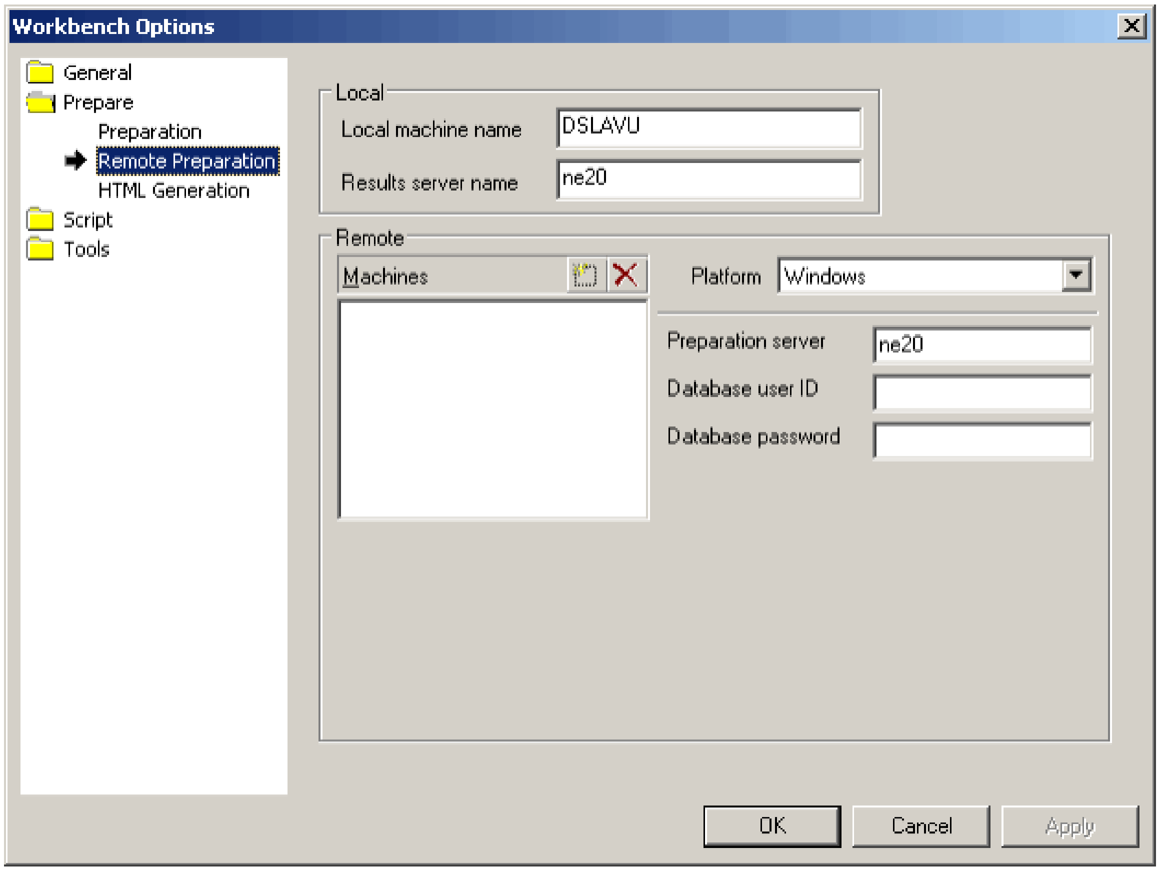
Remote Preparation Options

AppBuilder allows you to specify the options for remote preparation including relevant information about:

- Local machine
- Remote machines

To display to the Remote Preparation set of options in the Workbench Options window, select **Tools** > **Workbench Options**, then click **Prepare** > **Remote Preparation**. The Remote Preparation set of options is displayed as shown in Remote Preparation options.
Read more about performing preparation in the *Deploying Applications Guide*.

Remote Preparation options


Workbench Options

- General
- Prepare
  - Preparation
  - Remote Preparation
  - HTML Generation
- Script
- Tools

Local
Local machine name: DSLAVU
Results server name: ne20

Remote
Machines
Platform: Windows
Preparation server: ne20
Database user ID:
Database password:

OK | Cancel | Apply

In the Local area, type the local machine name and the name of the results server.

In the Remote area, you can add several remote machine names to the list and specify different server names and user identifiers. For each remote machine, specify various other parameters that depend on the machine and platform. Depending on the platform you select, the appropriate fields will display.

To specify a remote machine, complete the following steps:

1. Click the **Add** button (above the Machines list box) and type a name for the remote machine.
2. Type a remote preparation server name and user identifier for logging in to that remote server.

Follow this procedure for each remote machine to which you prepare part of the application. If you need to remove any machine from the list, select the name in the list and click the **Delete** button.

If the machine name list is left blank or if the machine you are preparing to is not in the list, the system uses *ne20* and no user identifier. If the machine name is specified, the system uses the remote preparation server name and the user identifier corresponding with that machine.

Specify the remote machine's platform in the Platform list box. Depending on the platform you select, the list of parameters changes. See Parameters for each machine platform for the parameters for each platform.

The Windows platform provides the option for you to specify a database user ID and password. The user ID and password are no longer hard-coded in the HPS.INI file. Specify a user ID and password here. If you leave these fields blank, the system uses the system default user ID and password.

Parameters for each machine platform

Platform	Parameters
Windows	• Preparation server • Database User ID • Database Password

Unix	• Preparation server • User ID
Mainframe	• Preparation server • User ID • Password • Repository • Version • Code page

The Client-Side Codegen settings apply to UNIX and Mainframe computers only See Client Side Codegen settings for more details.

Client Side Codegen settings

Setting	Description
Client-side code generation	If selected, code is generated on the PC but is not submitted to the host for compilation. This setting is dependent on OpenCOBOL Client-side Codegen being selected.
Submit sets individually	If selected, sets are submitted on Super Prepare. If not selected, sets are not submitted because they are automatically pulled in with the Rule prepare. If you select Submit sets individually, you can only Super Prepare or Rebuild.
Client-side macro expansion	If selected, the macro preprocessor is run on the client. Use this setting if the host macro preprocessor is enabled, as there is no need to use the expanded source from the PC.

After you make changes, do one of the following:

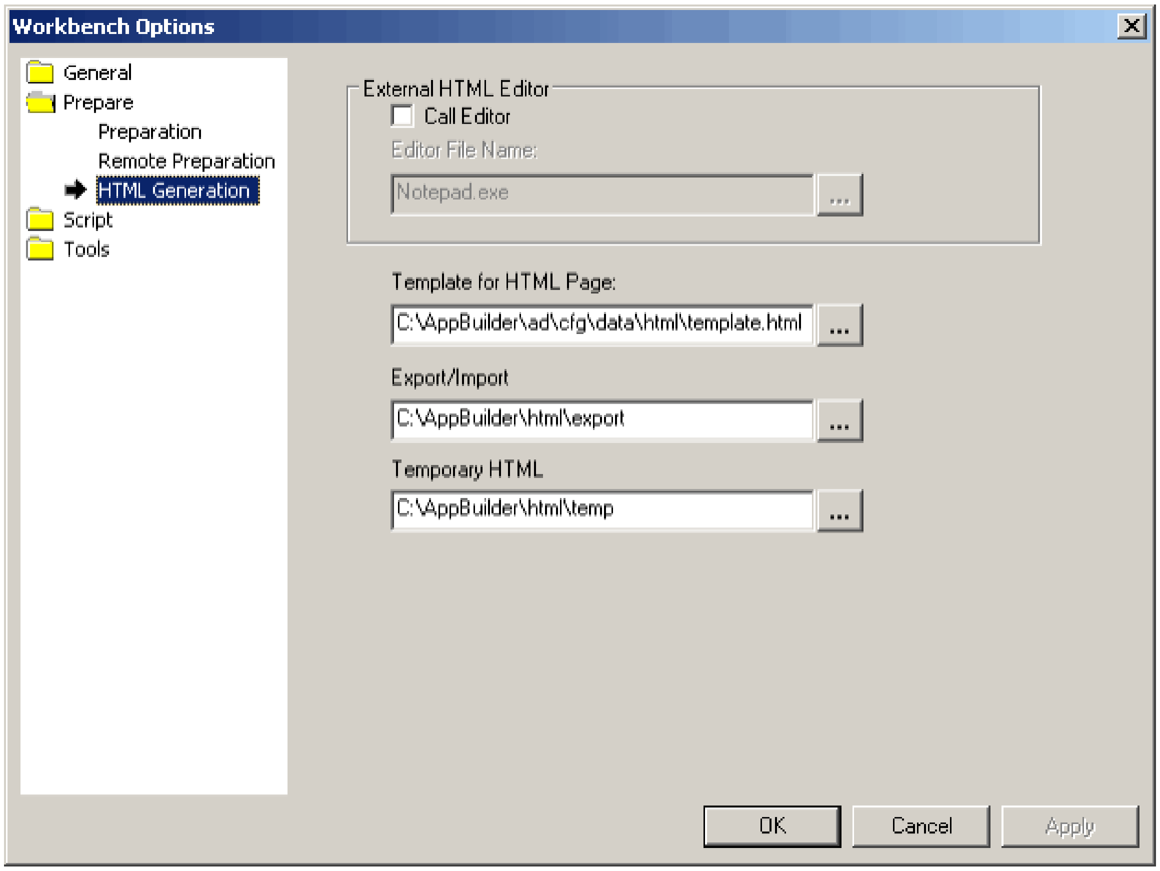
- Click **Apply** to apply the changes for all the options and keep the Options window open.
- Click **OK** to apply the changes for all the options and close the Options window.
- Click **Cancel** to close the Options window without applying any changes.

You can control the case sensitivity of the macro preprocessor using the CODEGEN and CGMEX parameters. If you enable this option, the preprocessor ignores the case of the macro value. For more details about case sensitivity, refer to *Rules Language Reference Guide*.

HTML Generation Options

These options relate to HTML generation. To get to the HTML Generation section of the Workbench Options window, select **Tools** > **Workbench Options**, then click **Prepare** > **HTML Generation**. The HTML Generation set of options is displayed as shown in HTML Generation options. Read more about generating HTML for a window in HTML Generation.

HTML Generation options


Workbench Options

- General
- Prepare
  - Preparation
  - Remote Preparation
  - HTML Generation
- Script
- Tools

External HTML Editor

Call Editor

Editor File Name: Notepad.exe

Template for HTML Page: C:\AppBuilder\ad\cfg\data\html\template.html

Export/Import: C:\AppBuilder\html\export

Temporary HTML: C:\AppBuilder\html\temp

HTML Generation options

Option	Description
External HTML Editor	Options that refer to using an HTML editor outside of AppBuilder. See Matrix Builder Options for more information.
Call Editor	Select this option to have edit authority in your HTML files. If this option is not selected, you cannot edit the HTML. You can only view it. Notepad is the default editor.
Editor File Name	Notepad is the default editor, but you can select another HTML editor here. Use the browse button to locate another HTML editor file.
Template for HTML page	Use the browse button to locate the HTML template file. See Template for HTML Page for more information.
Export / Import	Use the browse button to specify the location and directory path for imports and exports of HTML files. See Export/Import Location for more information.
Temporary HTML	Use the browse button to specify the location and directory path for temporary HTML files when generating HTML. See Temporary HTML Location for more information.

After you make changes, do one of the following:

- Click **Apply** to apply the changes for all the options and keep the Options window open.
- Click **OK** to apply the changes for all the options and close the Options window.
- Click **Cancel** to close the Options window without applying any changes.

Script Section

These options are associated with TurboScripter and TurboCycler. To get to the Script section of the Workbench Options window, select **Tools** > **Workbench Options**, then click on **Script** in Workbench Options window. The Script section contains two sets of options:

- TurboScripter Options
- TurboCycler Options.

TurboScripter Options

The TurboScripter set of options is displayed by default when you click on **Prepare** in Workbench Options window. These options specify the location of the folder for scripts and the level of tracing, if any, to be run when the script is executed. The TurboCycler set of options is displayed as shown in TurboScripter options.

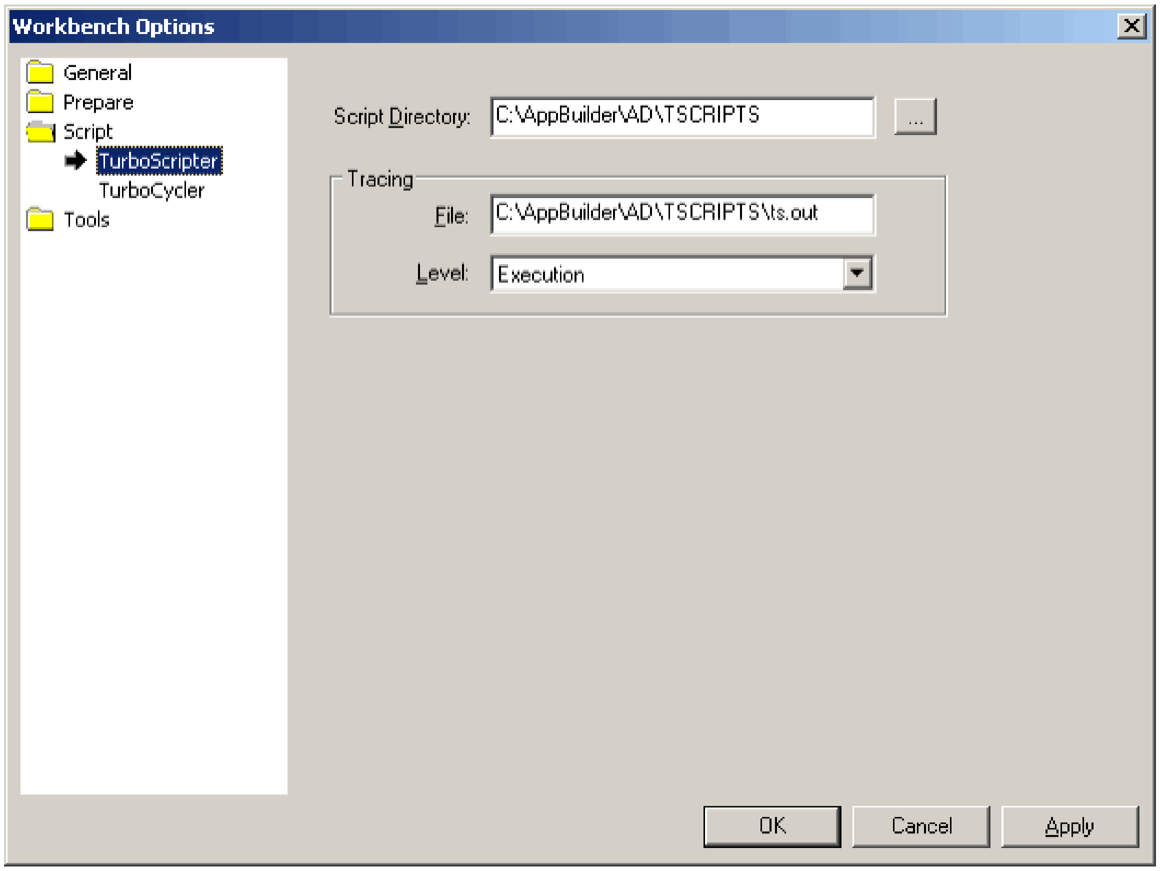
TurboScripter tab options

Option	Description
Script Directory	The location of the directory containing the TurboScripter scripts.
Tracing	
File	The name of the file to which messages are written.
Level	Tracing levels provided are: • None • Execution: traces errors only • Debug: traces warnings, errors, and function • All: traces what the program is doing

After you make changes, do one of the following:

- Click **Apply** to apply the changes for all the options and keep the Options window open.
- Click **OK** to apply the changes for all the options and close the Options window.
- Click **Cancel** to close the Options window without applying any changes.

TurboScripter options



Script Directory: C:\AppBuilder\AD\TSCRIPTS

Tracing

File: C:\AppBuilder\AD\TSCRIPTS\ts.out

Level: Execution

TurboCycler Options

These are options for the TurboCycler scripting tool. To get to the TurboCycler set of options of the Workbench Options window, select **Tools** > **Workbench Options**, then click **Prepare** > **TurboCycler** in Workbench Options window. The TurboCycler set of options is displayed as shown in

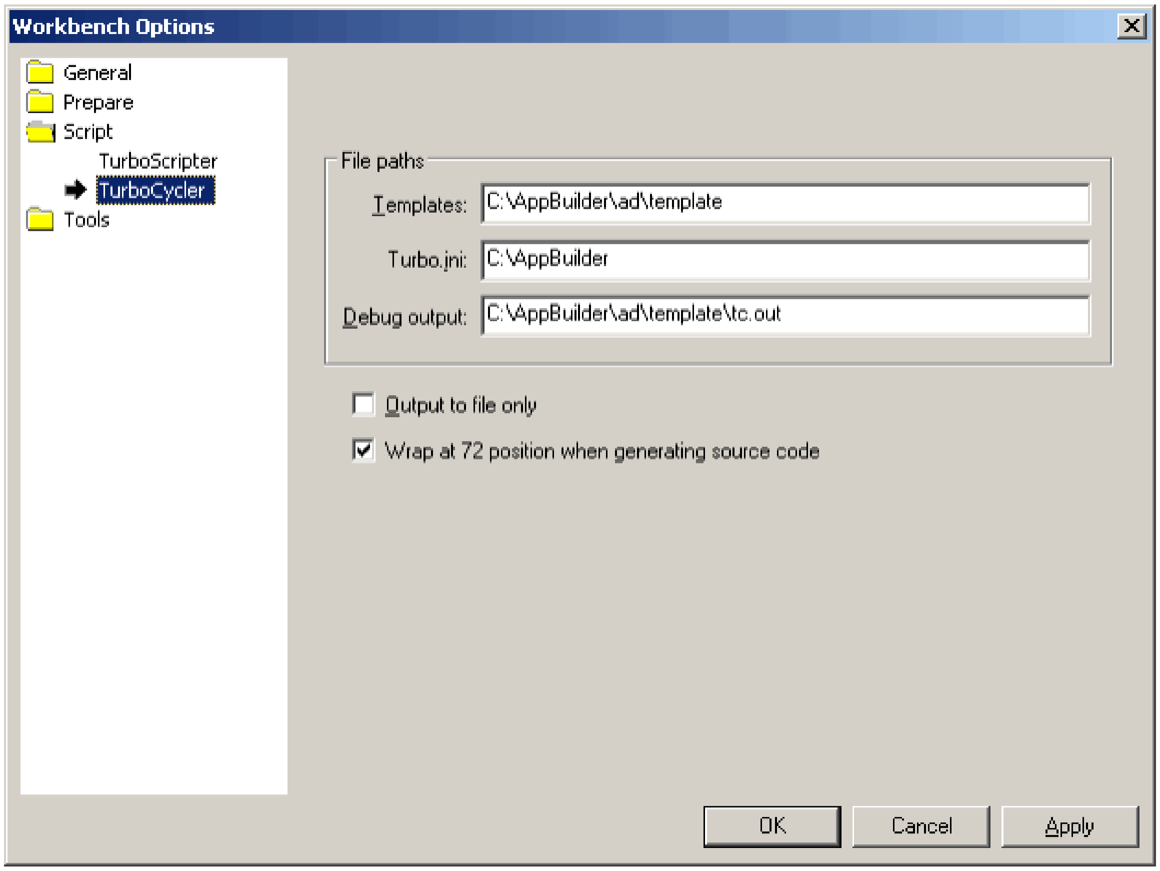
TurboCycler options.

Read more about TurboCycler in the *Scripting Tools Reference Guide*.

TurboCycler options summary

Option	Description
Templates	Specify the location (full directory path) of the TurboCycler templates.
Turbo.INI	Specify the location (full directory path) of the TURBO.INI file.
Debug output	Specify the location (full directory path) and the file name for the debug file.
Output to file only	When this option is checked, output is only written to a file, preventing the output from being displayed on the analysis window in the Workbench.
Wrap at 72 position when generating source code	When this option is checked, the code wraps at line position 72 when generating source code.

TurboCycler options



Templates: C:\AppBuilder\ad\template

Turbo.ini: C:\AppBuilder

Debug output: C:\AppBuilder\ad\template\tc.out

☐ Output to file only

☑ Wrap at 72 position when generating source code

After you make changes:

- Click **Apply** to apply the changes for all the options and keep the Options window open.
- Click **OK** to apply the changes for all the options and close the Options window.
- Click **Cancel** to close the Options window without applying any changes.

Tools Section

These options are associated with different drawing tools used in AppBuilder. To get to the Tools section of the Workbench Options window, select **Tools** > **Workbench Options**, then click on **Tools** in Workbench Options window. The Tools section contains the following sets of options:

- Text Editor Options
- Drawing Tools Options

- Engineering Options
- Matrix Builder Options
- Report Painter Options
- Code Assistant Options
- Set Builder Options
- Window Painter Options

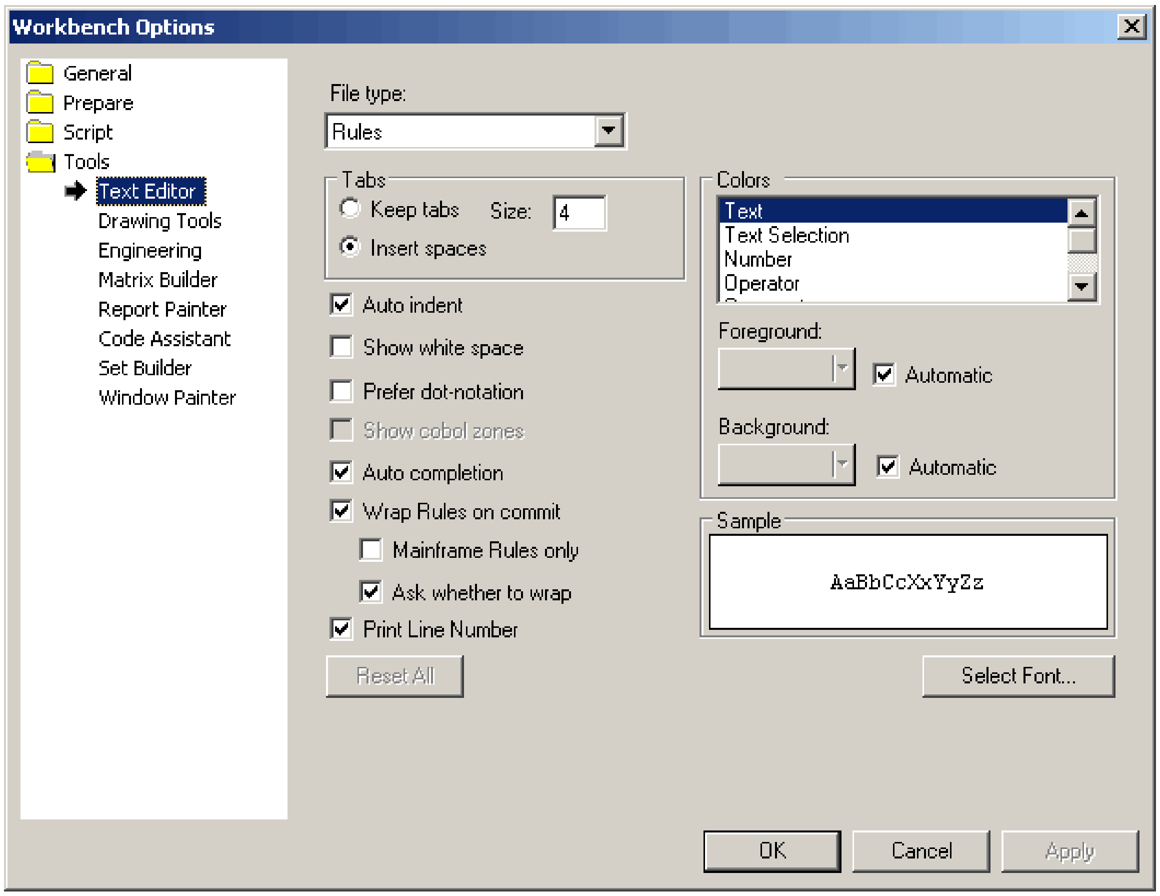
Text Editor Options

These options control the text editor called from the Construction Workbench in the work area. To get to the Text Editor set of options of the Workbench Options window, select **Tools** > **Workbench Options**, then click **Tools** in the Workbench Options window. Click **Text Editor** if this set of options is not displayed by default (see Text Editor options).

Text Editor options summary

Option	Description
File type	Select the file type before setting any of the other options. Choose from among a list that includes: • Rules • OORules • C • VBScript • Java • COBOL • PL/1 • Assembler
Tabs	Keep tabs or not; insert spaces or not.
Insert spaces	Converts tab to the number of spaces specified in Size.
Keep tabs	Saves tab as a tab character.
Size	Select the number of spaces that a tab becomes. (The default is four spaces.)
Auto indent	Specifies if the Rule Painter maintains the indentation for each line.
Show white space	Specifies if the white space is shown.
Prefer dot-notation	Specifies if objects appear in dot notation when inserted or drag-and-dropped.
Show cobol zones	Specifies if the zones (columns) in COBOL are shown.
Auto Completion	Specifies if a dialog containing the list of possible values is populated for a dot notation in the SET statement in the Rule Painter.
Wrap Rules on commit	Specifies whether AppBuilder automatically adds a line return if there are more than 72 characters on a line. You can specify this for mainframe Rules only or all Rules. You can specify whether to ask before wrapping.
Print Line Number	Check this option to print the line number when printing Rule source.
Colors	Specifies the foreground or background color (or both) for various elements from a list.
Sample	Offers a preview of the selected font and colors.
Select Font	Selects a font. You can view the results in the **Sample** area.

Text Editor options


Workbench Options

- General
- Prepare
- Script
- Tools
  - Text Editor
  - Drawing Tools
  - Engineering
  - Matrix Builder
  - Report Painter
  - Code Assistant
  - Set Builder
  - Window Painter

File type: Rules

Tabs
- ○ Keep tabs Size: 4
- ● Insert spaces

- ☑ Auto indent
- ☐ Show white space
- ☐ Prefer dot-notation
- ☐ Show cobol zones
- ☑ Auto completion
- ☑ Wrap Rules on commit
  - ☐ Mainframe Rules only
  - ☑ Ask whether to wrap
- ☑ Print Line Number

Colors
- Text
- Text Selection
- Number
- Operator

Foreground: ☑ Automatic

Background: ☑ Automatic

Sample: AaBbCcXxYyZz

After you make changes, do one of the following:

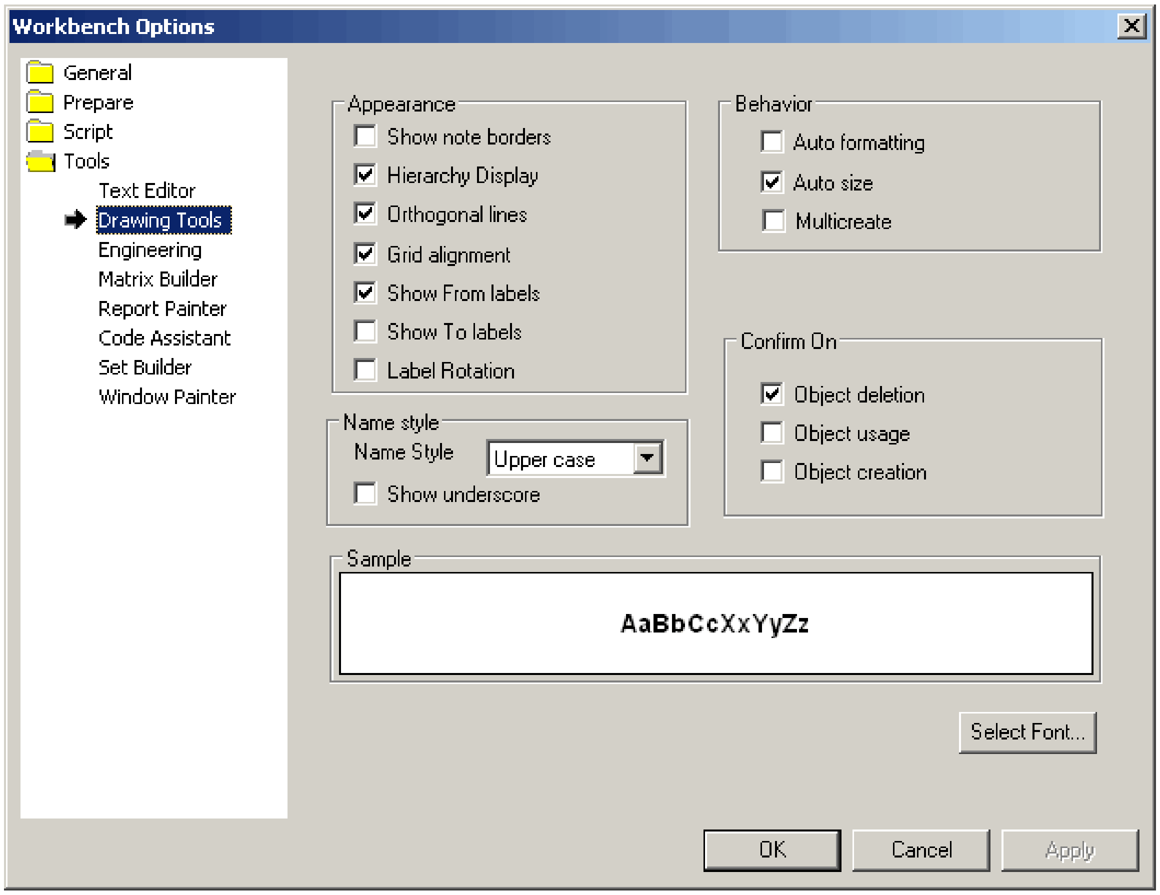
- Click **Apply** to apply the changes for all the options and keep the Options window open.
- Click **OK** to apply the changes for all the options and close the Options window.
- Click **Cancel** to close the Options window without applying any changes.

Drawing Tools Options

These options specify default choices for the design tools that create and modify diagrams (drawings). To get to the Drawing Tools section of the Workbench Options window, select **Tools** > **Workbench Options**, then click **Tools** > **Drawing Tools**. The Drawing Tools set of options is displayed as shown in Drawing Tools options.

Read more about the drawing tools in Modeling Tools.

Drawing Tools options


Workbench Options

- General
- Prepare
- Script
- Tools
  - Text Editor
  - Drawing Tools
  - Engineering
  - Matrix Builder
  - Report Painter
  - Code Assistant
  - Set Builder
  - Window Painter

Appearance
- [ ] Show note borders
- [✓] Hierarchy Display
- [✓] Orthogonal lines
- [✓] Grid alignment
- [✓] Show From labels
- [ ] Show To labels
- [ ] Label Rotation

Behavior
- [ ] Auto formatting
- [✓] Auto size
- [ ] Multicreate

Name style
- Name Style: Upper case
- [ ] Show underscore

Confirm On
- [✓] Object deletion
- [ ] Object usage
- [ ] Object creation

Sample: AaBbCcXxYyZz

Drawing Tools options summary

Option	Description
Show note borders	Select this option to specify that the notes are shown with a thin border.
Hierarchy Display	Select this option to display the child items of an object.
Orthogonal lines	Select this option to specify that the system draws lines in individual segments, rather than in an automatic single line.
Grid alignment	Select this option to specify that the system automatically aligns objects to the grid.
Show from labels	Select this option to display the labels of the "from" end of the relations.
Show to labels	Select this option to display the labels of the "to" end of the relations.
Label Rotation	Select this option to enable the label rotation.
Auto formatting	Select this option to specify that the system creates lines using as few segments as possible, regardless of how many segments you draw with the mouse.
Auto size	Select this option to specify that the system automatically resizes objects to display all details.
Multicreate	Select this option to enable the creation of multiple objects. If the option is not selected, a single click will create an object and the status will return to selection.
Confirm on	Select this option to specify that the system prompts for confirmation when you use **Object deletion**, **Object usage**, or **Object creation** for an object. This is the global setting for drawing tools. Each existing drawing tool will have its own options which override the global settings.

Name style	Choose from Upper case, Lower Case, and Mixed Case. Click Show Underscore to display underlined names.
Sample and Select Font	The Sample window shows what a selected font looks like. Click the **Select Font** button to specify the font the system uses in diagrams.

After you make changes, do one of the following:

- Click **Apply** to apply the changes for all the options and keep the Options window open.
- Click **OK** to apply the changes for all the options and close the Options window.
- Click **Cancel** to close the Options window without applying any changes.

Engineering Options

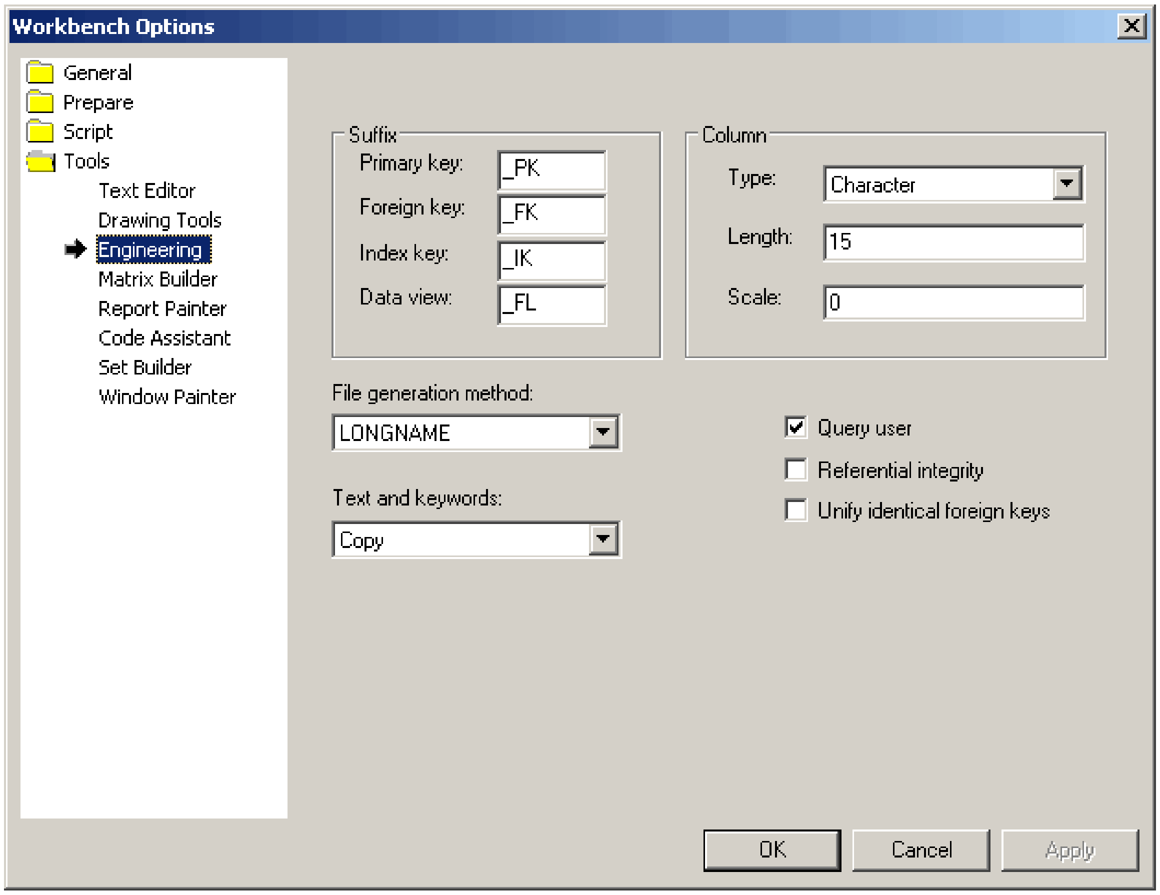
These options specify default choices for the engineering tools. To get to the Engineering set of options of the Workbench Options window, select **Tools** > **Workbench Options**, then click **Tools** > **Engineering**. The Engineering set of options is displayed as shown in Engineering options. Read more about engineering tools in Modeling Tools.

By default, the transformation process uses the implementation name for the entities it converts. Use the **File Generation Method** field to specify if the generation method uses long names instead of implementation names.

If you specify that transformation uses long names instead of implementation names, you can not perform view-to-view mapping.

Develop a database schema that promotes view-to-view mapping by renaming each object when you create it. To create custom names for generated objects, select the **Query user** option on the **Engineering** set of options.

Engineering options



Suffix
Primary key: _PK
Foreign key: _FK
Index key: _IK
Data view: _FL

Column
Type: Character
Length: 15
Scale: 0

File generation method: LONGNAME

Text and keywords: Copy

[✓] Query user
[ ] Referential integrity
[ ] Unify identical foreign keys

Engineering options summary

Option	Description
Suffix	Determines the suffix that you want to assign to certain objects that forward engineering creates. Type a value for each of these types of suffix: • Primary_Key_Suffix • Foreign_Key_Suffix • Index_Key_Suffix • Data_View_Suffix
Column	Specify default attributes for data types. You can modify the default value here or modify values for a specific data type in the data type Object Property window. See Creating Attribute Hierarchies in the Entity Relationship Diagram.
File generation method	Select either the Longname or the Implementation name. The Longname is the name used in the database diagram. The implementation name is a name you provide according to naming conventions for your project.
Query user	The forward engineering process gives new objects system-created names based on the name of the original entity. To create custom names for generated objects, select this check box.
Referential Integrity	If this box is checked, the system verifies the referential integrity of the database.
Unify identical foreign keys	If an entity has inherited two or more foreign keys from multiple super type entities, they are migrated into a single foreign key.
Text and keywords	Select the action to do with duplicate text and keywords: **Copy**, **Overwrite**, and **Don't Copy**.

The Primary_Key_Suffix, Foreign_Key_Suffix, Index_Key_Suffix, and Data_View_Suffix determine what prefix or suffix you want to assign certain objects that forward engineering creates. You can also change the Default_Column_Type, Default_Column_Length, and Default_Column_Scale settings.

Set the various suffixes in the **Suffix** area.

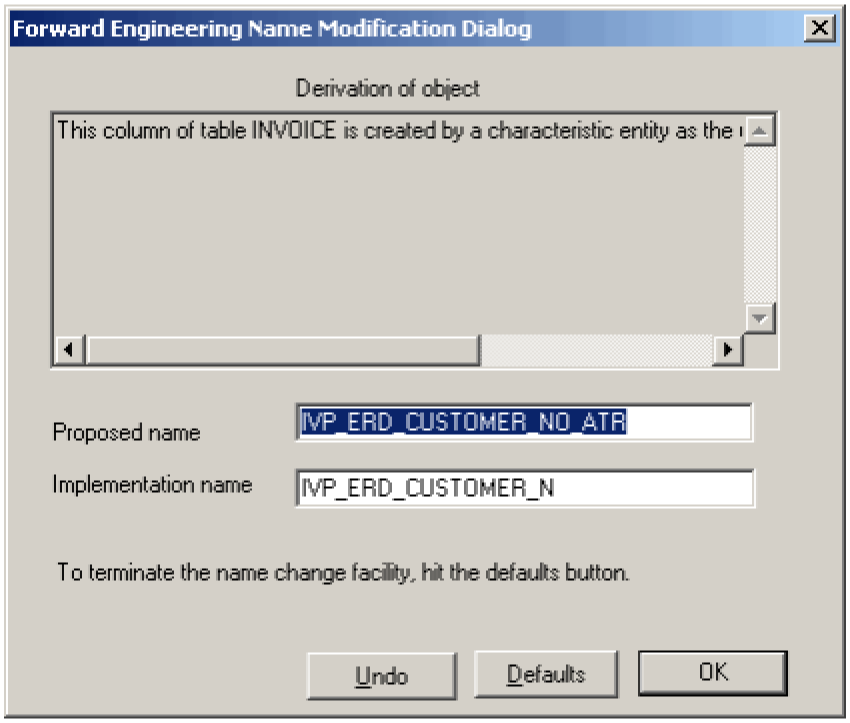
Set the column options in the **Column** area.

Select the file generation method from the **File generation method** list, either the longname or the implementation name.

Select the action to do with duplicate text and keywords, either copy, overwrite, or do not copy, from the **Text and keywords** list.

The **Query user** option causes the system to display the Forward Engineering Name Modification window (as in Changing generated names) when forward engineering creates each new object. You can specify a prefix or suffix to be used with the object name when forward engineering creates new objects, as well as a default column type for any attribute that does not have a data type.

Changing generated names



This column of table INVOICE is created by a characteristic entity as the

Proposed name IVP_ERD_CUSTOMER_NO_ATR

Implementation name IVP_ERD_CUSTOMER_N

To terminate the name change facility, hit the defaults button.

To stop displaying the window that is displayed for renaming the objects and resume automatic object naming, click **Defaults**. Selecting the **Defaults** button is equivalent to deselecting the **Query user** option on the Engineering section of the Workbench Options window.

After you make changes:

- Click **Apply** to apply the changes for all the options and keep the Options window open.
- Click **OK** to apply the changes for all the options and close the Options window.
- Click **Cancel** to close the Options window without applying any changes.

Matrix Builder Options

These are the options for the Matrix Builder tool. Use the Matrix Builder to construct a matrix that shows the association between objects. This set of options defines the look and feel of your matrix. To get to the Matrix Builder set of options of the Workbench Options window, select **Tools** > **Workbench Options**, then click **Tools** > **Matrix Builder**. The Matrix Builder options are displayed as shown in Matrix Builder options.

Read more about Matrix Builder in Construction Tools.

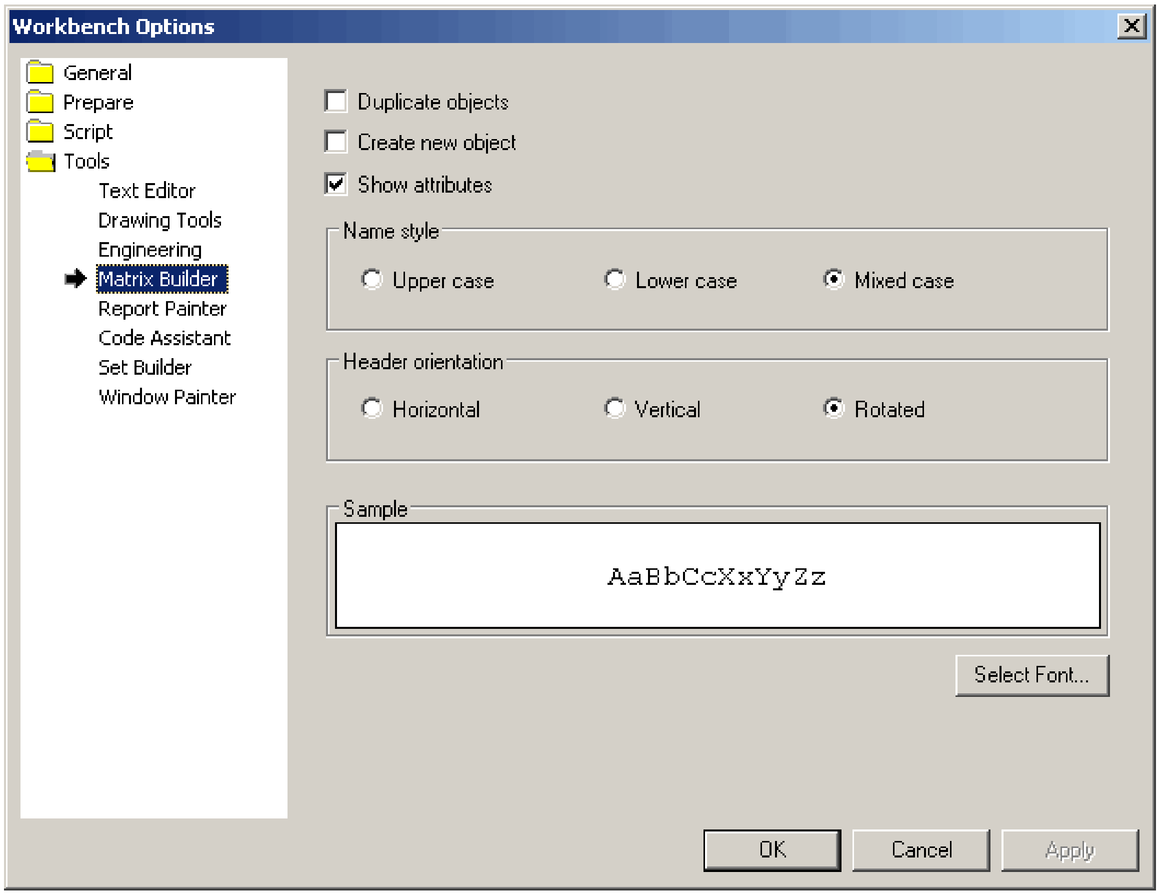
Matrix Builder options summary

Option	Description
Duplicate objects	Specify if the system allows duplicate objects in rows and columns of a matrix. This option indicates whether objects can be duplicated.
Create new object	Specify if the system can create a new object by typing its name in the header. If not selected, the system displays an error message when you attempt to insert an object that is not in the repository. This option indicates whether a new object can be created.
Show attributes	Specify if the system displays the relationship properties (or attributes) in the matrix table cells. For example, the letters C, R, U, or D for the default template. If unselected, the system displays a big X across the table cell and does not display the attributes.
Name style	Specify the case of the text shown in matrix row and column label names. Upper case means all the letters are in upper case. Lower case means all the letters are in lower case. Mixed case means that letters can be either upper or lower case.
Header orientation	Specify orientation or direction of the header table cells. They may be displayed horizontal, vertical, or at an angle (rotated). The default is rotated.
Sample and Select Font	Specify text font used in the matrix labels. Select the **Change Font** button to change the properties of your default font. View a sample of the selected text in the Default font area.

After you make changes, do one of the following:

- Click **Apply** to apply the changes for all the options and keep the Options window open.
- Click **OK** to apply the changes for all the options and close the Options window.
- Click **Cancel** to close the Options window without applying any changes.

Matrix Builder options



☐ Duplicate objects
☐ Create new object
☑ Show attributes

Name style: Upper case / Lower case / [✓ Mixed case]

Header orientation: Horizontal / Vertical / [✓ Rotated]

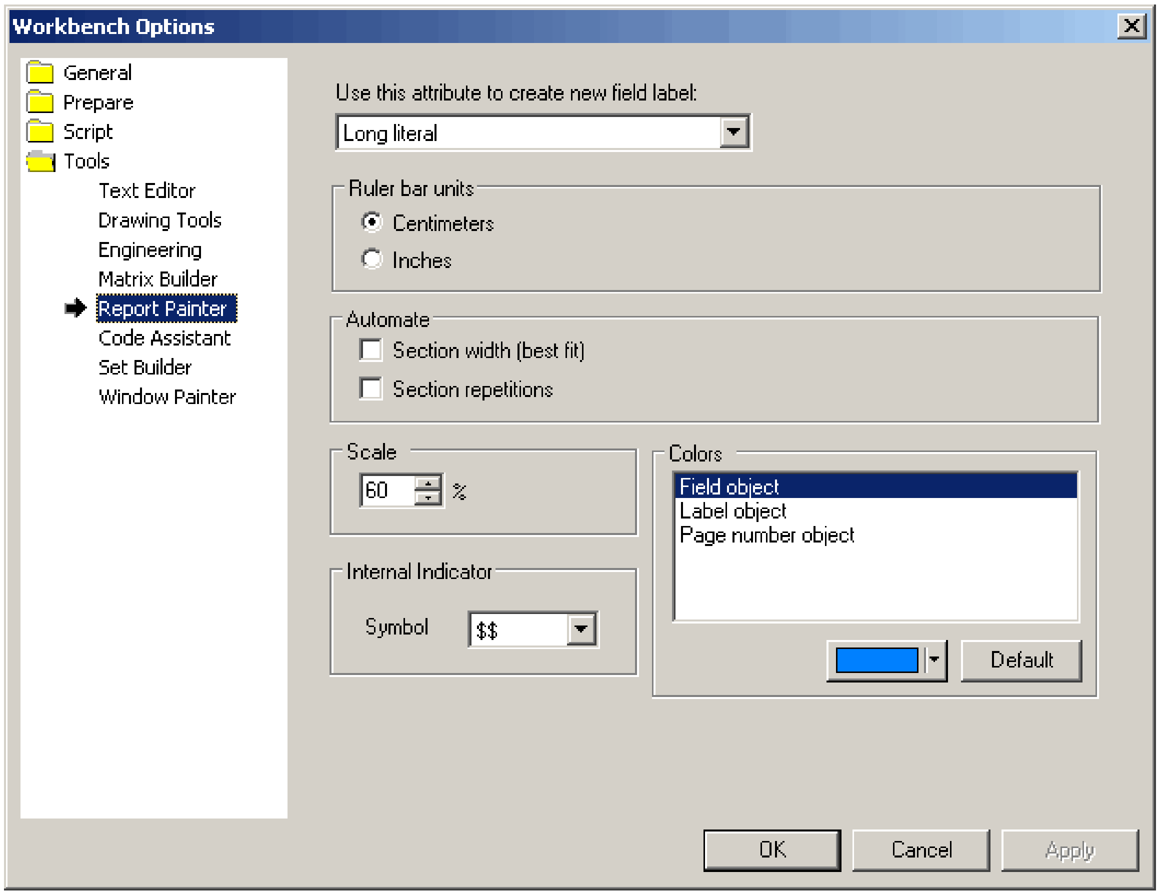
Sample: AaBbCcXxYyZz

Report Painter Options

Use the **Report Painter** set of options to specify options that apply to the use of Report Painter. To get to the Report Painter set of options of the Workbench Options window, select **Tools** > **Workbench Options**, then click **Tools** > **Report Painter**. The Report Painter options are displayed (see Report Painter options).

Read more about Report Painter in the *Reports Guide*.

Report Painter options



Use this attribute to create new field label: Long literal

Ruler bar units: (● Centimeters) ○ Inches

Automate: ☐ Section width (best fit) ☐ Section repetitions

Scale: 60 %

Internal Indicator: Symbol $$

Colors: Field object, Label object, Page number object

Report Painter options summary

Option	Description
Use this attribute to create new field label:	Section type of labels: • Long Literal • Short Literal • Name • Short Name • None
Ruler Bar Units	Set the ruler bar units to the following: • Centimeters • Inches
Automate section width (best fit)	Specify whether to set the section width automatically.
Automate section repetitions	Specify whether to set the section repetition automatically.
Scale	Set the scale of the display of the report in the Report Painter window. Valid choices are integer values from 20% to 250%.
Internal Indicator	Use this section to select the currency symbol. To change the symbol, select the symbol that you want to use from the Symbol combo box.

Colors	Select the objects to which the color choice applies: • Field object • Label object • Page number object Then select the color by clicking the arrow and selecting a color from the **Color** drop-down as shown in Report Painter options. Click **Default** to select the default color.

After you make changes:

- Click **Apply** to apply the changes for all the options and keep the Options window open.
- Click **OK** to apply the changes for all the options and close the Options window.
- Click **Cancel** to close the Options window without applying any changes.

Code Assistant Options

These options control how event procedures are added to the text editor called from the Construction Workbench in the work area. Using the options in Code Assistant, you can automate the Event Wizard to create event procedures in the Java block, the HTML block, the Server block, or without block. You can also have the Event Wizard prompt you for where to create the event procedure. To get to the Code Assistant set of options of the Workbench Options window, select **Tools** > **Workbench Options**, then click **Tools** > **Code Assistant**. The Code Assistant options are displayed as shown in Code Assistant options.

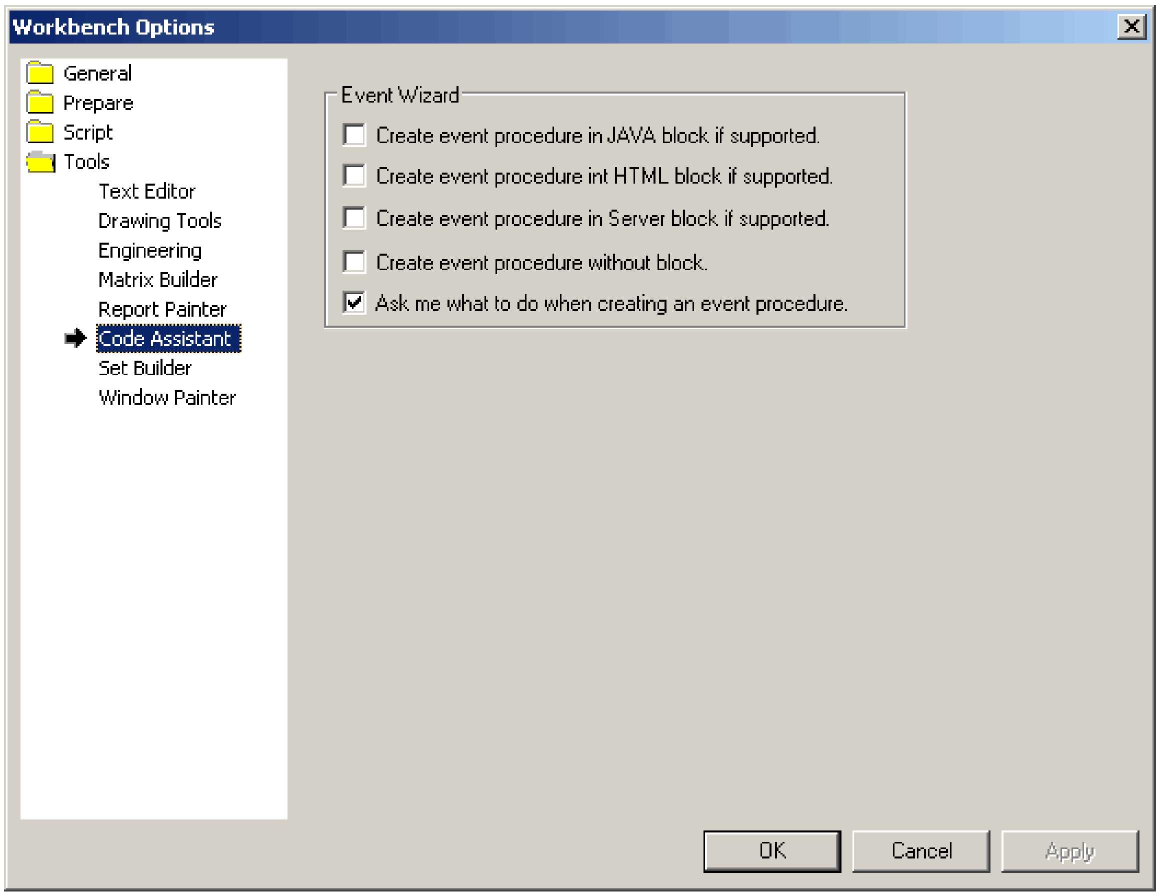
Code Assistant options summary

Option	Description
Create event procedure in JAVA block if supported.	Creates an event procedure in the JAVA block.
Create event procedure in HTML block if supported.	Creates an event procedure in the HTML block.
Create event procedure in Server block if supported.	Creates an event procedure in the Server block.
Create event procedure without block.	Adds the event procedure to the text editor without adding it to a block.
Ask me what to do when create event procedure.	Prompts you to define the behavior of an event procedure.

After you make changes:

- Click **Apply** to apply the changes for all the options and keep the Options window open.
- Click **OK** to apply the changes for all the options and close the Options window.
- Click **Cancel** to close the Options window without applying any changes.

Code Assistant options



Event Wizard

- [ ] Create event procedure in JAVA block if supported.
- [ ] Create event procedure int HTML block if supported.
- [ ] Create event procedure in Server block if supported.
- [ ] Create event procedure without block.
- [✓] Ask me what to do when creating an event procedure.

Set Builder Options

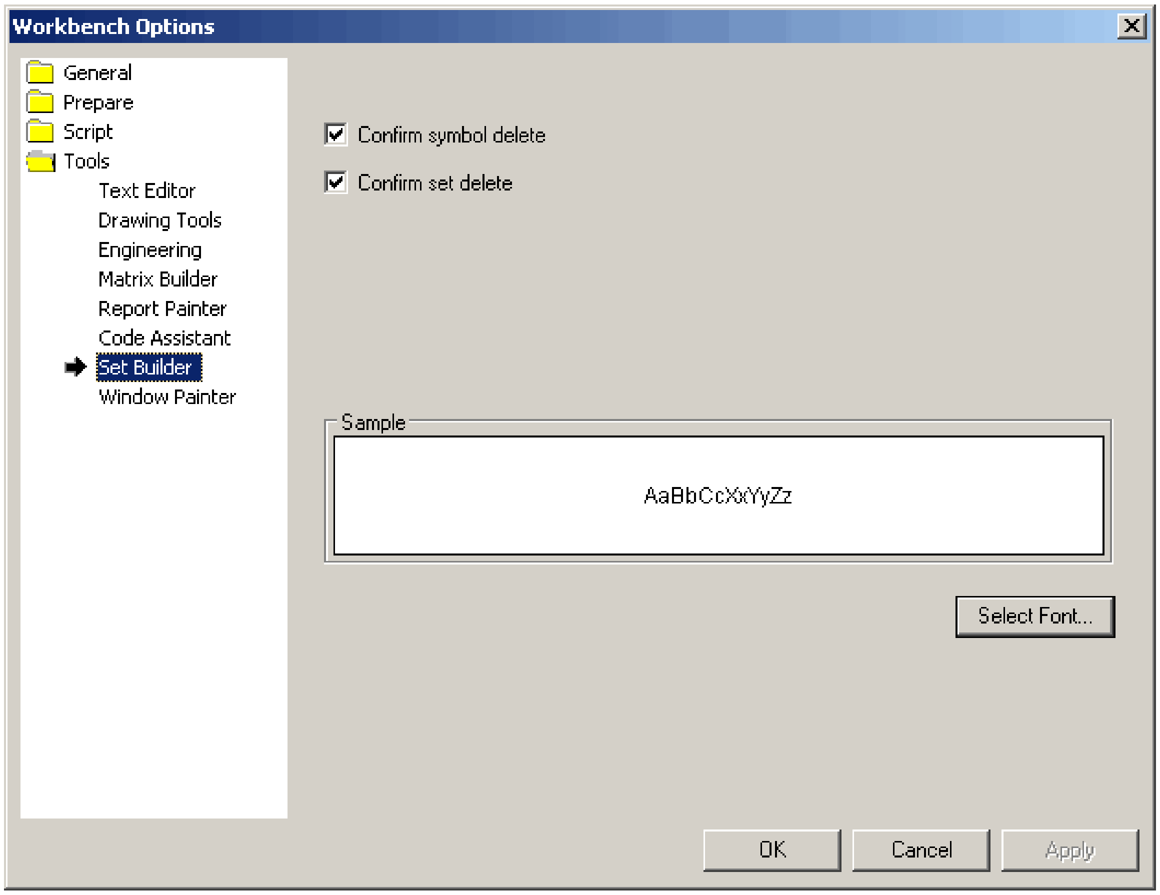
These are the options for the Set Builder tool. To get to the Set Builder set of options of the Workbench Options window, select **Tools** > **Workbench Options**, then click **Tools** > **Set Builder**. The Set Builder options are displayed as shown in Set Builder options.

Read more about Set Builder in Construction Tools.

Set Builder options summary

Option	Description
Confirm symbol delete	Specifies if AppBuilder prompts for confirmation when you **Delete** set symbols.
Confirm set delete	Specifies if AppBuilder prompts for confirmation when you **Delete** sets.
Sample and Select font	Specifies the font for Set Builder. To view or modify the font used in Set Builder, click Select **Font**. In the Font window, select the font, style, size, and script. View a sample of the font in the Sample area. Click **OK** when done.

Set Builder options



☑ Confirm symbol delete

☑ Confirm set delete

After you make changes, do one of the following:

- Click **Apply** to apply the changes for all the options and keep the Options window open.
- Click **OK** to apply the changes for all the options and close the Options window.
- Click **Cancel** to close the Options window without applying any changes.

Window Painter Options

These are options for the Window Painter tool. To get to the Window Painter set of options of the Workbench Options window, select **Tools** > **Workbench Options**, then click **Tools** > **Window Painter**. The Window Painter options are displayed as shown in Window Painter options.

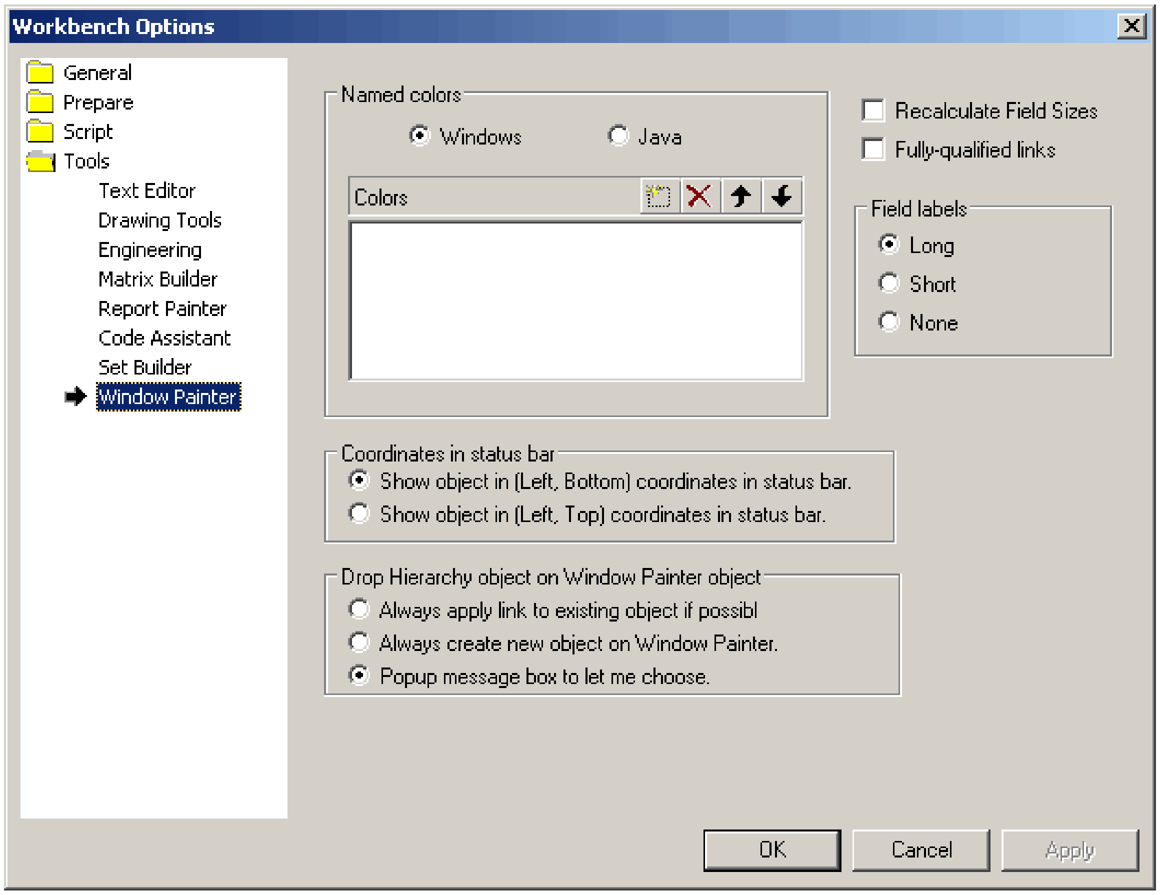
Read more about Window Painter in Window Painter.

Window Painter Options Summary

Option	Description
Named colors	Select Windows or Java
Colors	Define named colors for specific environments here. The defined named colors are specific to the environment in which they are defined. Defining a specific named color in Windows means it is only defined in a Windows environment, and not in a Java environment. This allows the colors with the same name to be used with different color definitions in different environments. You can add, delete, move up, or move down the items in the list using the buttons on the top of that list. To define a new color: 1. Click the **New (insert)** button. 2. Click the ... button. The Color window is displayed. 3. Select the color properties and click **OK**.

Recalculate field sizes	This option is unchecked by default. Select this option to have the system automatically recalculate field size when the object size is changed while Window Painter is closed. Resizing is also dependent on the Form Fit property, which determines whether the object size is based on the length of the data in the field. If the window is created or updated using the AppBuilder Window Painter, this setting should not be used. **Recalculate field sizes** tells the Window Painter to recalculate the width and height of edit fields if the field size has changed since the window was last opened. In other words, if the Window Painter is closed after you modified a field size, when you open Window Painter, it resizes the field based on the length of the data. The **Recalculate field sizes** option is associated with the **Resize** and **Form Fit** field properties discussed in Resizing an Edit Field in Window Painter. If **Recalculate field sizes** is unchecked, the width and height of the edit field in the panel file is retained in the window regardless of changes made outside of Window Painter. Window Painter shows the object with the same field size and does not recalculate it regardless of the Resize or Form Fit property values.
Fully-qualified links	Select this option to specify if links are fully qualified.
Field Labels	Select this option to specify the default label (**Long**, **Short**, or **None**) used for fields.
Show objects in (Left, Bottom) coordinates in status bar	Select this option to display left and bottom coordinates on the status bar from the workbench options.
Show objects in (Left, Top) coordinates in status bar	Select this option to display left and top coordinates on the status bar from the workbench options.
Always apply link to exist object if possible	Select this option to automatically apply a link to an existing Window Painter object.
Always create new object on Window Painter	Select this option to create a new object to apply a link to.
Popup message box to let me choose.	Select this option to choose whether to apply a link to an existing object or to create a new object in Window Painter. This is the default selection in the Window Painter section.

Window Painter options


Workbench Options

General
Prepare
Script
Tools
Text Editor
Drawing Tools
Engineering
Matrix Builder
Report Painter
Code Assistant
Set Builder
Window Painter

Named colors
[✓ Windows] Java

Colors

Recalculate Field Sizes
Fully-qualified links

Field labels
[✓ Long]
Short
None

Coordinates in status bar
[✓ Show object in (Left, Bottom) coordinates in status bar.]
Show object in (Left, Top) coordinates in status bar.

Drop Hierarchy object on Window Painter object
Always apply link to existing object if possibl
Always create new object on Window Painter.
[✓ Popup message box to let me choose.]

OK Cancel Apply

After you make changes:

- Click **Apply** to apply the changes for all the options and keep the Options window open.
- Click **OK** to apply the changes for all the options and close the Options window.
- Click **Cancel** to close the Options window without applying any changes.

Using the Hierarchy window

The hierarchy provides the application structure and flow of your project. The Hierarchy window shows all the objects in the repository and their relationship to one another. Use the Hierarchy window to interact with the applications and objects in the repository.
This chapter covers the following topics:

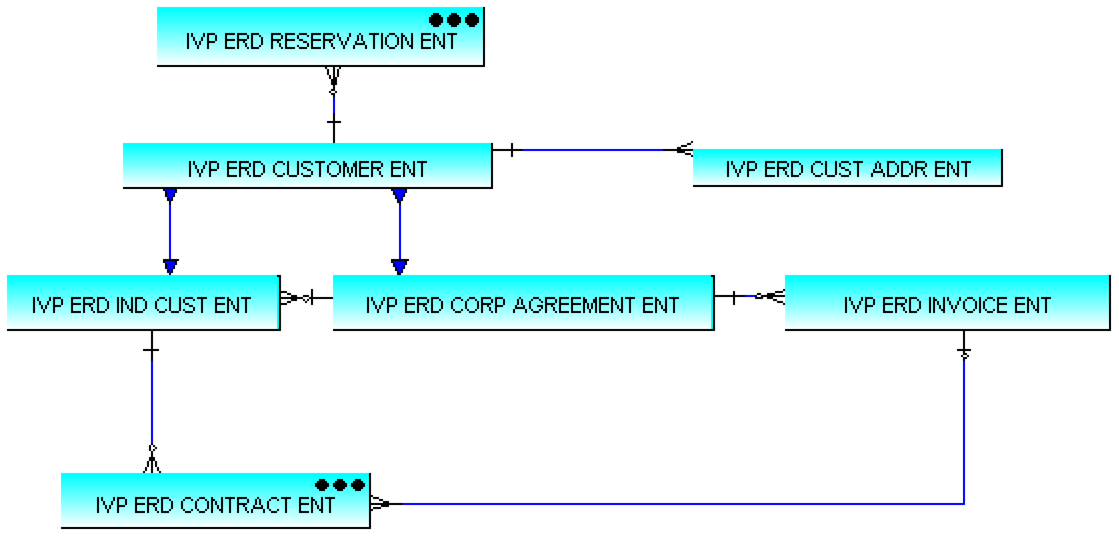
- Understanding Parts of a Hierarchy
- Building a Hierarchy
- Navigating the Hierarchy
- Hierarchy Objects
- Tabs in the Hierarchy Window
- Searching for Objects and Text
- Showing Impact of Changes
- Building Attribute Hierarchies
- Hierarchy Object Properties
- Customizable ToolTips

Building Attribute Hierarchies

To build an attribute hierarchy, complete the following steps:

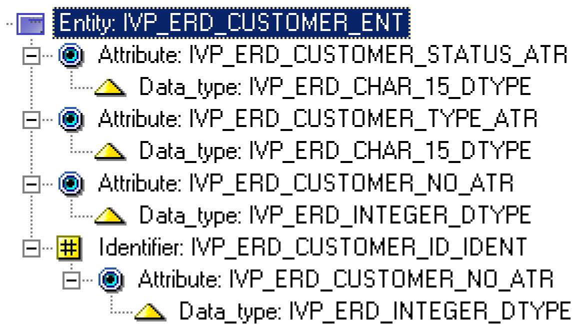
1. Use the Entity Relationship Diagram tool to create the graphical representation as shown in Entity relationship diagram. Refer to Creating or Opening an Entity Relationship Diagram in Modeling Tools for more information about creating entity relationship diagrams.

Entity relationship diagram



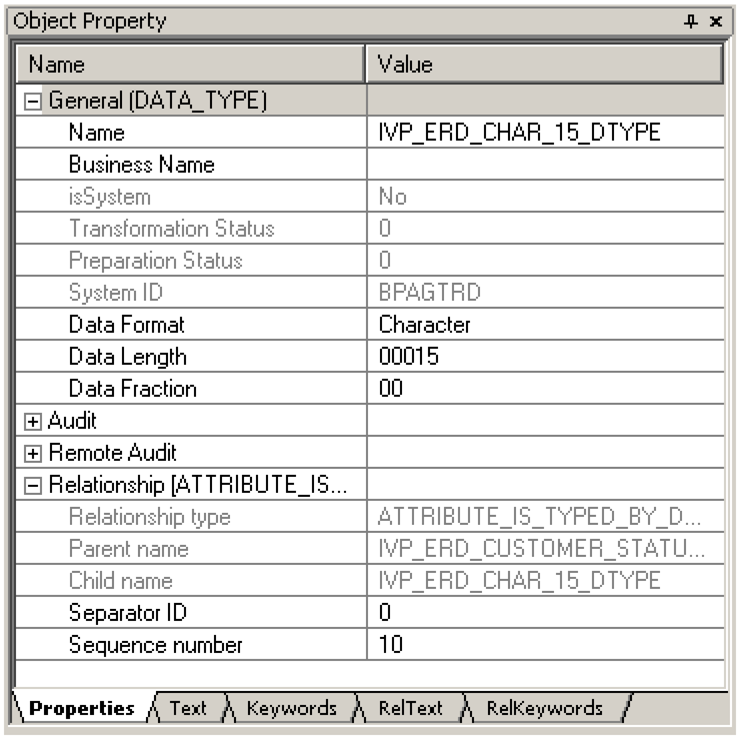
2. From the Construction Workbench menu, select **Edit > Select all** , then right-click and select **Open as Hierarchy** to open a scoped version of the entity's hierarchy.
3. Create the attributes, identifiers, and data types for the **IVP_ERD_CUSTOMER_ENT** entity as in IVP_ERD_CUSTOMER_ENT entity hierarchy. For each Data type, use the Object Property window to set the format and length properties, as in Data type properties dialog with General attributes for the data type CHAR_15.

IVP_ERD_CUSTOMER_ENT entity hierarchy


```
Entity: IVP_ERD_CUSTOMER_ENT
  Attribute: IVP_ERD_CUSTOMER_STATUS_ATR
    Data_type: IVP_ERD_CHAR_15_DTYPE
  Attribute: IVP_ERD_CUSTOMER_TYPE_ATR
    Data_type: IVP_ERD_CHAR_15_DTYPE
  Attribute: IVP_ERD_CUSTOMER_NO_ATR
    Data_type: IVP_ERD_INTEGER_DTYPE
  Identifier: IVP_ERD_CUSTOMER_ID_IDENT
    Attribute: IVP_ERD_CUSTOMER_NO_ATR
      Data_type: IVP_ERD_INTEGER_DTYPE
```

4. Select the correct **Data format** and **Data length** in the **General** section. For example, the data type entity named CHAR_15 should have a data format of character and a data length of 15, as shown in Data type properties dialog with General attributes for the data type CHAR_15.

Data type properties dialog with General attributes for the data type CHAR_15

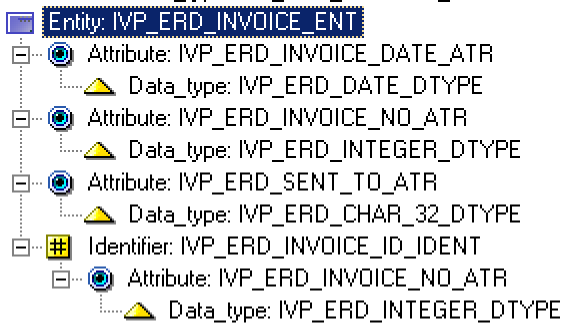


Object Property

Name	Value
⊟ General (DATA_TYPE)	
Name	IVP_ERD_CHAR_15_DTYPE
Business Name	
isSystem	No
Transformation Status	0
Preparation Status	0
System ID	BPAGTRD
Data Format	Character
Data Length	00015
Data Fraction	00
⊞ Audit	
⊞ Remote Audit	
⊟ Relationship (ATTRIBUTE_IS...	
Relationship type	ATTRIBUTE_IS_TYPED_BY_D...
Parent name	IVP_ERD_CUSTOMER_STATU...
Child name	IVP_ERD_CHAR_15_DTYPE
Separator ID	0
Sequence number	10

5. Display the Object Property window for the **IVP_ERD_CUSTOMER_ID_IDENT** identifier in the hierarchy if it is not already displayed.
6. On the *General* section, change the **Type** field to **Primary**.
7. Create the hierarchy shown in IVP_ERD_INVOICE_ENT Hierarchy for the **IVP_ERD_INVOICE_ENT** entity.

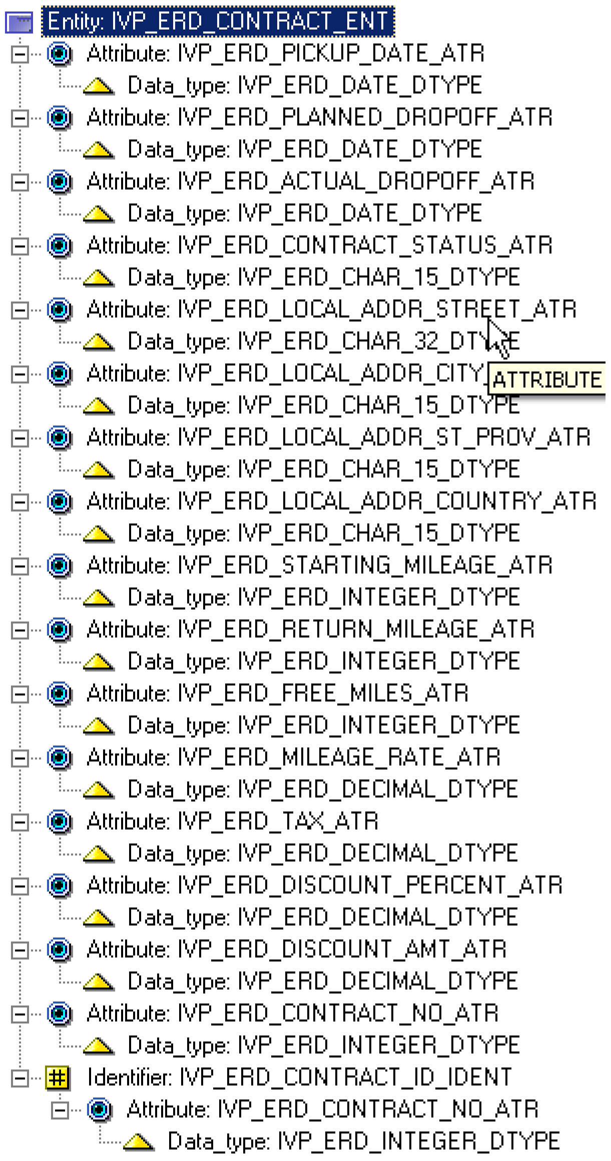
IVP_ERD_INVOICE_ENT Hierarchy



```
Entity: IVP_ERD_INVOICE_ENT
  Attribute: IVP_ERD_INVOICE_DATE_ATR
    Data_type: IVP_ERD_DATE_DTYPE
  Attribute: IVP_ERD_INVOICE_NO_ATR
    Data_type: IVP_ERD_INTEGER_DTYPE
  Attribute: IVP_ERD_SENT_TO_ATR
    Data_type: IVP_ERD_CHAR_32_DTYPE
  Identifier: IVP_ERD_INVOICE_ID_IDENT
    Attribute: IVP_ERD_INVOICE_NO_ATR
      Data_type: IVP_ERD_INTEGER_DTYPE
```

8. Create the hierarchy shown in IVP_ERD_CONTRACT_ENT Hierarchy for the **IVP_ERD_CONTRACT_ENT** entity.

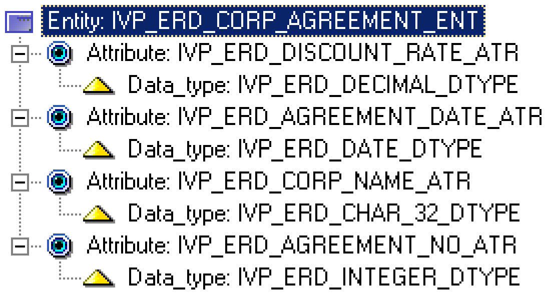
IVP_ERD_CONTRACT_ENT Hierarchy



```
Entity: IVP_ERD_CONTRACT_ENT
  Attribute: IVP_ERD_PICKUP_DATE_ATR
    Data_type: IVP_ERD_DATE_DTYPE
  Attribute: IVP_ERD_PLANNED_DROPOFF_ATR
    Data_type: IVP_ERD_DATE_DTYPE
  Attribute: IVP_ERD_ACTUAL_DROPOFF_ATR
    Data_type: IVP_ERD_DATE_DTYPE
  Attribute: IVP_ERD_CONTRACT_STATUS_ATR
    Data_type: IVP_ERD_CHAR_15_DTYPE
  Attribute: IVP_ERD_LOCAL_ADDR_STREET_ATR
    Data_type: IVP_ERD_CHAR_32_DTYPE
  Attribute: IVP_ERD_LOCAL_ADDR_CITY
    Data_type: IVP_ERD_CHAR_15_DTYPE
  Attribute: IVP_ERD_LOCAL_ADDR_ST_PROV_ATR
    Data_type: IVP_ERD_CHAR_15_DTYPE
  Attribute: IVP_ERD_LOCAL_ADDR_COUNTRY_ATR
    Data_type: IVP_ERD_CHAR_15_DTYPE
  Attribute: IVP_ERD_STARTING_MILEAGE_ATR
    Data_type: IVP_ERD_INTEGER_DTYPE
  Attribute: IVP_ERD_RETURN_MILEAGE_ATR
    Data_type: IVP_ERD_INTEGER_DTYPE
  Attribute: IVP_ERD_FREE_MILES_ATR
    Data_type: IVP_ERD_INTEGER_DTYPE
  Attribute: IVP_ERD_MILEAGE_RATE_ATR
    Data_type: IVP_ERD_DECIMAL_DTYPE
  Attribute: IVP_ERD_TAX_ATR
    Data_type: IVP_ERD_DECIMAL_DTYPE
  Attribute: IVP_ERD_DISCOUNT_PERCENT_ATR
    Data_type: IVP_ERD_DECIMAL_DTYPE
  Attribute: IVP_ERD_DISCOUNT_AMT_ATR
    Data_type: IVP_ERD_DECIMAL_DTYPE
  Attribute: IVP_ERD_CONTRACT_NO_ATR
    Data_type: IVP_ERD_INTEGER_DTYPE
  Identifier: IVP_ERD_CONTRACT_ID_IDENT
    Attribute: IVP_ERD_CONTRACT_NO_ATR
      Data_type: IVP_ERD_INTEGER_DTYPE
```

9. Create the hierarchy shown in IVP_ERD_CORP_AGREEMENT_ENT hierarchy for the **IVP_ERD_CORP_AGREEMENT_ENT** entity.

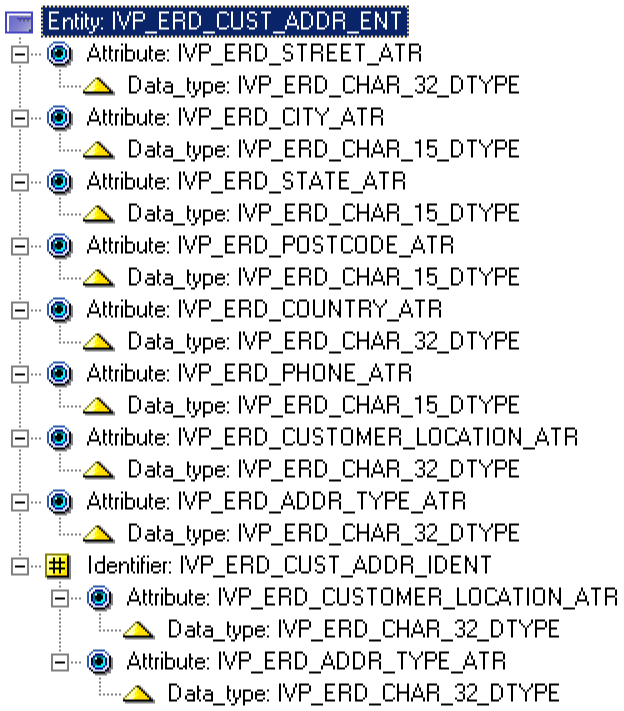
IVP_ERD_CORP_AGREEMENT_ENT hierarchy



```
Entity: IVP_ERD_CORP_AGREEMENT_ENT
  Attribute: IVP_ERD_DISCOUNT_RATE_ATR
    Data_type: IVP_ERD_DECIMAL_DTYPE
  Attribute: IVP_ERD_AGREEMENT_DATE_ATR
    Data_type: IVP_ERD_DATE_DTYPE
  Attribute: IVP_ERD_CORP_NAME_ATR
    Data_type: IVP_ERD_CHAR_32_DTYPE
  Attribute: IVP_ERD_AGREEMENT_NO_ATR
    Data_type: IVP_ERD_INTEGER_DTYPE
```

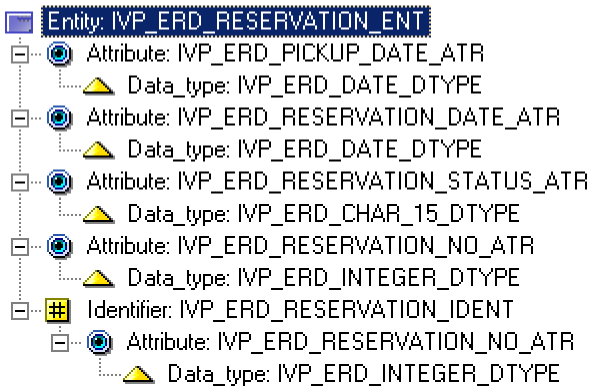
10. Create the hierarchy shown in IVP_ERD_CUST_ADDRESS_ENT hierarchy for the **IVP_ERD_CUST_ADDRESS_ENT** entity.

IVP_ERD_CUST_ADDRESS_ENT hierarchy


```
Entity: IVP_ERD_CUST_ADDR_ENT
  Attribute: IVP_ERD_STREET_ATR
    Data_type: IVP_ERD_CHAR_32_DTYPE
  Attribute: IVP_ERD_CITY_ATR
    Data_type: IVP_ERD_CHAR_15_DTYPE
  Attribute: IVP_ERD_STATE_ATR
    Data_type: IVP_ERD_CHAR_15_DTYPE
  Attribute: IVP_ERD_POSTCODE_ATR
    Data_type: IVP_ERD_CHAR_15_DTYPE
  Attribute: IVP_ERD_COUNTRY_ATR
    Data_type: IVP_ERD_CHAR_32_DTYPE
  Attribute: IVP_ERD_PHONE_ATR
    Data_type: IVP_ERD_CHAR_15_DTYPE
  Attribute: IVP_ERD_CUSTOMER_LOCATION_ATR
    Data_type: IVP_ERD_CHAR_32_DTYPE
  Attribute: IVP_ERD_ADDR_TYPE_ATR
    Data_type: IVP_ERD_CHAR_32_DTYPE
  Identifier: IVP_ERD_CUST_ADDR_IDENT
    Attribute: IVP_ERD_CUSTOMER_LOCATION_ATR
      Data_type: IVP_ERD_CHAR_32_DTYPE
    Attribute: IVP_ERD_ADDR_TYPE_ATR
      Data_type: IVP_ERD_CHAR_32_DTYPE
```

11. Create the hierarchy shown in IVP_ERD_RESERVATION_ENT hierarchy for the **IVP_ERD_RESERVATION_ENT** entity.

IVP_ERD_RESERVATION_ENT hierarchy


```
Entity: IVP_ERD_RESERVATION_ENT
  Attribute: IVP_ERD_PICKUP_DATE_ATR
    Data_type: IVP_ERD_DATE_DTYPE
  Attribute: IVP_ERD_RESERVATION_DATE_ATR
    Data_type: IVP_ERD_DATE_DTYPE
  Attribute: IVP_ERD_RESERVATION_STATUS_ATR
    Data_type: IVP_ERD_CHAR_15_DTYPE
  Attribute: IVP_ERD_RESERVATION_NO_ATR
    Data_type: IVP_ERD_INTEGER_DTYPE
  Identifier: IVP_ERD_RESERVATION_IDENT
    Attribute: IVP_ERD_RESERVATION_NO_ATR
      Data_type: IVP_ERD_INTEGER_DTYPE
```

12. Select **File > Commit** from the Construction Workbench menu. Name the entity relationship diagram IVP_ERD_EX_CUSTOMER.

Building the Hierarchy

Building a hierarchy includes the following:

- Adding Objects to a Hierarchy
- Removing Objects from a Hierarchy
- Removing Relationships Between Objects

Adding Objects to a Hierarchy

You can add an existing object to the hierarchy or create a new object. There are several ways you can add an object to the hierarchy. To insert an object to the hierarchy, do the following:

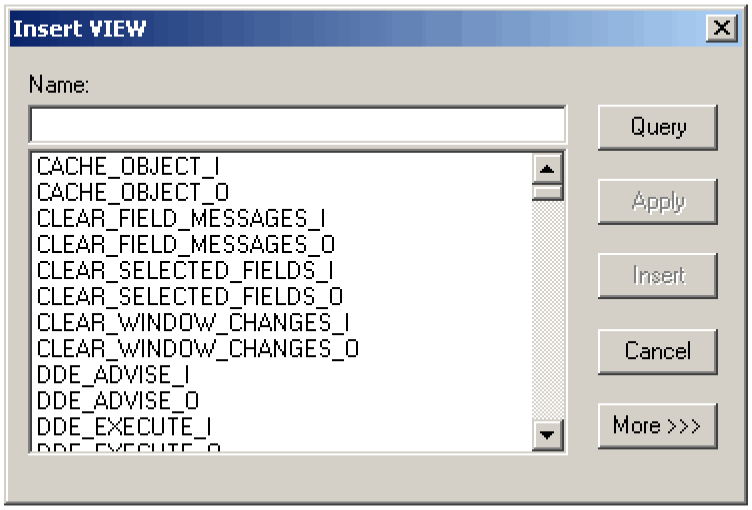
1. Select an existing object in the hierarchy.
2. Follow one of the next procedures.

- In the Hierarchy - objects toolbar, click the icon representing the object to add. The available objects depend on which object you have selected in the hierarchy.
- Right-click an existing object in the hierarchy and select **Insert**, **Insert Child**, or **Insert Sibling**, or **Insert Sibling** [*Object*], where [*Object*] is the same type of object you selected, from the pop-up menu.

- Click the **Insert Child** or **Insert Sibling** button in the Hierarchy - operations toolbar and select the object to insert from the pop-up menu.
- From the Construction Workbench menu, select **Insert > Insert Child**, or **Insert > Insert Sibling**, or **Insert Sibling** [*Object*], where [*Object*] is the same type of object you selected.
- If you are in the Repository tab of Hierarchy Window, select **Insert** > [*Menu*], then select the object, where *Menu* is one of the following menus: Development, ~~OO Development~~, Database Diagram, State Transition Diagram, Entity Relationship Diagram, Process Dependency Diagram, Configuration.

The Insert *object* dialog displays as shown in Insert dialog example.

Insert dialog example



CACHE_OBJECT_I
CACHE_OBJECT_O
CLEAR_FIELD_MESSAGES_I
CLEAR_FIELD_MESSAGES_O
CLEAR_SELECTED_FIELDS_I
CLEAR_SELECTED_FIELDS_O
CLEAR_WINDOW_CHANGES_I
CLEAR_WINDOW_CHANGES_O
DDE_ADVISE_I
DDE_ADVISE_O
DDE_EXECUTE_I

Unless explicitly specified, AppBuilder adds the object as a child or sibling, depending on your selected option in the **Hierarchy** set of options in the Workbench Options window.

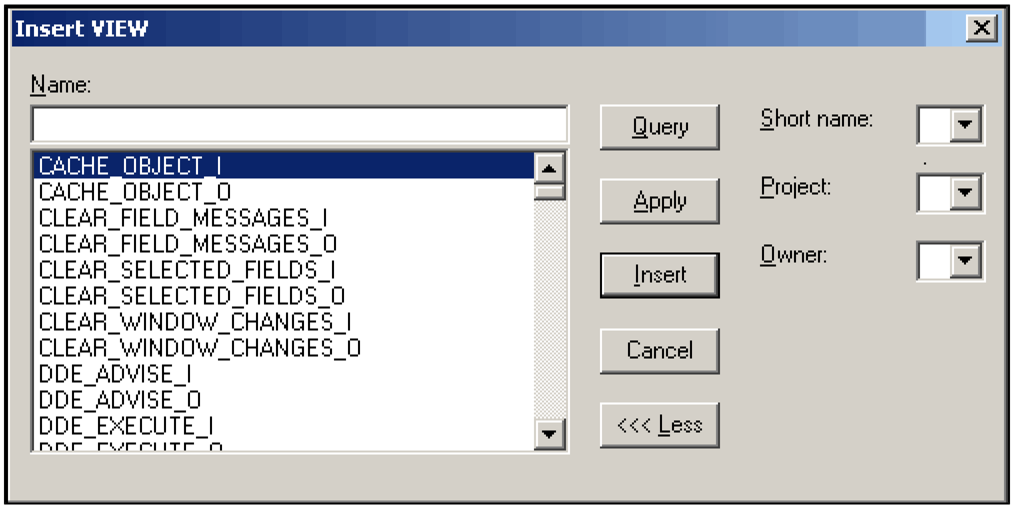
3. If you are creating a new object, in the **Insert** dialog, type the name of the object without spaces. Use underscore (_) to break up the parts of the name. Click **Insert** to create and add the object to the hierarchy.

4. If you are inserting objects that already exist in the repository, type part (first few characters) of the name in the **Name** field and click **Query**.

You can also click Query without entering characters in the Name field to display a complete list of that type of object.

You can narrow your query by specifying more parameters. Click the **More** button, and the Insert dialog is expanded as shown in Extended Insert dialog example.

Extended Insert dialog example



CACHE_OBJECT_I
CACHE_OBJECT_O
CLEAR_FIELD_MESSAGES_I
CLEAR_FIELD_MESSAGES_O
CLEAR_SELECTED_FIELDS_I
CLEAR_SELECTED_FIELDS_O
CLEAR_WINDOW_CHANGES_I
CLEAR_WINDOW_CHANGES_O
DDE_ADVISE_I
DDE_ADVISE_O
DDE_EXECUTE_I

Use any combination of the following parameters in addition to the string you typed in the **Name** field to be used as your query criteria.

- **Short name** : If you are using this field as your query criteria, type the system ID (SHORTNAME property) of an object.
- **Project** : Select a project from the pull-down list to query objects within the specified project.
- **Owner** : Select the owner from the pull-down list to query objects that belong to a specific owner.

The list of objects that match your query criteria is shown in the dialog. Select one or more of the objects from the list. Use **Shift** or **Ctrl** to select multiple objects.

5. Click **Apply** to insert the selected objects without closing the Insert dialog. The **Apply** button is useful when you are inserting several objects of the same type. Click **Insert** to insert the object and close the dialog. Click **Cancel** to close the dialog without inserting an object.

This procedure adds an object to the hierarchy – not to the repository. When you add an object or relationship, the change you make is not stored in the repository until you commit it. From the Construction Workbench menu bar, click **File > Commit**.

You can also work with individual objects from the repository using the Repository Tab.

Removing Objects from a Hierarchy

To remove an object from a hierarchy, complete the following steps.

If you select All in the hierarchy, all objects are selected except the Repository Heading (Identifier) to which you are connected. This selects all the hierarchy objects only.

1. Select the object in the hierarchy.
2. Do one of the following:

- Click the **Clear** button in the Hierarchy - operations toolbar. AppBuilder removes the object from the hierarchy.
- Select **Edit > Clear** from the Construction Workbench menu.
- Press the **Del** key.
- Right-click the object in the hierarchy and select **Clear** from the pop-up menu. You can also choose **Select All** from the pop-up menu to select all objects in the hierarchy except the repository identifier.

This procedure removes the object from the hierarchy – not from the repository. When you remove an object or relationship, the change you make is not stored in the repository until you commit it. From the Construction Workbench menu bar, click **File > Commit**.

Removing Relationships Between Objects

Deleting a relationship in a hierarchy removes the relationship between the selected object and its parent, but both objects remain in the repository.

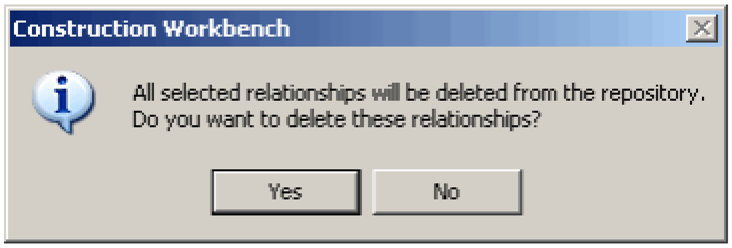
To delete a relationship, complete the following steps:
1. In the Hierarchy window, select the child object of the relationship to remove.

2. Do one of the following:

- Click the **Delete Relationship** button in the Hierarchy - operations toolbar. The system prompts for confirmation.
- Select **Edit > Delete Relationship** from the Construction Workbench menu.
- Press the **Ctrl+Backspace** keys.
- Right-click the object in the hierarchy and select **Delete Relationship** from the pop-up menu.

3. In the confirmation window (see Remove relationship confirmation dialog), click **Yes**. The system deletes the relationship from the repository.

Remove relationship confirmation dialog


Construction Workbench

All selected relationships will be deleted from the repository.
Do you want to delete these relationships?

This procedure removes the relationship from the hierarchy – not from the repository. When you remove an object or relationship, the change you make is not stored in the repository until you commit it. From the Construction Workbench menu bar, click **File > Commit**.

Customizable Tool Tips

When you roll your cursor over any of the objects in the hierarchy and pause, a ToolTip (a yellow box with information about that object) is displayed. For most objects, only the object type and object name are displayed. But for other objects, more information is displayed. The type of information to display – one or more of the properties of that object – is based on a file that you can configure. The file is tooltips.dat and is in **<AppBuilder>**\AD\CFG\DATA, where **<AppBuilder>** is the letter and directory where AppBuilder is installed. You can edit this file with a text editor. You can add or remove the properties to display.

For example,

```
[RULE]
DBMS Usage=RULE.Sys_Source
Execution Environment=RULE.Exec_Environ
Implementation name=RULE.Rule_Imp_Name

[VIEW]
Type=OWNS_VIEW.View_Usage
Occurs=VIEW_INCLUDES.Occurs_Times

[FIELD]
Type=FIELD.Field_Type
Length=FIELD.Fld_Len
Fraction=FIELD.Fld_Frac

[SET]
Type=SET.Set_Type
Length=SET.Set_Length
Fraction=SET.Set_Fraction
Style=SET.Set_Style

[VALUES]
Symbol=SET_CONTAINS_VALUE.Symbol
Value=VALUES.Values_Imp_Name
```

Each object has a section heading, such as [RULE] and then a list of properties. On each line of the list:

```
Description=TYPE.Property
```

where **TYPE** is the type of object or relationship (**RULE** , or **OWNS_VIEW**).

If you want to add more information, say for a component, you can add a new section for that object and then the list of properties to display. For example, you can add the following:

```
[COMPONENT]
DBMS Usage=COMPONENT.DBMS_Usage
Execution Environment=COMPONENT.Exec_Environ
Language=COMPONENT.Language
Implementation name=COMPONENT.Comp_Imp_Name
```

The properties to display must be one of the properties of that object. After the file is edited and saved, you can see the result in the AppBuilder hierarchy in the Construction Workbench.

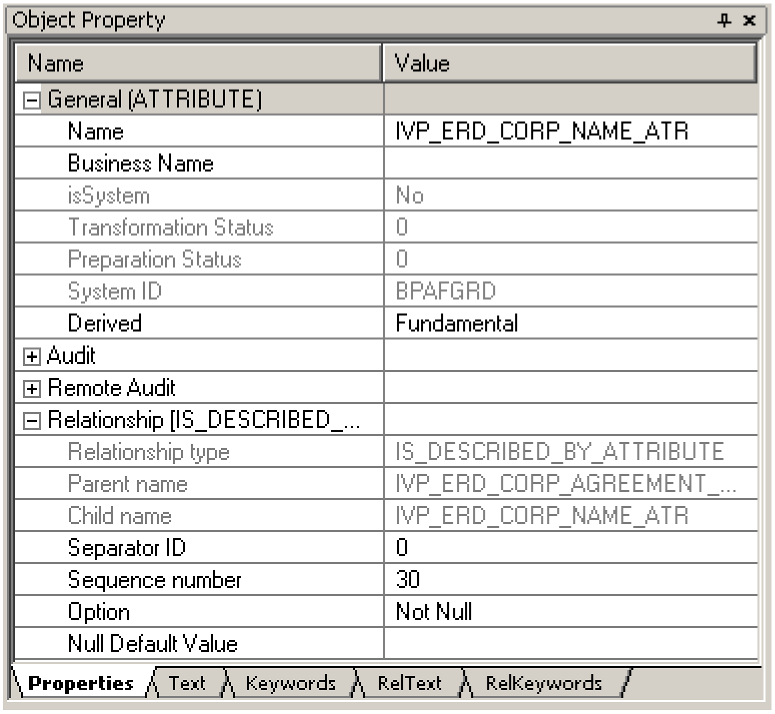
Hierarchy Object Properties

The Object Property window displays by default when you select a hierarchy object. In case it was previously closed and it's not displayed, use one of the following three methods:

- Right-click the object in the hierarchy and select **Properties** from the pop-up menu.
- Select the object in the hierarchy, then select **View > Toolbars > Object Property** from the Construction Workbench menu.
- Click to select the object in the hierarchy, or double-click it.

The properties for an object vary depending on the type of object; therefore, the actual window may vary, depending on the object you selected. See Object Property window sample.

Object Property window sample



Name	Value
⊟ General (ATTRIBUTE)	
Name	IVP_ERD_CORP_NAME_ATR
Business Name	
isSystem	No
Transformation Status	0
Preparation Status	0
System ID	BPAFGRD
Derived	Fundamental
⊞ Audit	
⊞ Remote Audit	
⊟ Relationship (IS_DESCRIBED_...	
Relationship type	IS_DESCRIBED_BY_ATTRIBUTE
Parent name	IVP_ERD_CORP_AGREEMENT_...
Child name	IVP_ERD_CORP_NAME_ATR
Separator ID	0
Sequence number	30
Option	Not Null
Null Default Value	

The changes are available to the workstation for as long as the Construction Workbench is open and the connection to the repository is maintained. (Closing the project does not lose the changes.) Saving is not the same as committing to the repository. If you want to make the changes available to others using the repository, you must commit the changes.
The Object Property window contains the following tabs:

- Properties Tab
- Text Tab
- Keywords Tab
- RelText Tab
- RelKeywords Tab

Properties Tab

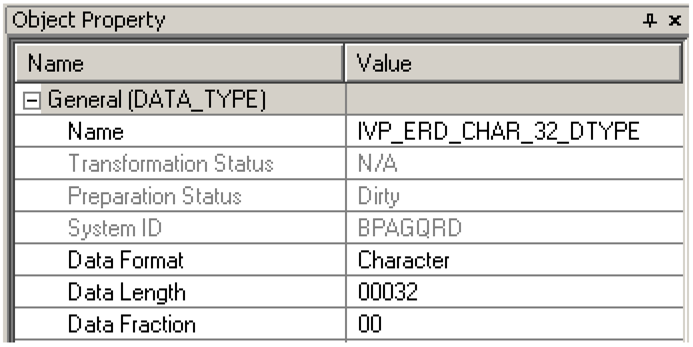
The Properties tab contains the following sets of properties:

- General Properties
- Audit Properties
- Remote Audit Properties
- Relationship Properties

General Properties

The **General** properties specify the general properties for the object. The fields in this section might vary depending on the object. Not every property is available for each object.

General properties section



Object Property

Name	Value
⊟ General (DATA_TYPE)	
Name	IVP_ERD_CHAR_32_DTYPE
Transformation Status	N/A
Preparation Status	Dirty
System ID	BPAGQRD
Data Format	Character
Data Length	00032
Data Fraction	00

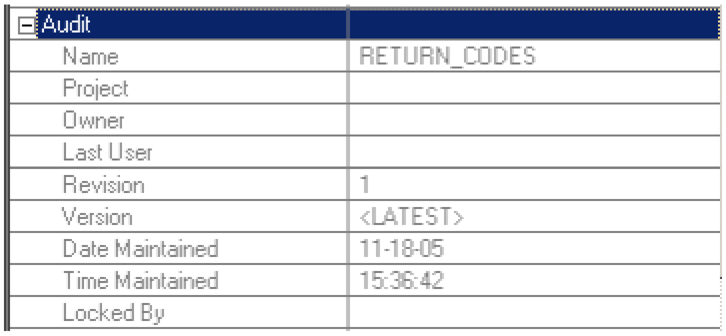
General properties description

Property	Description
Name	The name of the object.
System ID	The unique system-generated identifier.
Description	A description of the object that you are adding.
Implementation name	The implementation name of the object; this is usually the system identifier, but you can customize it according to naming conventions for your project.

Audit Properties

The **Audit** properties specify information that can help you track the usage of this entity locally. This is version control information. This information can only be viewed and cannot be edited.

Audit properties section



⊟ Audit	
Name	RETURN_CODES
Project	
Owner	
Last User	
Revision	1
Version	<LATEST>
Date Maintained	11-18-05
Time Maintained	15:36:42
Locked By	

Audit properties description

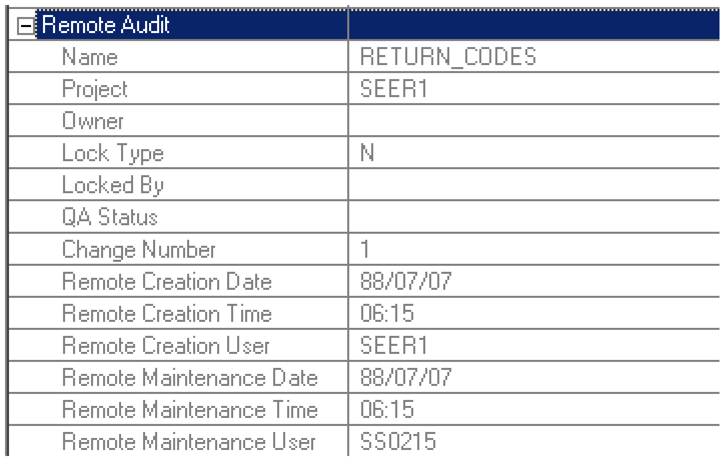
Property	Description
Name	The long name of the object.

Project	The name of the project to which this object belongs.
Owner	The user name of the creator of this object.
Last user	The user name of the person who last modified the object.
Revision	The number of revisions – incremented each time you change the object.
Version	This is not used on the workstation; simply says <LATEST>.
Date maintained	The date (year, month, day) that the object was last stored in repository.
Time maintained	The time (hour, minutes, seconds) that the object was last stored in repository.
Locked by	The user name of the person who has the object locked.

Remote Audit Properties

The **Remote Audit** properties helps you track information about remote (mainframe) usage of this entity. This information can only be viewed and cannot be edited.

Remote Audit properties section



Remote Audit	
Name	RETURN_CODES
Project	SEER1
Owner	
Lock Type	N
Locked By	
QA Status	
Change Number	1
Remote Creation Date	88/07/07
Remote Creation Time	06:15
Remote Creation User	SEER1
Remote Maintenance Date	88/07/07
Remote Maintenance Time	06:15
Remote Maintenance User	SS0215

Remote Audit properties description

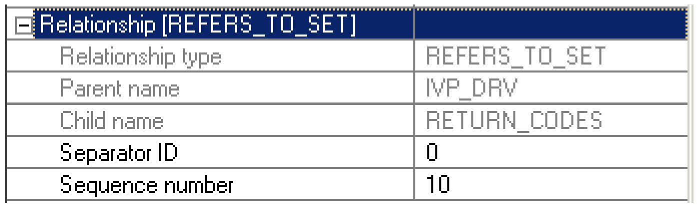
Property	**Description**
Name	The name of the object.
Project	The name of the Project of which this object is a child.
Owner	The user name of the person who created this object.
Lock Type	The type of the lock can be put on an entity. There are four types of lock: Y=Download, R=Read, U=Update, and E=Exclusive.
Locked By	The ID of owner or user who has locked the object.
QA Status	Used in reference to host migration. See the *Enterprise Migration Guide* .
Change Number	Used for synchronizing with the mainframe for uploads and downloads.
Remote Creation Date	The date information of the object on the mainframe side.
Remote Creation Time	The time information of the object on the mainframe side.
Remote Creation User	The user name of the object on the mainframe side.
Remote Maintenance Date	The date (year/month/day) the entity was last modified on the mainframe side.
Remote Maintenance Time	The time, based on a 24-hour clock, the entity was last modified on the mainframe side.

Remote Maintenance User	The user ID of the person who last modified the entity on the mainframe.

Relationship Properties

The **Relationship** properties specify the relationship properties for the object. The fields in this section might vary depending on the object. Not every property is available for each object. Use them to define the relationship of each object to its parent in the hierarchy.

Relationship properties section



Relationship [REFERS_TO_SET]	
Relationship type	REFERS_TO_SET
Parent name	IVP_DRV
Child name	RETURN_CODES
Separator ID	0
Sequence number	10

Relationship properties description

Property	**Description**
Relationship type	The type of the relationship.
Parent name	Parent object of the relationship.
Child name	Child object of the relationship, in this case the selected object.
Separator ID	A number automatically assigned to each entity in a sequence and used to define the order of relationships attached to a common entity.
Sequence number	For internal purposes. Uniquely distinguishes duplicate relationships between the same parent and child entities.
View usage	This field is displayed only for View objects.
Null indicator	This field is displayed only for Field objects. The available options are: Null with Default, Not Null with Default, Not Null, Redefines View, Null.
Null Default Value	This field is displayed only for Field objects. Insert the default value in case the Null indicator property has one of the Null with Default or Not Null with Default values.
Occurs Times	This field is displayed only for Field objects.

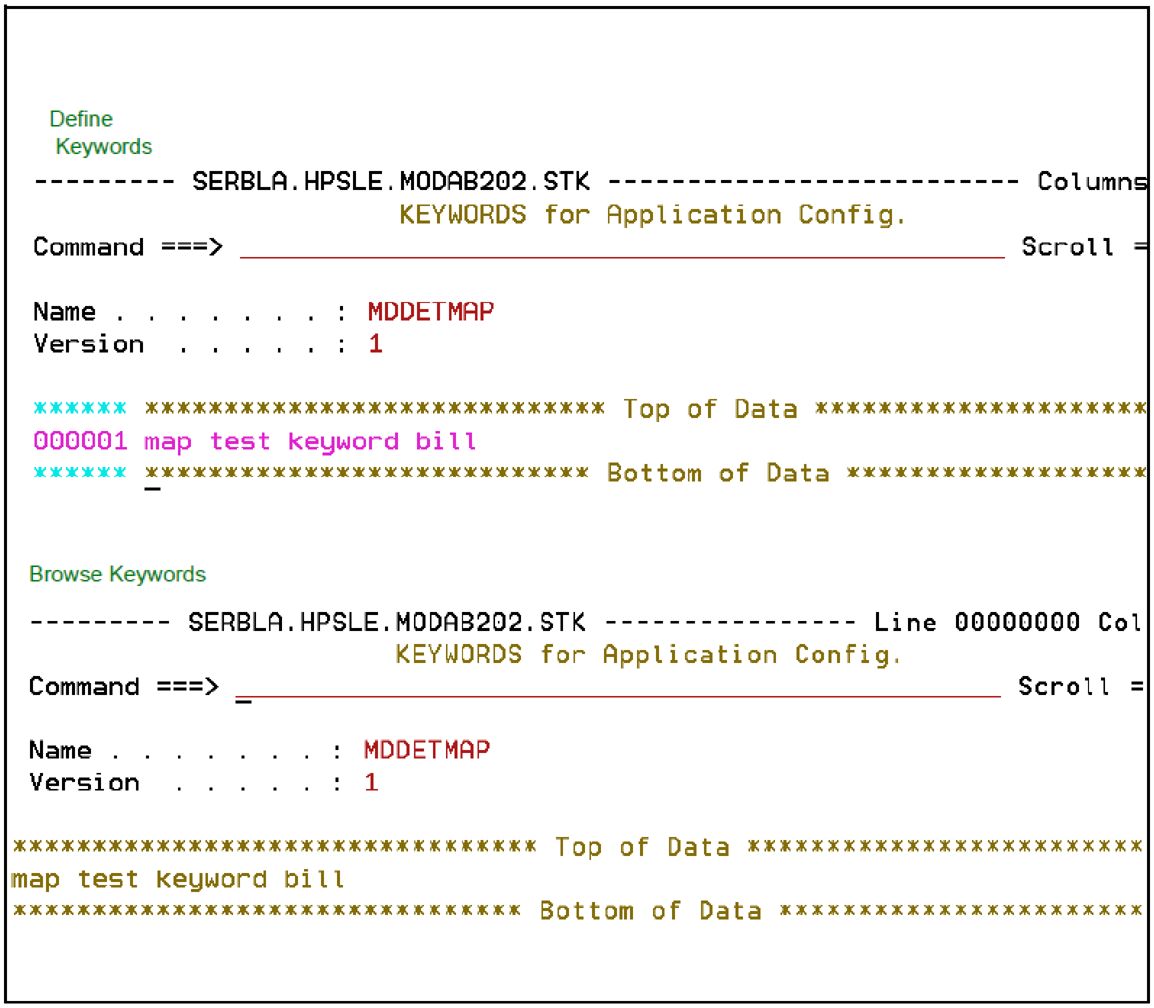
Text Tab

The **Text** tab has text information, notes, that you might want to keep with this entity. It might be a more complete description or special instructions about the use of this entity.

Keywords Tab

You can create a list of keywords to make searching for an entity easier using the **Define Keywords** command. Type your keyword list here.
The way the list of keywords is displayed in the mainframe screen is different from the way they are displayed on the workstation in the Keywords tab of the properties dialog for that entity, but they function the same.
The results of a define keywords and a browse keywords commands are shown in Keywords on mainframe. The keywords (in this example: map, test, keyword, and bill) are listed horizontally across the screen in both cases.

Keywords on mainframe



Define
Keywords

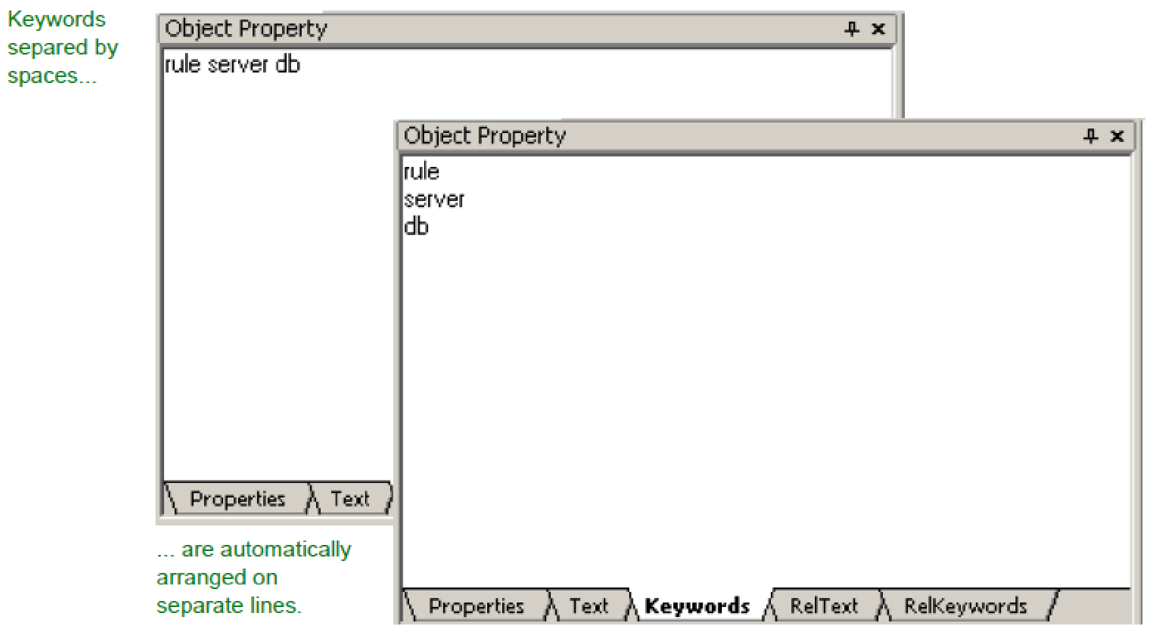
```
--------- SERBLA.HPSLE.MODAB202.STK -------------------------- Columns
                      KEYWORDS for Application Config.
Command ===> ________________________________________________ Scroll =

Name . . . . . . . : MDDETMAP
Version  . . . . . : 1

****** ****************************** Top of Data *********************
000001 map test keyword bill
****** ***************************** Bottom of Data *******************
```

Browse Keywords

```
--------- SERBLA.HPSLE.MODAB202.STK ---------------- Line 00000000 Col
                      KEYWORDS for Application Config.
Command ===> ________________________________________________ Scroll =

Name . . . . . . . : MDDETMAP
Version  . . . . . : 1

********************************** Top of Data *************************
map test keyword bill
********************************* Bottom of Data **********************
```

On the workstation, create keywords by typing them in the **Keywords** tab of the Object Property window for that entity. When you begin to type the words, you can type them on one line, separated by a space, or one at a time and press **Enter** after each. Both methods are equivalent. Next time when you come back to the Keywords tab, they are arranged vertically as shown in Keywords on workstation, regardless of how they are displayed when you typed them originally.

Keywords on workstation



RelText Tab

Any text associated with the relationship is displayed in this tab. Refer to Text Tab for more information.

RelKeywords Tab

Any keywords associated with the relationship are displayed in this tab. Refer to Keywords Tab for more information.

Hierarchy Objects

Use the icons on the Hierarchy - objects toolbar to add new objects to the application hierarchy. Which objects appear in the toolbar depend on where you are in the hierarchy. Only those objects that can be placed appear in the toolbar. You can also add existing objects from the repository to your project. See Repository Tab for more information.
Hierarchy objects that can be inserted in the Repository tab are classified as:

- Development Objects
- -[OO Development Objects|Hierarchy Objects#50606134_41556]-
  * [Native File objects|Hierarchy Objects#50606134_41563]
- Database Diagram Objects
- Entity Relationship Diagram Objects
- State Transition Diagram Objects
- Process Dependency Diagram Objects
- Configuration Objects

Development Objects

To see the list of all availabledevelopment objects, select the **Repository** tab of Hierarchy window, then select **Insert > Development**. The list of development objects is displayed in the Development submenu. AppBuilder development objects presents them alphabetically as reference.

AppBuilder development objects

Bitmap	File	Report	Value
Component	Function	Rule	View
Component Folder	Physical Event	Section	Window
Field	Process	Set	

Function

A Function represents the major class or activity of your application. It is the highest level identifier in the project. The name of the function that you create is the name of the application. A Function contains groups of processes.

When prepared for execution from the workstation, the function typically is displayed as a Start menu option for Windows platforms or a pull-down menu option that, when selected, executes the application. Each process under the function is displayed as a menu choice in the menu for that function. You can use the **Run** menu to execute directly from the Construction Workbench, and you can execute from the **Start** menu in Windows using the following procedure:

1. Select **Start > All Programs > AppBuilder > Execution Clients**.
2. Select either **Java Client** or **Windows Client** , depending on your implementation.

If you have selected the **Create Start menu shortcut** option from the Preparation section of the Workbench Options window, the prepared function becomes a menu choice from **Start > All Programs** directly as its own application. The **Create Start menu shortcut** option is only applicable for Windows applications; it is not supported in Java.
A project has the relationship between a function and its application configuration, which enables you to find out which application configurations are used to partition the application. This information plus additional settings comprise a project.

> ⚠ AppBuilder automatically adds the top-level function to a project hierarchy. Use this procedure when adding a function to another hierarchy.

To add a Function to the hierarchy, see Adding Objects to a Hierarchy. To modify the properties of the object, see Hierarchy Object Properties. To define the relationship of the object to its parent in the hierarchy, see Relationship Properties.

Process

A Process represents the application, subsystem, or set of applications within a high-level function. Each process contains logical units of work. In an AppBuilder application, processes appear as options on pull-down menus at runtime.
A process is a set of predefined activities that create, read, update, or delete the data the business group uses. Events trigger processes, which are described by a verb-noun combination.
To add a Process to the hierarchy, see Adding Objects to a Hierarchy. To modify the properties of the object, see Hierarchy Object Properties. To define the relationship of the object to its parent in the hierarchy, see Relationship Properties.
For more information about a Process in conjunction with configuring an application, refer to the *Deploying Applications Guide*.

Rule

In an AppBuilder application, Rules are programming statements that define the logic of the application, or how the application works. Each rule corresponds to a module that executes at runtime. To make rules reusable, design each rule to perform only one task, such as routing the logic flow to another rule, displaying a window, accessing a database, or performing a calculation. This approach maximizes the potential for reuse. If you combine several tasks into one rule and then perform one or two of them again, it might not be feasible to reuse your original rule because it also performs unneeded tasks.
A rule can be a parent or child rule, depending on its position in the hierarchy.
The **root rule** is the first module executed when the user selects a menu choice that corresponds to the parent process of the rule. The root rule typically displays the primary window. The child rule of that root rule may display a secondary window. The rule at the very bottom of the hierarchy is called the leaf rule. Leaf rules typically do not display windows. These rules can access a database.
Use the Rule Painter tool to create the Rules Language code for the Rule object. See Rule Painter in Construction Tools for more information.
For tips on developing applications using rules, refer to the *Developing Applications Guide*. For details about Rules Language syntax, refer to the *Rules Language Reference Guide* .
To add a Rule to the hierarchy, see Adding Objects to a Hierarchy. To modify the properties of the object, see Hierarchy Object Properties. To define the relationship or the object to its parent in the hierarchy, see Relationship Properties.

Window

Windows are user interface displays or screens that accept input or display data to the end user. Windows enable users to manipulate data and perform actions. Each window is associated with one **view** , called its window view, through the **Owns** relationship (see View for more information about Views). The window view contains the actual data elements (such as fields) for the window and stores input to and output (new or modified data for the end user) from the window.
See Window Painter for more information about creating windows and adding objects to windows.
To add a Window to the hierarchy, see Adding Objects to a Hierarchy. To modify the properties of the object, see Hierarchy Object Properties. To define the relationship of the object to its parent in the hierarchy, see Relationship Properties.

View

A View is a structure that contains data elements, such as Fields or other Views, and defines the data structure of the application in the AppBuilder environment.
A View attached to a Window is called a window view. These views define data displayed to a user, data entered by a user, and data stored by the system. For example, a View defines the input to or the output from a Rule or the Fields that can appear on a Window. The order of the Fields listed in the hierarchy under the View is not related to the order of their appearance in the Window.
There is a limitation for 3270 Window views of 4026 entities of any combination of views and fields. This includes occurring subviews; each occurrence of a view or field in an occurring view is calculated as another entity. For example, if you have an occurring view that has 10 occurrences and that view has three fields, the parent view would have 40 entities (10 for the occurring view and 10 for each field). If you exceed the limit, the window fails to prepare.

Each view is similar to a node in a data tree – the leaf nodes of which are fields (see Field for more information about fields). The root view of each such structure is associated with a rule, a component, a window, a file, or a section entity type through an **Owns** relationship type, and with fields and other views through an **Includes** relationship type. Any included views can include other views and fields.
To add a View to the hierarchy, see Adding Objects to a Hierarchy. To modify the properties of the object, see Hierarchy Object Properties. To define the relationship of the object to its parent in the hierarchy, see Relationship Properties.

Field

A Field represents the smallest unit of data used in the AppBuilder environment. The fundamental building block in any application is the variable, which in AppBuilder is the field object. Fields can record the input/output definition of objects or contain information about part of a file (such as a column in a database table). An edit field object defines a field in which the end user can either enter and change data or view read-only data. The format and length are two crucial properties of a Field.

> When setting the Range-Minimum value of a field of format Small Integer or Integer, you must set the Range-Maximum value before inputting the minimum value

To add a Field to the hierarchy, see Adding Objects to a Hierarchy. To modify the properties of the object, see Hierarchy Object Properties. To define the relationship of the object to its parent in the hierarchy, see Relationship Properties.

Set

Use sets for messages, window object domains, and to simplify rules. Use the Set Builder tool to define the members of a set. The members of a set consist of either symbols or values.

- Symbol – An entity that represents the name of a particular value within a set.
- Value – An entity that represents a constant value. See Value for more information.

For more information about building sets, refer to Set Builder in Construction Tools.
Use sets to create named values in the repository that can then be used by any number of rules in any number of applications. Sets can be used as **system choices** or **messages**.

- **System choices** - Choices available to window objects, for example to a combo box in an application window where an end user might have to choose the month from a displayed list of January, February, etc.
- **Messages** - Predefined strings, such as error messages, in a set referred to by the system component, SET_WINDOW_MESSAGE, that displays the message. These are used primarily to make Rules Language code more readable by enabling developers to substitute meaningful references for cryptic variables. For example, when you create a message set, each message is associated with a value (such as 100). To invoke a particular message you can specify that value in your Rules Language code. Alternatively, you can associate a meaningful symbolic name with the message (for example, RECORD_NOT_FOUND) and refer to the message using the symbolic name instead of the value.

To add a Set to the hierarchy, see Adding Objects to a Hierarchy. To modify the properties of the object, see Hierarchy Object Properties. To define the relationship of the object to its parent in the hierarchy, see Relationship Properties.
There are four different types (or styles) of sets:

- Define Set
- Lookup Set
- Error Set
- Values Set

Define Set

A Define Set is the simplest set created and modified in the Set Builder and is displayed with two columns, Define and Encoding. Both of these columns are mandatory fields, and the value entered in the Define field must be unique.

Lookup Set

A Lookup Set provides more functionality than a Define Set, and (among other things) specifies the possible values appearing in a combo box or other window object in an application window. In addition to the Define and Encoding columns, this set includes a Display column. The Define, Encoding, and Display column fields are mandatory. The entries in these fields must be unique. If an entry is displayed for one symbol in any of the language columns, then the fields for that column for all other defined symbols also must be filled.

Error Set

An Error Set is similar to a Lookup Set, but provides additional functionality for application error messages. There are essentially two significant differences: this set can only be of type small integer (the other kinds of sets can be any of several data types) and you can associate text with the symbols. This text becomes the error message associated with the error number the symbol defines and is displayed to users when the error occurs (assuming that the rule in question uses error-handling via the standard system components). If multiple languages are defined, you can associate different text for each language with a single symbol.

Values Set

The Values Set is supported mainly for legacy applications. Use the other types of set for new application development. In a Values set, the symbol property of a set can be used as a substitute for the value entity type to which it relates. For example, many sets can contain the value 1. However, the relationship type between the value 1 and the set Months can have a symbol of "JAN," while the relationship type between the value 1 and the set Mode can have the symbol "on." Conventions for the symbol and its accompanying text depend on the usage of the set (for instances of the Set-Contains-a-Value relationship).

Value

Values represent a constant value within a Set. Because a value can belong to more than one Set, you can use a single value in multiple ways. For example, the value **4** may represent **April** in the set **MONTHS** , but it may also represent **Wednesday** in the set **DAYS**.
To add a Value to the hierarchy, see Adding Objects to a Hierarchy. To modify the properties of the object, see Hierarchy Object Properties. To define the relationship of the object to its parent in the hierarchy, see Relationship Properties.

Bitmap

Bitmap objects contain graphic image files used in windows. They can also be used with Hot Spot to create hyperlinks that you can click to jump to a file. A bitmap can be a device-dependent or device-independent picture image read from a file.

⚠ You must use the correct bitmap image for the targeted execution environment.

For more information about viewing bitmap images, see Bitmap Viewer.
To add a Bitmap to the hierarchy, see Adding Objects to a Hierarchy. To modify the properties of the object, see Hierarchy Object Properties. To define the relationship of the object to its parent in the hierarchy, see Relationship Properties.

Component

There are two kinds of components: system components and user components. System components are components that come with AppBuilder for your use in developing Applications. User components contain computer code (for example, Assembler, C, COBOL, or Java) to accomplish functions that are not possible with the AppBuilder Rules Language. User components are different from functions in pre-defined system components. For information about system components, refer to the *System Components Reference Guide*. For more information about developing user components, refer to the *Developing Applications Guide*.
User components might include:

- Complicated mathematical functions, such as square root
- Non-SQL data access logic, such as the IMS database interface
- Hardware specific functions for the selected execution environment

⚠ Because user components are written in a specific language, they are not portable between environments.

When creating a new component, make sure that the object properties correctly specify the component's language and execution environment. For more information about Component Painter, refer to Searching for Regular Expressions in Construction Tools.
To add a component to the hierarchy, see Adding Objects to a Hierarchy. To modify the object properties, see Hierarchy Object Properties. To define the relationship of the object to its parent in the hierarchy, see Relationship Properties.

File

A File represents the physical files used for data access during execution. These files correspond to tables in the relational database management system. Rules and Components can read from and write to disk files whose file entity types they are related to. Files can be automatically generated by AppBuilder tools.
To add a File to the hierarchy, see Adding Objects to a Hierarchy. To modify the properties of the object, see Hierarchy Object Properties. To define the relationship of the object to its parent in the hierarchy, see Relationship Properties.

Report

A Report, in conjunction with the Section entity type, defines the paper based output that an application produces for the end user. This application is usually a batch application. You typically define an instance of a Report entity type using the Report Painter tool in the Construction Workbench. For more information about Report Writer, refer to *Reports Guide* .
To add a Report to the hierarchy, see Adding Objects to a Hierarchy. To modify the properties of the object, see Hierarchy Object Properties. To define the relationship of the object to its parent in the hierarchy, see Relationship Properties.

Section

A Section, in conjunction with the Report entity type, defines the paper-based output that an application produces for a user. The application is usually a batch application. Each section includes definition information about a particular part of the report, such as a header section or a footer section. You typically create a report and its associated sections with the Report Painter tool. For more information about Report Writer, refer to the *Reports Guide* .

⚠ The particular part of a report defined by each of its sections is determined by any property of the Contains relationship type, not any property of the section entity type.

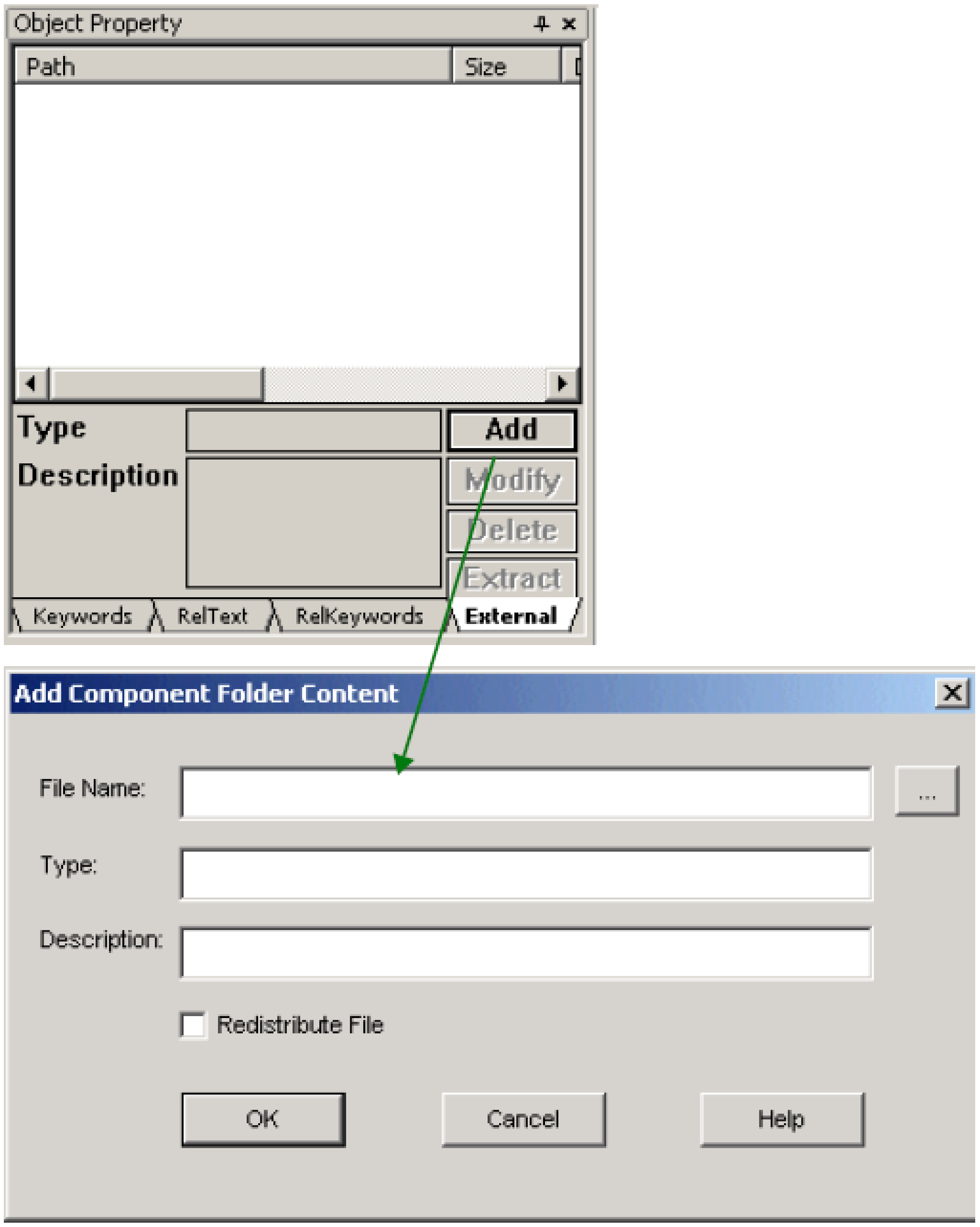
To add a Section to the hierarchy, see Adding Objects to a Hierarchy. To modify the properties of the object, see Hierarchy Object Properties. To define the relationship of the object to its parent in the hierarchy, see Relationship Properties.

Component Folder

A Component Folder is a container for external files.To modify the properties of the object, see Hierarchy Object Properties. To define the relationship of the object to its parent in the hierarchy, see Relationship Properties.
To add external files to an application, add a Component Folder object to a rule and add all the external source code files to that component folder.
To add a Component Folder and external files to the Hierarchy and populate it with external files, complete the following steps:

1. Right-click the component in the Hierarchy window and select **Insert > Component Folder** to add a Component Folder to the Rule in the Hierarchy window.
2. Type the name of the Component Folder in the Name field and click **Apply**.
3. The Object Property window for the Component Folder displays. Type the type of Component Folder in the Folder Type field and a description of the folder in the Description field.
4. Click the **External** tab of the Object Property window. An example is shown in External tab of Component Folder properties.

External tab of Component Folder properties



5. Click **Add** . Click the **...** button to browse for the external files you want to insert and then type the type and description of the component in the Type and Component fields.
6. Click **OK** to close the dialog.

You do not have to remove components and external files and add them again to update components and external files.

Add Component Folder Content dialog

Field	**Description**
File Name	Specifies the path and file name for the external file to be added.
Type	Specifies the file type of the external file to be added.
Description	Describes the external file to be added.
Redistribute File	Specifies whether to include this user component when packaging the application (preparing the calling rule).

You can also sort external files in the Component Folder by Path, Size, Date, Time, Attributes, and Name. To sort external files in the Component Folder, click the column corresponding to the way you want to sort the external files. Your sort can be toggled between ascending order and descending order. To toggle the order of the file sort, click the column again.
You can extract files from the Component Folder for viewing and editing. To extract a file from a Component Folder, complete the following steps.

> If you extract a file from a component folder, it gets the current date/time as an attribute. Extract files with date/time stored in the Component Folder.

1. Display the Object Property window for the Component Folder, if it is not already displayed.
2. Click the **External** tab.

   > You can select only one external file at a time for extraction.

3. Select the component you want to extract and click **Extract**. You are prompted with a dialog asking if you want to overwrite existing files and components.
   The Save As window displays.
4. Click **Save**.

Physical Event

A Physical Event represents a user-defined (business) event. Examples of such an event could be a change in the price of a stock or the withdrawal of money greater than a specified amount. It is attached to a Rule.
To add a Physical Event to the hierarchy, see Adding Objects to a Hierarchy. To modify the properties of the object, see Hierarchy Object Properties. To define the relationship of the object to its parent in the hierarchy, see Relationship Properties.

~~OO Development Objects~~

~~To see the list of all availableOO objects, select the *Repository* tab of Hierarchy window, then select *Insert > OO Development* . The list of OO development objects is displayed in the OO Development submenu.~~ -[AppBuilder OO development objects|AB32:Hierarchy Objects#50606134_80090]- ~~presents them alphabetically as reference.~~

~~AppBuilder OO development objects~~

-[Array	AB32:Hierarchy Objects#50606134_46246]- | -[List	AB32:Hierarchy Objects#50606134_56488]- | -[Native Implements	AB32:Hierarchy Objects#50606134_71037]- | -[Service	AB32:Hierarchy Objects#50606134_31655]- | | -[Class	AB32:Hierarchy Objects#506061 | -[Map

~~Interface~~

~~An Interface contains a map of public data elements and abstract methods. It is generated as child of a class. Developing to interfaces allows easy extensibility by creating new implementations of the interfaces.~~
~~To add an Interface to the hierarchy, see~~ -[Adding Objects to a Hierarchy|AB32:Hierarchy Object Properties#50606134_57286]~~. To modify the properties of the object, see -[Hierarchy Object Properties|AB32:Hierarchy Object Properties#50606134_79193]. To define the relationship of the object to its parent in the hierarchy, see~~ -[Relationship Properties|AB32:Hierarchy Object Properties#50606134_48976]-~~.~~

-

~~Class~~

~~A Class defines a reference type and it also is the compilation unit. It introduces a reference type that contains fields, constructors and methods. The variables of class type may be instantiated.~~
~~Classes support inheritance. Each class can derive from one base class.~~
~~A Class can implement one or more interfaces. It contains maps for public and private data elements and can contain one or more public or private methods, one or more public or private constructor definitions. If no explicit constructor definition is provided, a default constructor implementation will automatically be generated. A class can be defined to be abstract, static or final.~~
~~To add a Class to the hierarchy, see~~ -[Adding Objects to a Hierarchy|AB32:Hierarchy Object Properties#50606134_57286]~~. To modify the properties of the object, see -[Hierarchy Object Properties|AB32:Hierarchy Object Properties#50606134_79193]. To define the relationship of the object to its parent in the hierarchy, see~~ -[Relationship Properties|AB32:Hierarchy Object Properties#50606134_48976]-~~.~~

-

~~Structure~~

~~A Structure contains a map of public data elements and cannot be extended. Structures do not support inheritance. Their role is to store aggregated data and pass it to methods and constructors.~~
~~To add a Structure to the hierarchy, see~~ -[Adding Objects to a Hierarchy|AB32:Hierarchy Object Properties#50606134_57286]~~. To modify the properties of the object, see -[Hierarchy Object Properties|AB32:Hierarchy Object Properties#50606134_79193]. To define the relationship of the object to its parent in the hierarchy, see~~ -[Relationship Properties|AB32:Hierarchy Object Properties#50606134_48976]-~~.~~

-

~~***Exception***~~

~~An Exception signals a violation of the semantic constraints of the OO Rules programming language. It is a generalization of a class. Exceptions can define an error message to be associated with that particular exception type. There are two types of exceptions: exceptions generated by an AppBuilder application, and exceptions generated by the AppBuilder runtime.~~
~~To add an Exception to the hierarchy, see~~ -[Adding Objects to a Hierarchy|AB32:Hierarchy Object Properties#50606134_57286]~~. To modify the properties of the object, see -[Hierarchy Object Properties|AB32:Hierarchy Object Properties#50606134_79193]. To define the relationship of the object to its parent in the hierarchy, see~~ -[Relationship Properties|AB32:Hierarchy Object Properties#50606134_48976]-~~.~~

-

~~***Service***~~

~~A Service is a class construct that implements the business logic. It does not have state, and it only contains methods. Services are remotely deployed, based on AppBuilder partitions.~~
~~Services can be generated and deployed as application components like WebServices, EJBs, RMI, and .NET remote services.~~
~~To add a Service to the hierarchy, see~~ -[Adding Objects to a Hierarchy|AB32:Hierarchy Object Properties#50606134_57286]~~. To modify the properties of the object, see -[Hierarchy Object Properties|AB32:Hierarchy Object Properties#50606134_79193]. To define the relationship of the object to its parent in the hierarchy, see~~ -[Relationship Properties|AB32:Hierarchy Object Properties#50606134_48976]-~~.~~

-

~~***Array***~~

~~An Array is a data structure that contains a number of variables, which are accessed through computed indices. The variables contained in an array, also called the elements of the array, are all of the same type (or can be converted to the same type in case of reference types), and this type is called the element type of the array. There are fixed-size arrays and dynamic arrays.~~
~~Array has an associated size that is an integral number greater than or equal to zero. The array size is not part of the type of the array, but rather are established when an instance of the array type is created at run-time.~~
~~To add an Array to the hierarchy, see~~ -[Adding Objects to a Hierarchy|AB32:Hierarchy Object Properties#50606134_57286]~~. To modify the properties of the object, see -[Hierarchy Object Properties|AB32:Hierarchy Object Properties#50606134_79193]. To define the relationship of the object to its parent in the hierarchy, see~~ -[Relationship Properties|AB32:Hierarchy Object Properties#50606134_48976]-~~.~~

-

~~***Map***~~

~~A Map is a collection that maps keys to values. A map cannot contain duplicate keys; each key can map to at most one value.~~
~~To add a Map to the hierarchy, see~~ -[Adding Objects to a Hierarchy|AB32:Hierarchy Object Properties#50606134_57286]~~. To modify the properties of the object, see -[Hierarchy Object Properties|AB32:Hierarchy Object Properties#50606134_79193]. To define the relationship of the object to its parent in the hierarchy, see~~ -[Relationship Properties|AB32:Hierarchy Object Properties#50606134_48976]-~~.~~

-

~~***List***~~

~~A List is a container for a sequence of objects with fixed order.~~
~~To add a List to the hierarchy, see~~ -[Adding Objects to a Hierarchy|AB32:Hierarchy Object Properties#50606134_57286]~~. To modify the properties of the object, see -[Hierarchy Object Properties|AB32:Hierarchy Object Properties#50606134_79193]. To define the relationship of the object to its parent in the hierarchy, see~~ -[Relationship Properties|AB32:Hierarchy Object Properties#50606134_48976]-~~.~~

-

~~***Set (Container)***~~

~~A Set (Container) is a container for unique elements. Elements can be added, removed, checked for existence and iterated over.~~
~~To add an OO Set to the hierarchy, see~~ -[Adding Objects to a Hierarchy|AB32:Hierarchy Object Properties#50606134_57286]~~. To modify the properties of the object, see -[Hierarchy Object Properties|AB32:Hierarchy Object Properties#50606134_79193]. To define the relationship of the object to its parent in the hierarchy, see~~ -[Relationship Properties|AB32:Hierarchy Object Properties#50606134_48976]-~~.~~

-

~~Typedef~~

~~A Typedef introduces user-defined types.~~
~~To add a Typedef to the hierarchy, see~~ -[Adding Objects to a Hierarchy|AB32:Hierarchy Object Properties#50606134_57286]~~. To modify the properties of the object, see -[Hierarchy Object Properties|AB32:Hierarchy Object Properties#50606134_79193]. To define the relationship of the object to its parent in the hierarchy, see~~ -[Relationship Properties|AB32:Hierarchy Object Properties#50606134_48976]-~~.~~

-

~~Stereotype~~

~~A stereotype is an extension structure that contains zero or more stereotype attributes. By applying a stereotype to a specific AppBuilder object, you can attach the extension structure and assign values to its attributes.~~
~~Stereotype has an attribute "Target Type" that should match the object type it applies to. In order to apply a stereotype to an object, right click the object and choose *Apply Stereotype* from the context menu, then *Query* the repository for the desired stereotype, or create a new one. In the Object Property window, the object type appears in the Target Type field.~~

~~Note:~~	~~When trying to apply the same stereotype to another object, the queried stereotype appears as "not found" in the repository. This is because the requested stereotype is already applied to another object.~~

-

~~Cursor~~

~~A Cursor is a member of a Class, just like a method, used to declare a member variable whose type is a cursor instance. Alternatively, a cursor can be declared as a local variable within a method. If it is defined within a method, the enclosing class must have a USES_CURSOR relationship to the cursor being used in the method. A cursor is not a primitive type. It may have any number of return values. Together the set of return values comprise the result set of a single cursor fetch operation. A cursor also has an SQL body, which is a block of SQL code that defines the cursor query.~~
~~To add a Cursor to the hierarchy, see~~ -[Adding Objects to a Hierarchy|AB32:Hierarchy Object Properties#50606134_57286]~~. To modify the properties of the object, see -[Hierarchy Object Properties|AB32:Hierarchy Object Properties#50606134_79193]. To define the relationship of the object to its parent in the hierarchy, see~~ -[Relationship Properties|AB32:Hierarchy Object Properties#50606134_48976]-~~. For further details on adding and configuring a cursor, please consult the *Developing Applications Guide* .~~

-

~~Rule Wrapper~~

~~A Rule Wrapper allows AppBuilder OO classes to invoke procedural AppBuilder rules. This provides backwards compatibility with procedural AppBuilder and an invocation mechanism to classes by exposing rules as methods and converting views to structures.~~
~~Rule Wrappers have a shared ownership of the Rule objects that they contain, and cannot directly include the rules because rules should be defined elsewhere.~~
~~To add a Rule Wrapper to the hierarchy, see~~ -[Adding Objects to a Hierarchy|AB32:Hierarchy Object Properties#50606134_57286]~~. To modify the properties of the object, see -[Hierarchy Object Properties|AB32:Hierarchy Object Properties#50606134_79193]. To define the relationship of the object to its parent in the hierarchy, see~~ -[Relationship Properties|AB32:Hierarchy Object Properties#50606134_48976]-~~.~~

-

~~Native Class~~

~~A Native Class is the AppBuilder representation of certain language (platform) specific entities. It allows AppBuilder OO applications to access and use entities (classes, interfaces, structs, exceptions, etc.) which are written by third-parties or components from existing applications. Native Classes replace the existing procedural AppBuilder user components.~~
~~To add a Native Class to the hierarchy, see~~ -[Adding Objects to a Hierarchy|AB32:Hierarchy Object Properties#50606134_57286]~~. To modify the properties of the object, see -[Hierarchy Object Properties|AB32:Hierarchy Object Properties#50606134_79193]. To define the relationship of the object to its parent in the hierarchy, see~~ -[Relationship Properties|AB32:Hierarchy Object Properties#50606134_48976]-~~.~~

-

~~Native Exception~~

~~A Native Exception signals a violation of the semantic constraints of the Java programming language.~~
~~To add a Native Exception to the hierarchy, see~~ -[Adding Objects to a Hierarchy|AB32:Hierarchy Object Properties#50606134_57286]~~. To modify the properties of the object, see -[Hierarchy Object Properties|AB32:Hierarchy Object Properties#50606134_79193]. To define the relationship of the object to its parent in the hierarchy, see~~ -[Relationship Properties|AB32:Hierarchy Object Properties#50606134_48976]-~~.~~

-

~~Native Name~~

~~A Native Name is used to construct the names of Java objects.~~
~~To add a Native Name to the hierarchy, see~~ -[Adding Objects to a Hierarchy|AB32:Hierarchy Object Properties#50606134_57286]~~. To modify the properties of the object, see -[Hierarchy Object Properties|AB32:Hierarchy Object Properties#50606134_79193]. To define the relationship of the object to its parent in the hierarchy, see~~ -[Relationship Properties|AB32:Hierarchy Object Properties#50606134_48976]-~~.~~

-

~~Native Implements~~

~~A Native Implements adds a free-form *implements* string to the generated class. The generated class derives from a class implemented in a native language. In Java, for example, the string will be inserted as is into the code after the *implements* declaration. The string will not be validated by AppBuilder to verify that it matches an actual class. The responsibility of this verification lies with the user. If you implement a class which is part of a package, use the fully qualified class name.~~
~~To add a Native Implements to the hierarchy, see~~ -[Adding Objects to a Hierarchy|AB32:Hierarchy Object Properties#50606134_57286]~~. To modify the properties of the object, see -[Hierarchy Object Properties|AB32:Hierarchy Object Properties#50606134_79193]. To define the relationship of the object to its parent in the hierarchy, see~~ -[Relationship Properties|AB32:Hierarchy Object Properties#50606134_48976]-~~.~~

-

~~Native Extends~~

~~A Native Extends adds a free-form *extends* string to the generated class. The generated class derives from a class implemented in a native language. If you extend a class which is a part of a package, use the dot notation to correctly qualify the class.~~
~~To add a Native Extends to the hierarchy, see~~ -[Adding Objects to a Hierarchy|AB32:Hierarchy Object Properties#50606134_57286]~~. To modify the properties of the object, see -[Hierarchy Object Properties|AB32:Hierarchy Object Properties#50606134_79193]. To define the relationship of the object to its parent in the hierarchy, see~~ -[Relationship Properties|AB32:Hierarchy Object Properties#50606134_48976]-~~.~~

-

~~Native Imports~~

~~A Native Imports adds a a free-form *imports* string to the generated class. The generated class imports a class implemented in a native language. If you import the class which is a part of a package, use the dot notation.~~
~~To add a Native Imports to the hierarchy, see~~ -[Adding Objects to a Hierarchy|AB32:Hierarchy Object Properties#50606134_57286]~~. To modify the properties of the object, see -[Hierarchy Object Properties|AB32:Hierarchy Object Properties#50606134_79193]. To define the relationship of the object to its parent in the hierarchy, see~~ -[Relationship Properties|AB32:Hierarchy Object Properties#50606134_48976]-~~.~~

Native File objects

To see the list of all availablenative file objects, select the Repository tab of Hierarchy window, then select *Insert > Native File* . The list of native file objects is displayed in the Native File submenu. AppBuilder Native File objects presents them alphabetically as reference.

AppBuilder Native File objects

Data Field	Data Record	Data Source

Data Source

AData Source stores data file definitions into the repository. Each Data Source contains one or more Data Record entities and can be inserted as a child of a Rule or Class.
To add a Data Source to the hierarchy, see Adding Objects to a Hierarchy. To modify the properties of the object, see Hierarchy Object Properties . To define the relationship of the object to its parent in the hierarchy, see Relationship Properties.

Data Record

AData Record is an entity that describes one or more record formats that can be stored in a file.
To add a Data Record to the hierarchy, see Adding Objects to a Hierarchy. To modify the properties of the object, see Hierarchy Object Properties . To define the relationship of the object to its parent in the hierarchy, see Relationship Properties.

Data Field

AData Field object describes the attributes for each data record. One data record can have one or more data fields and a data field possesses one or more properties.
To add a Data Field to the hierarchy, see Adding Objects to a Hierarchy. To modify the properties of the object, see Hierarchy Object Properties. To define the relationship of the object to its parent in the hierarchy, see Relationship Properties.

Database Diagram Objects

To see the list of all availabledatabase diagram objects, select the Repository tab of Hierarchy window, then select **Insert > Database Diagram**. The list of database diagram objects is displayed in the Database Diagram submenu. AppBuilder database diagram objects presents them alphabetically as reference.

AppBuilder database diagram objects

Column	Key	Table

Table

A Table is a group of information, sorted into columns and stored in a database (a collection of related data). The table name and column names identify the meanings of the values of each row in the table.
To add a Table to the hierarchy, see Adding Objects to a Hierarchy. To modify the properties of the object, see Hierarchy Object Properties. To define the relationship of the object to its parent in the hierarchy, see Relationship Properties.

Key

A Key is, in general, one or more attributes that comprise an entity's identifier. There are three types of keys:

- **Primary** ---A primary key (the only required key) is one or more unique columns that identifies a single instance (a row in a table) of an entity.
- **Foreign** ---A foreign key is a unique index into another table and can be used to join two tables. A foreign key is one or more columns that uniquely identify rows in another table that associates two entities through a relationship.
- **Index** ---An index key is one or more non-unique columns that can locate more than one of an entity's instances.

To add a Key to the hierarchy, see Adding Objects to a Hierarchy. To modify the properties of each object, see Hierarchy Object Properties. To define the relationship of each object to its parent in the hierarchy, see Relationship Properties.

Column

A column identifies the meanings of the values for a row in the table. It contains a number of properties that determine the appearance and behavior of individual columns. The properties of a column effect all cells within that column.
To add a Column to the hierarchy, see Adding Objects to a Hierarchy. To modify the properties of the object, see Hierarchy Object Properties. To define the relationship of the object to its parent in the hierarchy, see Relationship Properties.

Entity Relationship Diagram Objects

To see the list of all available entity relationship diagram objects, select the Repository tab of Hierarchy window, then select **Insert > Entity Relationship Diagram**. The list of entity relationship diagram objects is displayed in the Entity Relationship Diagram submenu. AppBuilder entity relationship diagram objects presents them alphabetically as reference.

AppBuilder entity relationship diagram objects

Attribute	Data Type	Identifier
Business Object	Entity	Relationship

Entity

Entities represent the data used by an enterprise or organization within your application (for example, a customer). Typically, you would define entity types when building your data model. For example, an enterprise that rents automobiles might have a customer entity, a reservation entity, and a rental location entity. There are four types of entities: kernel, associative, characteristic, and intersection.

- **Kernel** — A kernel entity is a basic entity that can exist independently from other entities in a logical model. This does not imply that it cannot have a relationship with other objects; rather, it implies the entity must be a unique object with its own identifier. A kernel entity should not need relationships with other entities to be identified.
- **Associative** ---An associative entity associates two or more kernel or characteristic entities. Associative entities contain non-key properties and can be used to resolve many-to-many relationships.
- **Characteristic** ---A characteristic entity is a weak or dependent entity because it requires the existence of another entity in a logical model. Characteristic entities describe a kernel entity.
- **Intersection** ---An intersection entity associates two or more kernel entities and contains no properties. An intersection entity is the primary way to resolve many-to-many relationships between entities.

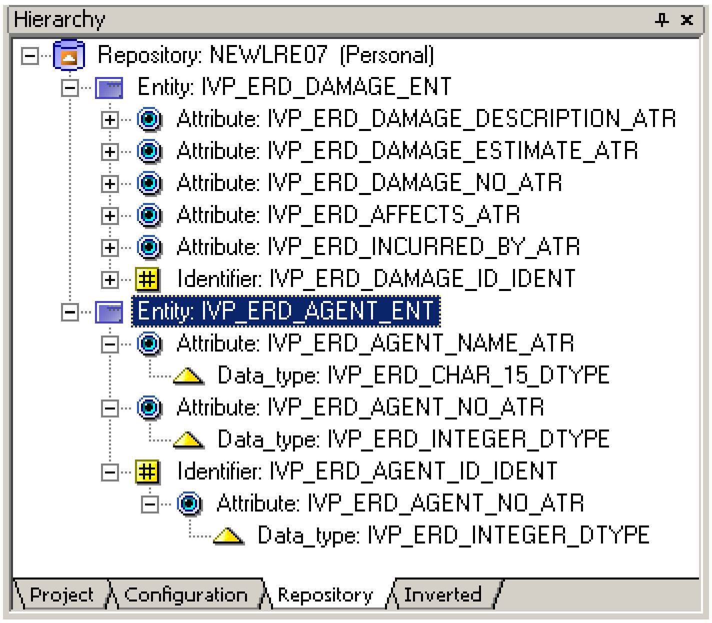
To add an Entity to the hierarchy, see Adding Objects to a Hierarchy. To modify the properties of the object, see Hierarchy Object Properties. To define the relationship of the object to its parent in the hierarchy, see Relationship Properties.

Attribute

An attribute is the smallest unit of information that describes a single characteristic, such as size, values, date, or usage, of an entity or

relationship. An attribute in an attribute hierarchy shown as a child of the entity or relationship it describes.
Attributes define aspects of entities in the Entity Relationship Diagram in the early stages of application design. When you forward engineer or begin the construction, the entity becomes a table, and the attributes become fields in the table. While you use the Entity Relationship Diagram to place the entities and relationships in a diagram, you use the Hierarchy window for adding attributes to an entity.

Hierarchy of entities with attributes


Hierarchy
Repository: NEWLRE07 (Personal)
Entity: IVP_ERD_DAMAGE_ENT
Attribute: IVP_ERD_DAMAGE_DESCRIPTION_ATR
Attribute: IVP_ERD_DAMAGE_ESTIMATE_ATR
Attribute: IVP_ERD_DAMAGE_NO_ATR
Attribute: IVP_ERD_AFFECTS_ATR
Attribute: IVP_ERD_INCURRED_BY_ATR
Identifier: IVP_ERD_DAMAGE_ID_IDENT
Entity: IVP_ERD_AGENT_ENT
Attribute: IVP_ERD_AGENT_NAME_ATR
Data_type: IVP_ERD_CHAR_15_DTYPE
Attribute: IVP_ERD_AGENT_NO_ATR
Data_type: IVP_ERD_INTEGER_DTYPE
Identifier: IVP_ERD_AGENT_ID_IDENT
Attribute: IVP_ERD_AGENT_NO_ATR
Data_type: IVP_ERD_INTEGER_DTYPE

Select an entity in the ERD, right-click and select **Open as Hierarchy**. The **Repository** tab opens in the Hierarchy window and displays the entity. Expand the object in the hierarchy to see its attributes.
To add an Attribute to a hierarchy, see Adding Objects to a Hierarchy. To modify the properties of the object, see Hierarchy Object Properties. To define the relationship of the object to its parent in the hierarchy, see Relationship Properties.

Data Type

Data types identify a physical description of the data (for example, character, date, or decimal). Typically, you should define data types when building your data model.
To add a Data Type to the hierarchy, see Adding Objects to a Hierarchy. To modify the properties of the object, see Hierarchy Object Properties. To define the relationship of the object to its parent in the hierarchy, see Relationship Properties.

Identifier

An Identifier is a logical key that becomes a physical key during database design. An Identifier consists of one or more attributes that uniquely identify an instance of a parent entity or cross-reference another entity.
To add an Identifier to the hierarchy, see Adding Objects to a Hierarchy. To modify the properties of the object, see Hierarchy Object Properties. To define the relationship of the object to its parent in the hierarchy, see Relationship Properties.

Business Object

A Business Object groups a set of related entities together as a data object. A data object conceptually defines the smallest unit of data users need to perform a meaningful business activity and always has a kernel entity as its focus. After the Business Object is created in the Entity Relationship Diagram, it is further defined and refined by creating its associated state models and process dependency diagrams.
To add a Business Object to the hierarchy, see Adding Objects to a Hierarchy. To modify the properties of the object, see Hierarchy Object Properties. To define the relationship of the object to its parent in the hierarchy, see Relationship Properties.

Relationship

A Relationship records information about the relationship type between two or more entities, between an entity and another relationship, or between two or more relationships. Typically, you define instances of Relationship types during business object analysis when you build your data model. For example, if you have two entity instances, Reservation and Car Type, you might have an instance of a relationship called Specifies to describe how they are connected.
To add a Relationship to the hierarchy, see Adding Objects to a Hierarchy. To modify the properties of the object, see Hierarchy Object Properties . To define the relationship of the object to its parent in the hierarchy, see Relationship Properties.

State Transition Diagram Objects

To see the list of all availablestate transition diagram objects, select the Repository tab of Hierarchy window, then select **Insert > State**

Transition Diagram. The list of state transition diagram objects is displayed in the State Transition Diagram submenu. AppBuilder state transition diagram objects presents them alphabetically as reference.

AppBuilder state transition diagram objects

State	Transition

State

A State represents the attribute values and current relationships for a set of data. There are three types of States:

- **Initial** - An initial state represents the condition of a data object before any processing and transitions.
- **Intermediate** - A data object resides in an intermediate state during most of its life cycle and leaves or enters it only when some external or internal event triggers a state change. By default, new states are created as intermediate, however, you can change a state to initial or final.
- **Final** - A final state is a state the data object enters and does not leave.

To add a State to the hierarchy, see Adding Objects to a Hierarchy. To modify the properties of the object, see Hierarchy Object Properties. To define the relationship of the object to its parent in the hierarchy, see Relationship Properties.

Transition

A Transition is a line object that represents a change in data from one state to another.
To add a Transition to the hierarchy, see Adding Objects to a Hierarchy. To modify the properties of the object, see Hierarchy Object Properties. To define the relationship of the object to its parent in the hierarchy, see Relationship Properties.

Process Dependency Diagram Objects

To see the list of all available process dependency diagram objects, select the Repository tab of Hierarchy window, then select **Insert > Process Dependency Diagram**. The list of process dependency diagram objects is displayed in the Process Dependency Diagram submenu. AppBuilder process dependency diagram objects presents them alphabetically as reference.

AppBuilder process dependency diagram objects

Event	Logical Process

Event

An Event represents an incident that acts as a stimulus to a business, system or object. It usually causes some activity or processing to be undertaken and might change the state of objects within the business or system. Events can be classified as external, internal or temporal. You can manipulate Events in the State Transition Diagram and the Process Dependency Diagram.
To add an Event to the hierarchy, see Adding Objects to a Hierarchy. To modify the properties of the object, see Hierarchy Object Properties. To define the relationship of the object to its parent in the hierarchy, see Relationship Properties.

Logical Process

A Logical Process specifies an activity that transforms input data into output data.
To add a Logical Process to the hierarchy, see Adding Objects to a Hierarchy. To modify the properties of the object, see Hierarchy Object Properties. To define the relationship of the object to its parent in the hierarchy, see Relationship Properties.

Configuration Objects

To see the list of all available configuration objects, select the Repository tab of Hierarchy window, then select **Insert > Configuration**. The list of configuration objects is displayed in the Configuration submenu. AppBuilder configuration objects presents them alphabetically as reference.

AppBuilder configuration objects

Application Configuration	Machine	Partition
Database	Package	Server

Application Configuration

An Application Configuration object groups a number of partitions, typically belonging to a single deployment configuration. This object contains the information and other objects needed to prepare a distributed application, to migrate it to a production environment, and to administer the application at runtime. Typically, each project contains a single application configuration.
Application configurations can contain a Partition as a child object.
To add an Application Configuration to the hierarchy, see Adding Objects to a Hierarchy. To modify the object's properties, see Hierarchy Object

Properties. To define the relationship of the object to its parent in the hierarchy, see Relationship Properties.
For more information about how to configure an application, refer to the *Deploying Applications Guide* .

Partition

A Partition defines the associations between a **client** or **server** and its associated machines or databases. Each partition must be associated with an application configuration.

- **Client partitions** – Client partitions contain the project processes to execute on the client-side. When you add a process from the project hierarchy to the configuration, AppBuilder automatically includes all the necessary child objects (for example, rules, views, etc.). In addition, the client partition includes a machine object that indicates the client runtime environment.
- **Server partitions** – Server partitions include a **server** object that defines the type of server (for example, an EJB server). In the server partition, include processes and rules that execute on the server-side.
- **Gateway partitions** – Gateway partitions forward client requests to a server partition running in the same or remote server. The Gateway partition has a machine object and should include an AppBuilder server object with all the associated rules. This server object should be the same as the one defined for the remote server as the gateway partition is forwarding requests for the same frontier rules. These rules are excluded from the rules on the clients. The rules designated as belonging to the server are then "subtracted" from the rules in the process in the client application configuration. A database object (no more than one) can be attached to a partition to specify the database type and the database name.
- **Batch partitions** – Batch partition type and Java language define a Batch partition. A Batch partition has a process as its child and as a container of the Batch Rules. A root rule callable from command-line will be attached to the Process object.

During preparation, use the Project Options window to select the specific configuration and partitions to use.
To add a Partition to the hierarchy, see Adding Objects to a Hierarchy. To modify the properties of the object, see Hierarchy Object Properties. To define the relationship of the object to its parent in the hierarchy, see Relationship Properties.
For more information about a Partition in conjunction with configuring an application, refer to the *Deploying Applications Guide* .

Server

A Server entity type represents a server process in a network configuration.
To add a Server to the hierarchy, see Adding Objects to a Hierarchy. To modify the properties of the object, see Hierarchy Object Properties. To define the relationship of the object to its parent in the hierarchy, see Relationship Properties.
For more information about a Server in conjunction with configuring an application, refer to the *Deploying Applications Guide*.

Machine

A Machine is any machine on a network, such as a developer's or administrator's workstation or a server.
To add a Machine to the hierarchy, see Adding Objects to a Hierarchy. To modify the properties of the object, see Hierarchy Object Properties. To define the relationship of the object to its parent in the hierarchy, see Relationship Properties.
For more information about a Machine in conjunction with configuring an application, refer to the *Deploying Applications Guide*.

Database

A Database represents a database in a network configuration.
To add a Database to the hierarchy, see Adding Objects to a Hierarchy. To modify the properties of the object, see Hierarchy Object Properties. To define the relationship of the object to its parent in the hierarchy, see Relationship Properties.
For more information about a Database in conjunction with configuring an application, refer to the *Deploying Applications Guide*.

Package

A Package groups classes, interfaces, exceptions and structures. Packages should have unique names and be created at the root level in the repository. The repository class definitions are all stored in packages under the root package.
To add a Package to the hierarchy, see Adding Objects to a Hierarchy. To modify the properties of the object, see Hierarchy Object Properties. To define the relationship of the object to its parent in the hierarchy, see Relationship Properties.

Navigating the Hierarchy

The following topics are explained in this section:

- Expanding or Collapsing the Hierarchy
- Moving Objects in the Hierarchy
- Stepping from Tab to Tab

Expanding or Collapsing the Hierarchy

In the Hierarchy window, click the plus symbol (+) in the small box to the left of the object to expand objects to see their child or subordinate objects. When expanded, it displays a minus symbol (-). To collapse the object and hide the child (subordinate) objects, click the minus symbol in the small box to the left of the object. If there is no plus symbol (+), the object has no children in the current scope.
You can also expand an object one of the following ways:

- Select the object and press the plus key (+) and collapse it by pressing the minus key (-). If you press the plus key repeatedly, AppBuilder

advances down to the child object in the hierarchy and expands that object and continues until it reaches the last object in that tree. If you press minus repeatedly, AppBuilder collapses objects and goes up the hierarchy.
- Select an object and press **F8**, which expands to show all the child objects of that object, or **Shift+F8**, which expands the entire hierarchy. Be careful with expanding the entire hierarchy if your hierarchy is very large.
- Select an object and then select **Edit > Expand**, then select the number of levels or **To Leaf** from the Expand submenu.
- Click the **Expand to Leaf** button in the Hierarchy - operations toolbar. Root objects have no parent of the same object type. Leaf objects have no children of the same object type.
- To expand the entire hierarchy, select one of the objects in the hierarchy and press **Alt** and one of the number keys (0, 1, 2...). The number of levels shown corresponds to the number typed. For example, to collapse all the objects in a Project, press **Alt+0**. To expand to the fourth level, which usually shows the child objects of rules, press **Alt+4**.
- You can also set the option to enable double-clicking an object to expand or collapse it in the Hierarchy. Use the **Hierarchy** set of options of the Workbench Options window (select **Tools > Workbench Options** , then click **Hierarchy**) to specify the double-click action for the Hierarchy window. You can set the double-click Action to **Expand/Collapse Object**. Refer to Workbench Options.

Moving Objects in the Hierarchy

To move an object in a hierarchy, click and drag the item from one place in the hierarchy to another. You can drag and drop objects from one part of the hierarchy to another as a way to move or copy the objects. To move an object drag it and drop it to the destination parent object. If the object is not a valid parent object, the pointer remains as a "not" symbol ( ). If it is a valid parent object, the pointer turns into a box pointer ( ). To copy an object, press and hold the **Ctrl** key while you drag and drop the object. The cursor changes to a box pointer with a plus sign ( )indicating that a copy will be made.
Other ways to move an object include the following:

- Select an object and press **Tab**(move right) or **Shift+Tab** (move left).
- Select an object and then select **Edit > Move Right** or **Edit > Move Left** from the Construction Workbench menu.
- Select an object and click the **Move Right** or **Move Left** buttons in the Hierarchy - operations toolbar.
- Right-click an object in the hierarchy, and select **Move Right** or **Move Left** from the pop-up menu.

Moving an object to right creates a relationship in the repository, and moving an object to left deletes a relationship and might create a new relationship in the repository. These relationships are in an uncommitted state until the changes are committed. A relationship between two objects cannot be deleted and recreated without a commit between the two actions. In other words, it is not possible to do multiple move right, move left actions for the same pair of objects, without doing commits between the actions.

Read more about copying and pasting in Copying, Cutting, or Pasting and Copying an Object from the Repository Tab to the Project Tab.

Stepping from Tab to Tab

You can step from one tab to the next by using the **Alt** and arrow keys. For example to go from the **Project** tab to the **Repository** tab, if all four tabs are open, press **Alt** +right arrow key twice.

Searching for Objects and Text

AppBuilder provides a powerful search facility to find objects within the current hierarchy or in the repository. You can also search for a particular string of text in rules or windows. This capability is especially useful when you want to analyze the impact of changing a certain attribute of an object. For example, if you want to change the system ID of a window, you might want to search for the instances of this system ID in all of the child objects under the current display rule.
This section discusses the following topics:

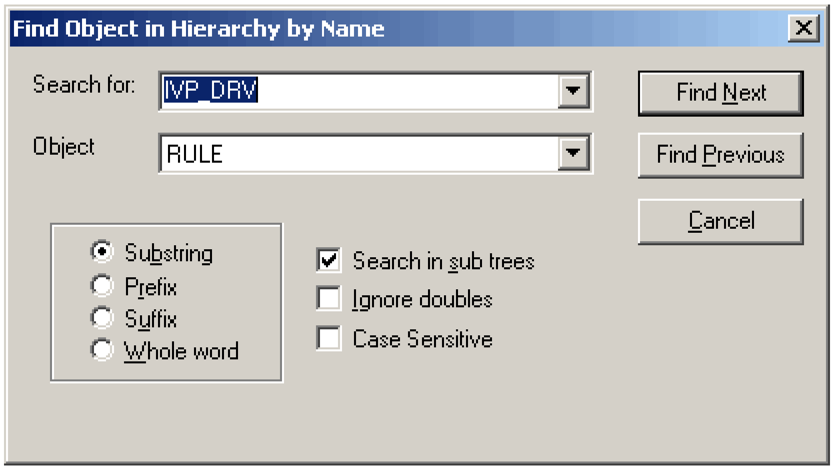
- Searching for Objects in a Hierarchy
- Searching for a String in Rules
- Searching for a String in a Window
- Searching for a String in a Component

Searching for Objects in a Hierarchy

Complete the following steps to search for objects by their names (LONGNAME, not system ID) in the current hierarchy:

1. From the Construction Workbench, select the object in the hierarchy window for which you want to search.
2. From the menu, select **Edit > Find Object** or right-click on the selected object in the Hierarchy window and select **Find Object**. The **Find Object** window displays.

Find Object window



Search for: IVP_DRV

Object: RULE

[✓ Substring] Prefix Suffix Whole word

[✓] Search in sub trees
[ ] Ignore doubles
[ ] Case Sensitive

3. Type all or part of the name of the object you want to find in the *Search for:* field or select a previously entered string from the pull-down list.
4. Select one of the following options:
   - **Substring** - to search for all objects that contain the specified string anywhere in the name
   - **Prefix** - to search for all objects whose names start with the specified string
   - **Suffix** - to search for all objects whose names end with the specified string
   - **Whole word** - to search for all objects whose names are exactly the same as specified string
5. Select the type of the object to search for from the pull-down menu in the **Object** field. You can select **All types** to search for objects whose names match the string you specified in step 3 above regardless of their types.
6. Select one or both of the following options:
   - **Search in sub trees** - to search for objects that match the search criteria even if they are not expanded in the Hierarchy window. Otherwise, the results include only objects that are displayed in the Hierarchy window.
   - **Ignore doubles** - to search only for the first appearance of the object within the hierarchy. Otherwise, the results include every occurrence of the object.
   - **Case Sensitive** - to make a case sensitive search.
7. Click **Find Next** to start the search. The object or objects that match the search criteria are highlighted one by one in the Hierarchy window. If **Search in sub trees** option is selected, the hierarchy expands to show the objects that match the search criteria.
8. Press **F3** to search the next match or **Shift + F3** to find the previous match. Or, from the menu, select **Edit > Find Next** (or **Find Previous**) to search for the next or previous match.
   When you search for the next match, the results include all objects that match the search criteria under the object you selected when you started the search. When you press **Shift + F3** or select **Find Previous** from the menu, the results include all objects that match the search criteria all the way up to the top of the hierarchy regardless of where you originally started the search.
   If no objects in the hierarchy match the search criteria, or no more matching objects are found in **Find Next** or **Find Previous** , a beep sounds, and the selection in the hierarchy remains the same.

You can stop the search anytime by pressing the **Esc** key.

Searching for a String in Rules

Complete the following steps to find a string of text in Rules:

1. To search for text in the Rules within an object, select an object in the hierarchy window.*
   To search for text in the Rules within the repository, begin from the Hierarchy window. It does not matter which object is selected.
2. From the menu, select **Edit > Search in Files**. The **Search in Files** window displays.

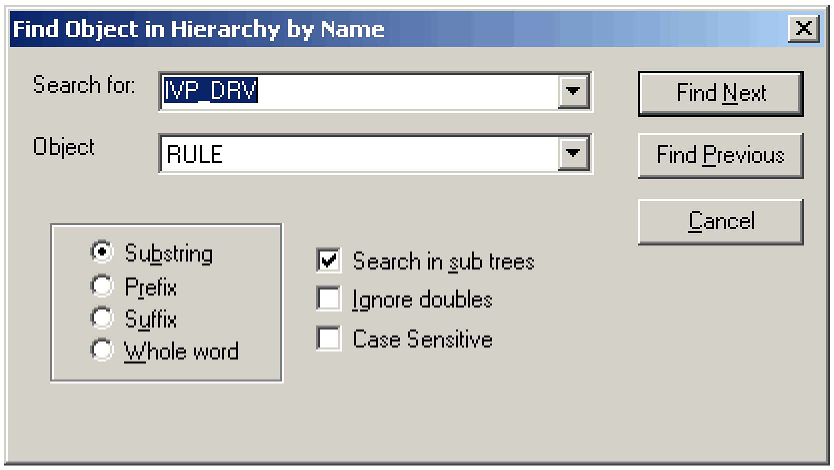
Search in Files dialog example for Rule

3. On the Rule tab, type a string of text to find in *Search for:* field or select one of the previously entered strings from the pull-down list.
4. Specify one or more of the following *Target types* :
   - **Match case** - to search for the text matching upper and lower case, as you typed it in the **Search for** field.
   - **Whole Word** - to search for entire word you typed in the **Search for**: field.
   - **Regular Expression** - to search for the text you entered in the **Search for**: field as a regular expression. For example, \< indicates the text is at the beginning of the word, |> indicates the text is at the end of the word, $ indicates the end of the line, and ^ indicates the beginning of the line. Select one or more of the following Search types:
   - **Search in Repository** - to search for the text in all Rules within the repository.
   - **Search in Hierarchy** - to search for the text in Rules that are in the current hierarchy.
   - **Search in sub trees** - to search for the text in Rules including those that are not currently expanded in the Hierarchy window. Otherwise, the search is performed only for those Rules that are displayed in the Hierarchy window.
5. Select one of the Display options described as follows:
   - **Rule source** - to show found lines in the output window

- **Only display object names** - to show only the names of found entities in the output window. If you select this option, you can copy and paste object names to create a list of found entities.

  The search results are displayed on the **Search** tab in the Output window. Double-click the name of the Rule or one of the found lines to display the Rule that contains the text in the Rule Painter. If no match is found, a message displays in the Output window.

Search result example



Find Object in Hierarchy by Name

Search for: IVP_DRV

Object: RULE

[● Substring] ○ Prefix ○ Suffix ○ Whole word

[✓] Search in sub trees
[ ] Ignore doubles
[ ] Case Sensitive

Press the **Esc** key stop the search at any time.
When you select Regular Expression as the Target Type, the search facility uses the search mechanism FINDSTR provided by Microsoft Windows. Characters used in regular expression shows some of the characters you can use in the regular expression. To find out more about the regular expression and its syntax, click **Start** menu from your desktop, select **Help and Support**, and search for Findstr.

Characters used in regular expression

Character	Value
.	Wildcard: any character
*	Repeat: zero or more occurrences of previous character or class
^	Line position: beginning of line
$	Line position: end of line
[class]	Character class: any one character in set
[^class]	Inverse class: any one character not in set
[x-y]	Range: any characters within the specified range
\x	Escape: literal use of metacharacter x
\<xyz	Word position: beginning of word
xyz\>	Word position: end of word

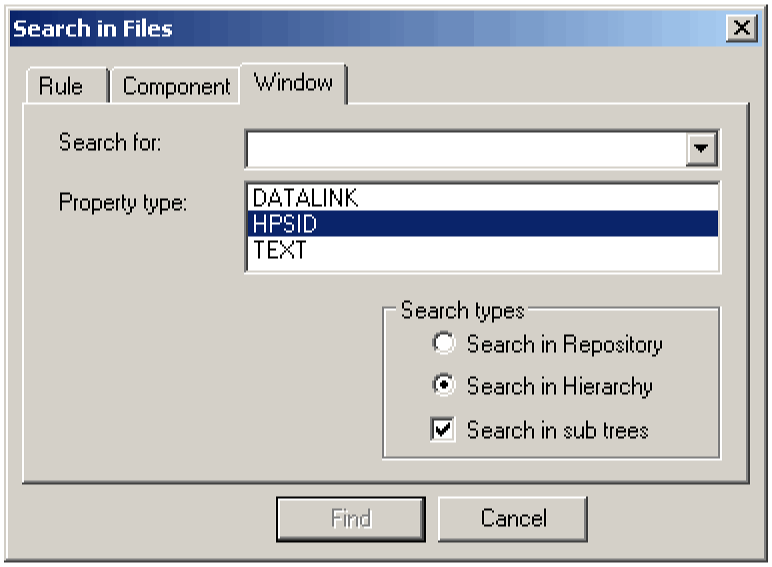
Searching for a String in a Window

Complete the following steps to find a string of text in a Window:

1. If you are searching for a string of text in a Window within the current hierarchy, from the Construction Workbench, select an object in the hierarchy window.

   If you are searching for a text in a Window in the repository, begin within the Hierarchy window. It does not matter which object is selected.
2. From the menu, select **Edit > Search in Files**. The **Search in Files** window displays.

   Search in Files dialog example for Window



3. Select the **Window** tab and type the string of text for which you are searching in the **Search for**: field or select a previously entered string of text from the pull-down list.
4. Select the **Property type** to use as the search criteria from the drop-down list, described as follows:
   - **DATALINK** - to search for objects in the Window that contain the value you entered in the **Property name** field as a DATALINK property
   - **HPSID** - to search for objects in the Window that contain the value you entered in the **Property name** field as HPSID property
   - **TEXT** - to search for objects in the Window that contain the value you entered in the **Property name** field as TEXT property
5. Select a Search type described as follows:
   - **Search in Repository** - to search for the text in all Windows within the repository
   - **Search in Hierarchy**- to search for the text in the Windows in the current hierarchy
   - **Expand sub trees** - to search for the text in the Windows including those that are not currently expanded in the Hierarchy window. Otherwise, the search is performed for only the Windows that are displayed in the Hierarchy window.

The search result displays on the **Search** tab in the Output window. Double-click the name of the Window or on one of the found lines to bring up the window that contains the text in the Window Painter. See Search result example for an example of the search result. If no match was found for the specified search criteria, a message displays in the Output window.

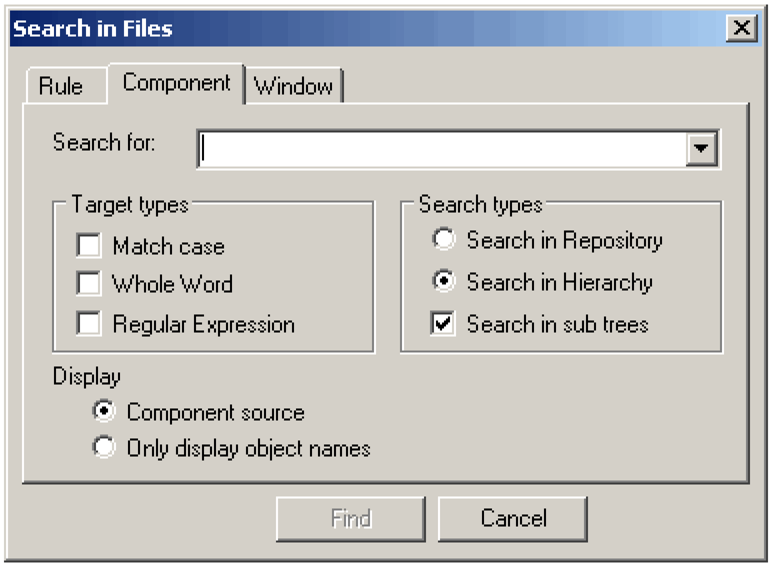
Press the **Esc** key to stop the search anytime.

Searching for a String in a Component

Complete the following steps to find a string of text in a Window:

1. If you are searching for a string of text in a Component within the current hierarchy, from the Construction Workbench, select an object in the hierarchy window.
   If you are searching for a text in a Component in the repository, begin within the Hierarchy window. It does not matter which object is selected.
2. From the menu, select **Edit > Search in Files**. The **Search in Files** window displays.

Search in Files dialog example for Component



Search for:

Target types: [ ] Match case, [ ] Whole Word, [ ] Regular Expression

Search types: ( ) Search in Repository, [✓ Search in Hierarchy], [✓ Search in sub trees]

Display: [✓ Component source], ( ) Only display object names

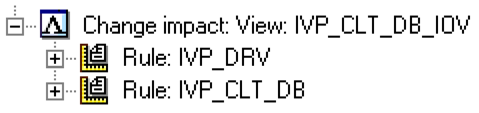
3. Select the **Component** tab and type the string of text for which you are searching in the **Search for**: field or select a previously entered string text from the pull-down list.
4. Click the check box to select the **Target types** to use as the search criteria, described as follows:
   - **Match case** - The search string must match the case of the search string that you type in the **Search for**: field.
   - **Whole Word** - The search string must be a whole word, as typed in the **Search for**: field.
   - **Regular Expression** - to search for objects in the Window that contain the value you entered in the **Property name** field as TEXT property.
5. Select a Search type described as follows:
   - **Search in Repository** - to search for the text in all Windows within the repository.
   - **Search in Hierarchy** - to search for the text in the Windows in the current hierarchy.
   - **Search in sub trees** - to search for the text in the Components including those that are not currently expanded in the Hierarchy window. Otherwise, the search is performed only for the Components that are displayed in the Hierarchy window.

The search result displays on the **Search** tab in the Output window. Double-click the name of the Window or one of the found lines to bring up the window that contains the text in the Window Painter. See Search result example for an example of the search result. If no match is found for the specified search criteria, a message displays in the Output window.
Press the **Esc** key to stop the search anytime.

Showing Impact of Changes

During the application development and maintenance, it is important to know the impact of the changes you make to an object. For example, if you change a CHAR 5 field into a CHAR 10 field, this requires all rules, components, and windows that have this field in their data universe to be prepared.
To show which objects need to be prepared again by the changes you make to an object, right click an object in the Configuration tab or the Repository tab and select **Show Impact** from the pop-up menu. The result is displayed in the Repository tab as shown in Show Impact example:

Show Impact example



Show Impact feature is based on the preparation analysis and not on the rebuild analysis.

Tabs in the Hierarchy Window

There are four tabs available in the Hierarchy window, each with its own purpose. Project tab and Configuration tab are not available until you create or open a project.

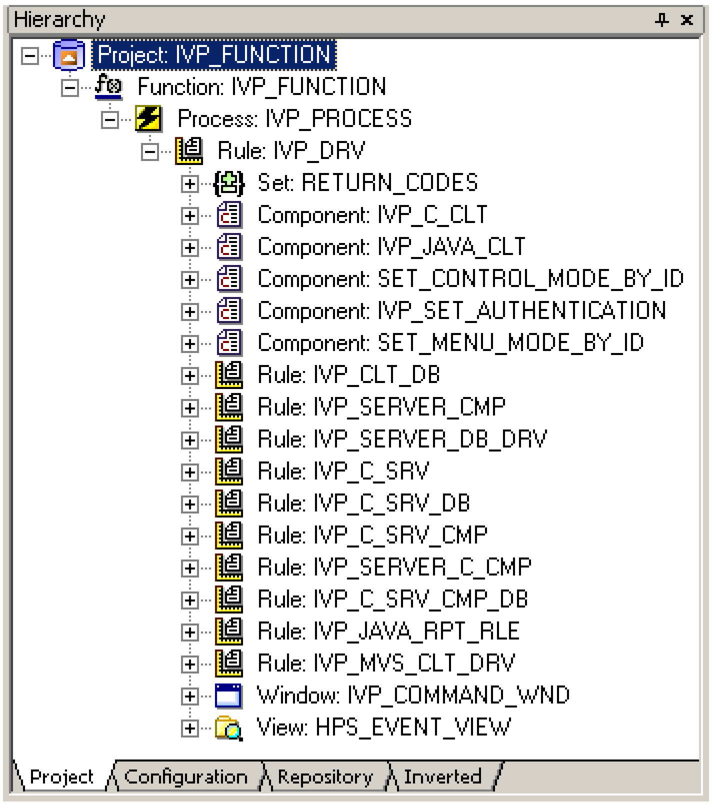
- Project Tab - construct application hierarchies of function, process, and rules and create new objects
- Configuration Tab - partition an application and build a configuration hierarchy

- Repository Tab - hierarchal view of the repository
- Inverted Tab - display any parents of an object

Project Tab

Use the **Project** tab of the Hierarchy window to view and create the structure or part of the structure of an application. This structure, the project hierarchy, specifies how the application executes. The **Project** tab is displayed only when a project is open.

Hierarchy window Project tab


Hierarchy
- Project: IVP_FUNCTION
  - Function: IVP_FUNCTION
    - Process: IVP_PROCESS
      - Rule: IVP_DRV
        - Set: RETURN_CODES
        - Component: IVP_C_CLT
        - Component: IVP_JAVA_CLT
        - Component: SET_CONTROL_MODE_BY_ID
        - Component: IVP_SET_AUTHENTICATION
        - Component: SET_MENU_MODE_BY_ID
        - Rule: IVP_CLT_DB
        - Rule: IVP_SERVER_CMP
        - Rule: IVP_SERVER_DB_DRV
        - Rule: IVP_C_SRV
        - Rule: IVP_C_SRV_DB
        - Rule: IVP_C_SRV_CMP
        - Rule: IVP_SERVER_C_CMP
        - Rule: IVP_C_SRV_CMP_DB
        - Rule: IVP_JAVA_RPT_RLE
        - Rule: IVP_MVS_CLT_DRV
        - Window: IVP_COMMAND_WND
        - View: HPS_EVENT_VIEW

* *

Use the **Project** tab to construct application hierarchies of function, process, and rules and create new objects. Use the buttons of project objects in the Hierarchy Objects toolbar to create new objects in the project hierarchy. You can also add existing objects from the repository to your project.
Use the display of the hierarchy to understand the contents of the project hierarchy:

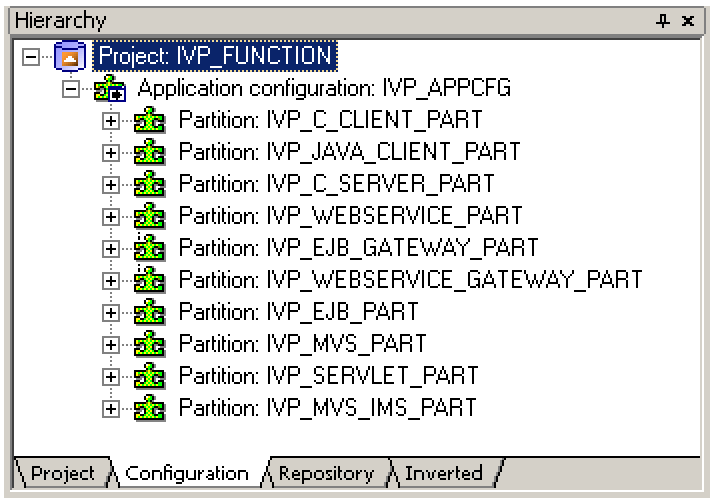
- See the child relationships of objects to a parent object.
- Find the root rule which is the highest level rule.
- Verify that a project is complete and that objects have their required components.

Refer to Expanding or Collapsing the Hierarchy for instructions to expand and collapse the objects in the hierarchy.

Configuration Tab

The **Configuration** tab is used to partition an application and build a configuration hierarchy. The configuration hierarchy associates client processes and logical servers with databases and physical machines, creating an application configuration. Each configuration corresponds to a specific project. You can create multiple configurations for the same project. Refer to Hierarchy window with Configuration tab for an example of the configuration hierarchy which shows possible configurations and deployment scenarios.
Refer to the *Deploying Applications Guide* for details about how to configure an application.

Hierarchy window with Configuration tab



Hierarchy

Project: IVP_FUNCTION
Application configuration: IVP_APPCFG
Partition: IVP_C_CLIENT_PART
Partition: IVP_JAVA_CLIENT_PART
Partition: IVP_C_SERVER_PART
Partition: IVP_WEBSERVICE_PART
Partition: IVP_EJB_GATEWAY_PART
Partition: IVP_WEBSERVICE_GATEWAY_PART
Partition: IVP_EJB_PART
Partition: IVP_MVS_PART
Partition: IVP_SERVLET_PART
Partition: IVP_MVS_IMS_PART

Use the **Configuration** tab to partition your application, which includes:

- Specifying the machines to which to prepare the parts of the application
- Specifying the execution environment
- Specifying the database parameters (type, database name) if your application uses a database.

Use the buttons of configuration objects in the Hierarchy Objects toolbar to create new objects in the configuration hierarchy. You can also add existing objects from the repository to your project.
Use the display of the hierarchy to understand the contents of the configuration hierarchy:

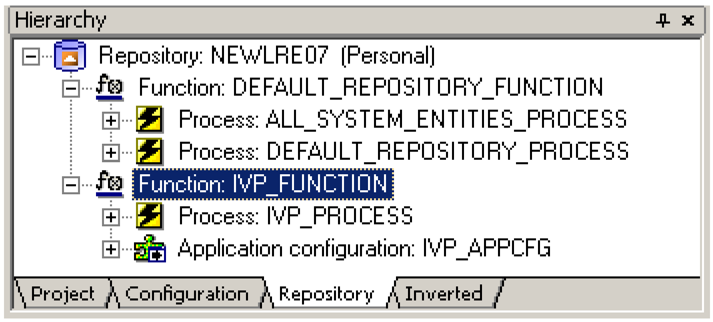
- See the multiple partitions of an application configuration
- See the machine associated with each partition
- See which rules run on which partitions (on which machines).

Refer to Expanding or Collapsing the Hierarchy for instructions to expand and collapse the objects in the hierarchy. The hierarchy in the **Configuration** tab shows the content of a partition (not just the frontier rules).

Repository Tab

The **Repositor** tab is a hierarchal view of the repository. Here you can browse or modify repository objects that do not belong to the current project. Use the **Repository** tab to add commonly used objects from the repository to your project. Use the **Repository** tab to query and browse existing objects in the repository, copy objects from the repository to the project hierarchy, and display an object's child objects. You can copy and paste from any tab to any tab. See Copying an Object from the Repository Tab to the Project Tab.
To display an object's parent in the **Inverted** Tab from the **Repository** tab, refer to Displaying an Object's Parent for more information.
To add an object to the repository, select the name of the repository in the **Repository** tab and then select **Insert > [Menu]** from the **Construction Workbench** menu and select the item you want to insert from the corresponding [*Menu*]. Refer to the following figure for an example of a Repository view.

Hierarchy window with Repository tab



Hierarchy

Repository: NEWLRE07 (Personal)
Function: DEFAULT_REPOSITORY_FUNCTION
Process: ALL_SYSTEM_ENTITIES_PROCESS
Process: DEFAULT_REPOSITORY_PROCESS
Function: IVP_FUNCTION
Process: IVP_PROCESS
Application configuration: IVP_APPCFG

You can use the icons on the **Hierarchy - objects** toolbar on **Repository** tab to add objects to the repository hierarchy. You can also add existing objects from the repository to your project.

Displaying an Object's Parent

To display an object's parent, right-click an object in the **Repository** tab and select **Show Parents** from the pop-up menu (or **View > Parent**). The system displays the object's parents in the **Inverted** tab. You can select more than one parent object, and all the children are displayed and highlighted in the Inverted Tab.

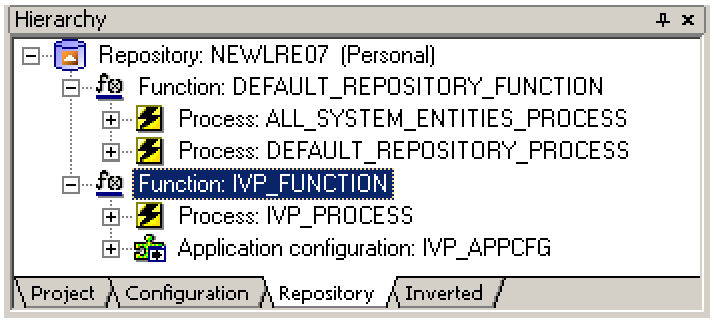
Copying an Object from the Repository Tab to the Project Tab

If you added objects to the hierarchy in the **Repository** tab, you can copy an object from this tab and paste it into your hierarchy in the **Project** tab. Select the object in the **Repository** tab and select **Edit > Copy** or right-click **Copy** or **Ctrl+C** . Then in the **Project** tab, with the parent object selected, the destination for the object to be pasted, select **Edit > Paste** or right-click **Paste** or **Ctrl+V**.

Inverted Tab

To display any parents of an object, all the objects that have that object as a child, click the **Inverted** tab in the Hierarchy window. This shows an inverted hierarchy, with the child object above the parent objects in the hierarchy.
Refer to Expanding or Collapsing the Hierarchy for instructions on expanding and collapsing the objects in the hierarchy. This hierarchy works in reverse, displaying parents when the hierarchy is expanded.

Hierarchy window with Inverted tab


Hierarchy

Repository: NEWLRE07 (Personal)
Function: DEFAULT_REPOSITORY_FUNCTION
Process: ALL_SYSTEM_ENTITIES_PROCESS
Process: DEFAULT_REPOSITORY_PROCESS
Function: IVP_FUNCTION
Process: IVP_PROCESS
Application configuration: IVP_APPCFG

Use the **Inverted** tab to:

- Browse existing objects in the repository in an inverted fashion.
- Display an object's parent objects.

While you can copy and paste from this hierarchy, it is more typical to use the Repository Tab to perform those actions.

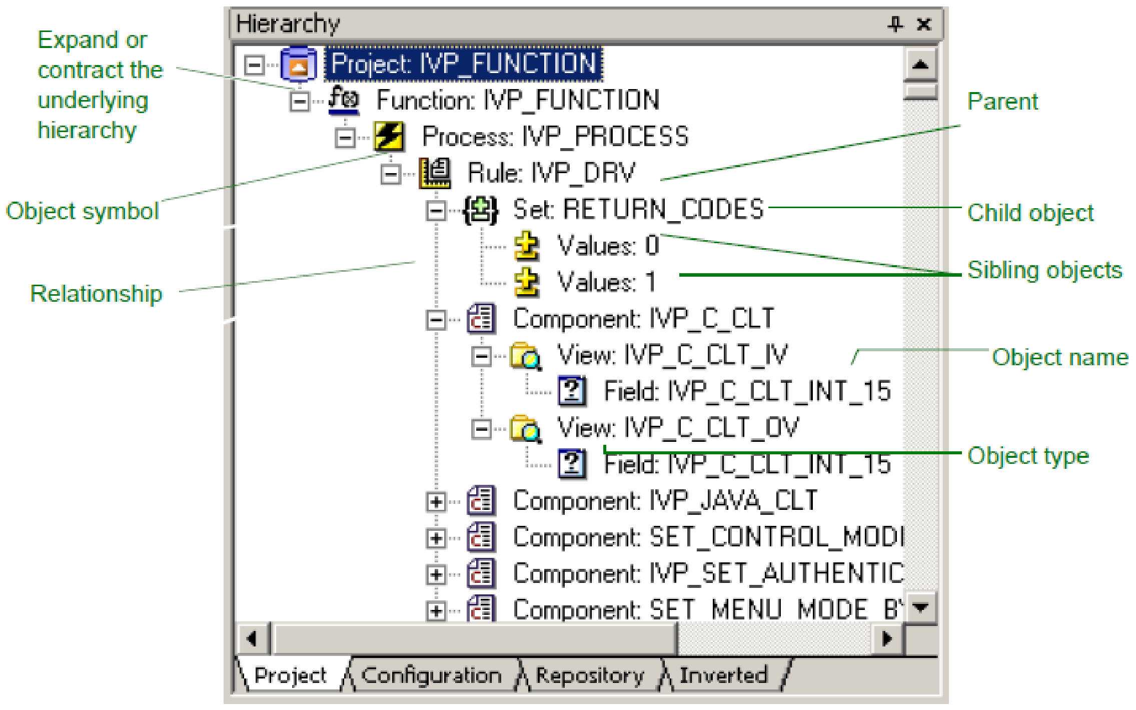
Displaying an Object's Children

To display an object's children, right-click an object in the **Inverted** tab and select **Show Children** from the pop-up menu (or **View > Children**). The selected object is added in the **Repository** tab if not already present and it becomes the selected object. The system displays the object's children in the **Repository** tab. You can select more than one child object and all the parents are displayed and highlighted in the Repository Tab.

Understanding Parts of Hierarchy

Sample hierarchy illustrates the different parts of the Hierarchy window.

Sample hierarchy



```
Project: IVP_FUNCTION
  Function: IVP_FUNCTION
    Process: IVP_PROCESS
      Rule: IVP_DRV
        Set: RETURN_CODES
          Values: 0
          Values: 1
        Component: IVP_C_CLT
          View: IVP_C_CLT_IV
            Field: IVP_C_CLT_INT_15
          View: IVP_C_CLT_OV
            Field: IVP_C_CLT_INT_15
        Component: IVP_JAVA_CLT
        Component: SET_CONTROL_MODI
        Component: IVP_SET_AUTHENTIC
        Component: SET_MENU_MODE_B
```

The - indicates an exploded view displaying all children. The + indicates a collapsed view hiding all children.

When you open a project, the window contains the following tabs:

- Project Tab - The functional hierarchy of the application as a project
- Configuration Tab - The configuration and partitioning hierarchy for preparation and deployment of that application
- Repository Tab - The hierarchy and relationship of objects within the repository
- Inverted Tab - An inverted hierarchy (showing ownership) of objects within the repository

The Project and Configuration tabs are available only when a project is open. In the Repository and Inverted tabs, you can use the Hierarchy Objects toolbar to add objects to the hierarchy in that tab.

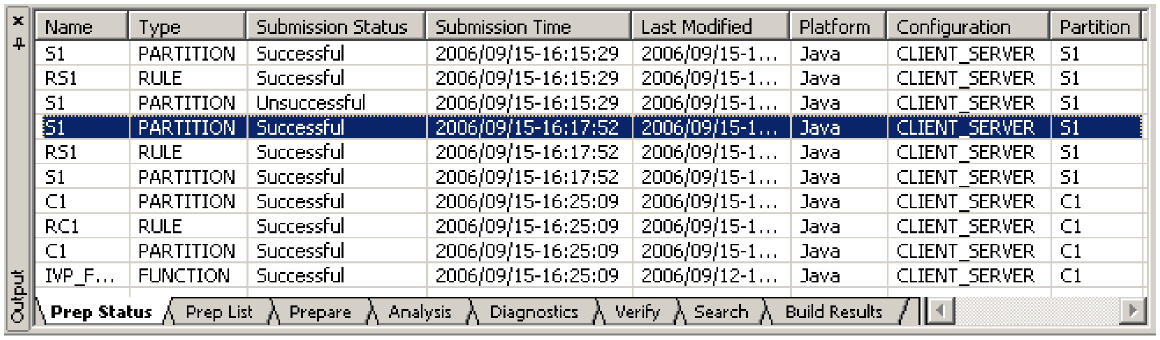
You can change the way objects are displayed in the Hierarchy window. Use the **Hierarchy** set of options of the Workbench Options window to specify default display options and double-click actions for the Hierarchy window. From the Construction Workbench menu bar, select **Tools > Workbench Options** , then click on **Hierarchy** . Read more about Workbench Options in Workbench Options.

You can either dock or float the Hierarchy window. In the gray area around the Hierarchy window, right-click and select **Allow Docking** , or **Hide** . Read more about Docking and Floating Windows in Introduction to Construction Workbench.

Output Window

For several operations in the Construction Workbench, from any of a number of tools, the status or outcome is displayed in the various tabs of the Output window. By default, the Output window is displayed at the bottom of the Construction Workbench. Sample Output window presents a sample output window.

Sample Output window



Name	Type	Submission Status	Submission Time	Last Modified	Platform	Configuration	Partition
S1	PARTITION	Successful	2006/09/15-16:15:29	2006/09/15-1...	Java	CLIENT_SERVER	S1
RS1	RULE	Successful	2006/09/15-16:15:29	2006/09/15-1...	Java	CLIENT_SERVER	S1
S1	PARTITION	Unsuccessful	2006/09/15-16:15:29	2006/09/15-1...	Java	CLIENT_SERVER	S1
S1	PARTITION	Successful	2006/09/15-16:17:52	2006/09/15-1...	Java	CLIENT_SERVER	S1
RS1	RULE	Successful	2006/09/15-16:17:52	2006/09/15-1...	Java	CLIENT_SERVER	S1
S1	PARTITION	Successful	2006/09/15-16:17:52	2006/09/15-1...	Java	CLIENT_SERVER	S1
C1	PARTITION	Successful	2006/09/15-16:25:09	2006/09/15-1...	Java	CLIENT_SERVER	C1
RC1	RULE	Successful	2006/09/15-16:25:09	2006/09/15-1...	Java	CLIENT_SERVER	C1
C1	PARTITION	Successful	2006/09/15-16:25:09	2006/09/15-1...	Java	CLIENT_SERVER	C1
IVP_F...	FUNCTION	Successful	2006/09/15-16:25:09	2006/09/12-1...	Java	CLIENT_SERVER	C1

The Output window contains the following tabs:

- Prep Status Tab
- Prep List Tab
- Prepare Tab
- Analysis Tab
- Diagnostics Tab
- Verify Tab
- Search Tab
- Build Results Tab
- Rebuild Report Tab

You can resize it and move it to any location in the Construction Workbench Work Area or undock it as a separate window. To move the Output window outside of the Construction Workbench, right-click in the Output window and select **Allow Docking** to uncheck the option. You can then move the window on your desktop. For more information, refer to Docking and Floating Windows.

To show or hide the Output window, select **View > Output** in the Construction Workbench.

Prep Status Tab

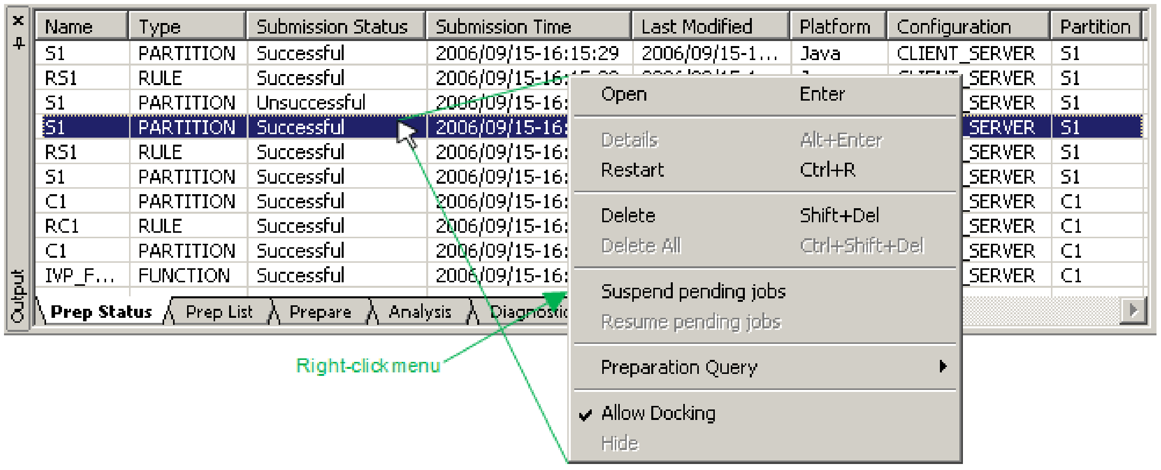
The **Prep Status** tab displays the status and result of the preparation of an object or objects. An example is shown in Sample Prep Status tab. The **Prep Status** tab in the Output window shows the information about preparation jobs (from Prepare or Super Prepare actions). This tab shows the current state of the operation of preparing, the real-time status and result of each object or objects being prepared. The various types of information are summarized in Preparation Status headings. Refer to Preparation Status descriptions for a description of the different status of the preparation being performed.

For more information about preparation, refer to the *Deploying Applications Guide*.

Within the **Prep Status** tab, there are several actions that can be performed:

- Displaying Preparation Details
- Restarting Preparation Jobs
- Suspending and Resuming Pending Jobs
- Opening or Viewing Preparable Objects
- Loading, Saving, and Submitting a Prep List

Sample Prep Status tab



Name	Type	Submission Status	Submission Time	Last Modified	Platform	Configuration	Partition
S1	PARTITION	Successful	2006/09/15-16:15:29	2006/09/15-1...	Java	CLIENT_SERVER	S1
RS1	RULE	Successful	2006/09/15-16:			_SERVER	S1
S1	PARTITION	Unsuccessful	2006/09/15-16:			_SERVER	S1
S1	PARTITION	Successful	2006/09/15-16:			_SERVER	S1
RS1	RULE	Successful	2006/09/15-16:			_SERVER	S1
S1	PARTITION	Successful	2006/09/15-16:			_SERVER	S1
C1	PARTITION	Successful	2006/09/15-16:			_SERVER	C1
RC1	RULE	Successful	2006/09/15-16:			_SERVER	C1
C1	PARTITION	Successful	2006/09/15-16:			_SERVER	C1
IVP_F...	FUNCTION	Successful	2006/09/15-16:			_SERVER	C1

Preparation Status headings

Heading	Description
Name	This is the unique name of the object being prepared as it is displayed in the hierarchy.
Type	This is the type of object being prepared, whether rule, component, window, set, etc.
Submission Status	This is the status of the preparation job, as summarized in Preparation Status descriptions.
Submission Time	This is the date and time when the object was submitted for preparation.
Last Modified	This is the date and time that the object was last updated.
Platform	This is the target platform, either Java or Windows, MVS, Unix, etc.

Configuration	This is the configuration to which the partition belongs (for a distributed application).
Partition	This is the partition to which the object belongs (for a distributed application).

Preparation Status descriptions

Status	Description of preparation job
Pending	Queued by AppBuilder but are not yet being prepared.
Starting	Begun preparing and waiting for previously submitted jobs to finish.
Executing	Currently being prepared.
Remote submitted	Submitted to a mainframe.
Waiting	Completed processing on the local workstation and are awaiting processing on the remote workstation.
Successful	Finished properly. (Might not display if you have the Workbench Option set to not display successful jobs.
Unsuccessful	Not finished properly. See the error messages for that job.

To clear or delete the records in this display, select the line and right-click. Select the **Delete** or **Delete All** option. If the selected job is Pending, deleting the job cancels it and removes it from the queue. If the selected job is Starting or Executing, the choice is grayed out – you cannot delete it. If the selected job is Remote submitted or Waiting, deleting removes it from the Output window. If the job is Successful, deleting does not remove executable files.

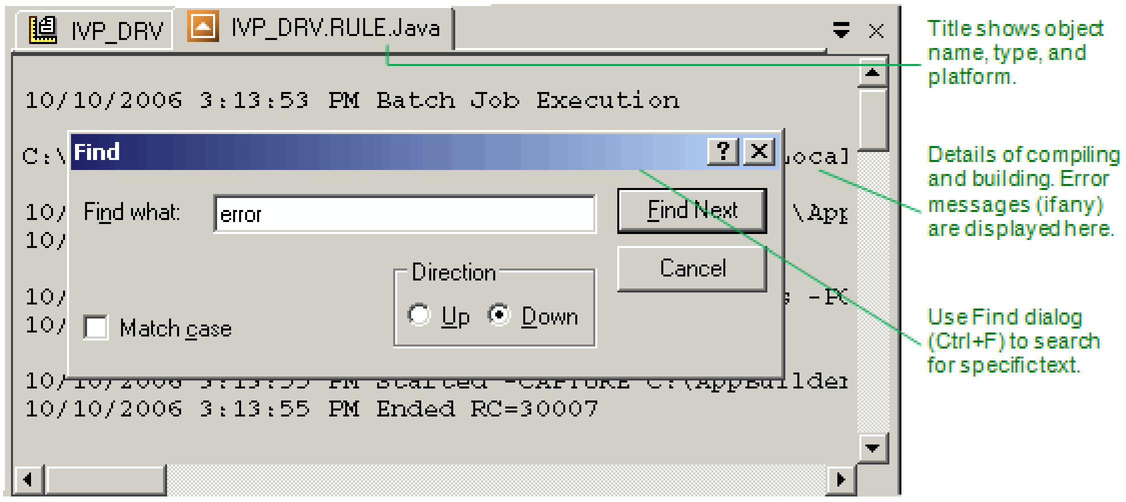
You can click any of the headings in the **Prep Status** tab to arrange the displayed items in order of that heading. For example, to see all the successfully prepared objects, click the **Status** heading and the successful items appear together.

Displaying Preparation Details

For details about any item in the *Prep Status* tab, view the Details window. To see details, right-click the line in the **Prep Status** tab and select **Details**. The system displays a detailed report for the object as a tab in the Work Area. Double-clicking an executing, waiting, or completed preparation job also displays the Details tab. The title of the tab contains the name of the object, the type of object, and the platform separated by periods.

You can also select a job and press **Alt+Enter** to display job details.

Sample Details tab example


IVP_DRV
IVP_DRV.RULE.Java
Find
Find what:
error
Find Next
Cancel
Direction
Up
Down
Match case
Title shows object name, type, and platform.
Details of compiling and building. Error messages (if any) are displayed here.
Use Find dialog (Ctrl+F) to search for specific text.

```
10/10/2006 3:13:53 PM Batch Job Execution
C:\                                     ocal
10/                                     \App
10/
10/                                     -PC
10/
10/10/2006 3:13:55 PM Started -CAPTURE C:\AppBuilder
10/10/2006 3:13:55 PM Ended RC=30007
```

To perform a text search inside a details window, to find a text string, click in the Details window and press **Ctrl+F**, which brings up the Find window. You can search the window for any text.

Restarting Preparation Jobs

To restart a preparation job listed in the **Prep Status** tab:

1. In the Output window, click the **Prep Status** tab.
2. Select the objects to prepare.

To select multiple jobs, press **Ctrl** while selecting the job(s). To select a group of consecutive jobs, select the first and press **Shift** when selecting the last.

3. Right-click the jobs and select **Restart** from the pop-up menu.
   You can also restart the job by pressing **Ctrl+R**.

You can also prepare an object again by doing a query. The Preparation Query window is available from the **Prep Status** tab by right-clicking and selecting **Preparation Query**. When you have selected the object to prepare, click **Prepare**.

For more information about preparation in general, refer to the *Deploying Applications Guide*.

Suspending and Resuming Pending Jobs

To suspend a pending preparation job, right-click in the **Prep Status** tab and select **Suspend Pending Jobs** from the pop-up menu.

To resume a pending job, right-click in the **Prep Status** tab and select **Resume Pending Jobs** from the pop-up menu.

Opening or Viewing Preparable Objects

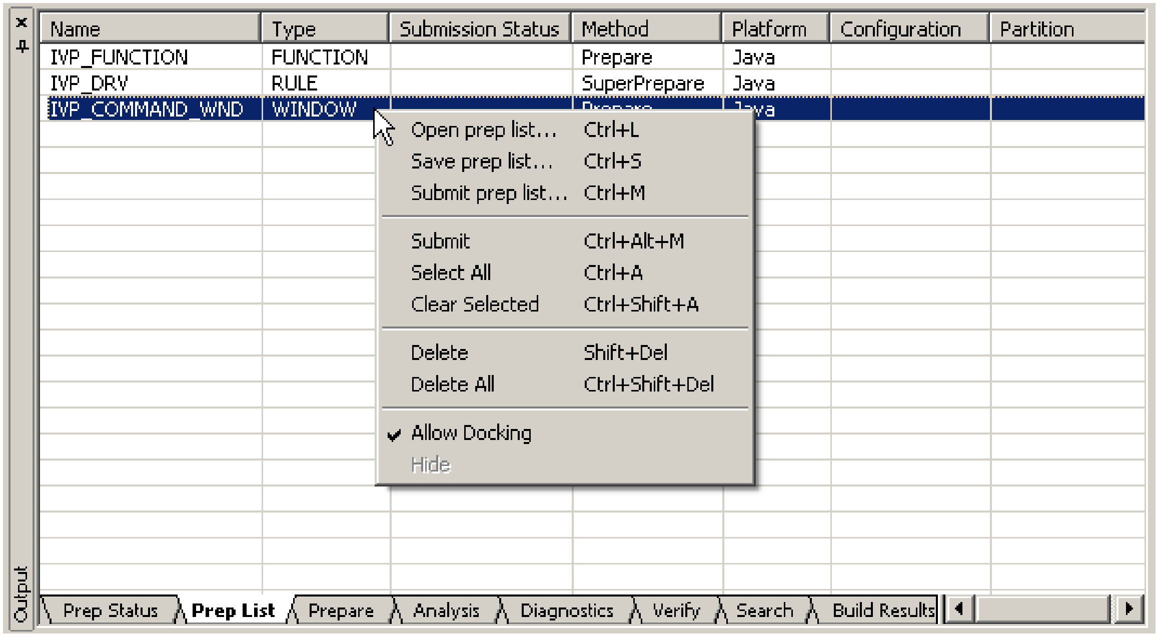
From the Prep Status tab you can open the object (a rule in Rule Painter, for example). Select the line of the object in the **Prep Status** tab, right-click and select **Open**. You can also drag objects from the **Prep Status** tab and drop them in Work Area to open them. For more information about the drag and drop feature, refer to Dragging and Dropping Preparable Objects.

Prep List Tab

The Prep List tab is essentially a history of the preparations performed. It displays the jobs that were previously prepared. This list can be saved to a file and retrieved from a file. An example is shown in Sample Prep List tab. You can perform three actions within this tab:

- Loading, Saving, and Submitting a Prep List
- Submitting Preparation Jobs
- Dragging and Dropping Preparable Objects

Sample Prep List tab



Name	Type	Submission Status	Method	Platform	Configuration	Partition
IVP_FUNCTION	FUNCTION		Prepare	Java		
IVP_DRV	RULE		SuperPrepare	Java		
IVP_COMMAND_WND	WINDOW		Prepare	Java		

From wherever the preparation is started, the job is shown in this list. If the preparation is started from the **Prep List** tab with a Submit command, it is not listed again, but the Submission Status changes to **Submitted**. If the preparation is started from the **Prep Status** tab with the Restart command, the job is not listed.

For more information about preparation in general, refer to the *Deploying Applications Guide*.

Loading, Saving, and Submitting a Prep List

You can save the list of preparation jobs to a file and then at a later time, or on a different machine, load that list of preparation jobs and run them for preparation.

To save a preparation list, complete the following steps:

1. In the **Prep List** tab, right-click and select **Save prep list** from the pop-up menu. The Save As window is displayed.
   You can also save the jobs by pressing **Ctrl+S**.
2. Select a filename and location for the preparation list and click **Save**.
   By default, AppBuilder saves the preparation list as pwbatch.dat in the <*AppBuilder*>\AD\CFG\DATA directory.

To load an existing preparation list, complete the following steps:

1. Right-click in the **Prep List** tab and select **Open prep list** from the pop-up menu. The Open window is displayed.
2. Select the preparation list (DAT file) to load and click **Open**. AppBuilder adds the preparation jobs to the **Prep List** tab. If the item is already there it replaces it, but it does not clear the tab or remove other jobs from the list.
3. To submit the preparation jobs, select the jobs, right-click and select **Submit** from the pop-up menu.

   > To select multiple jobs, press **Ctrl** while selecting the job(s). To select a group of consecutive jobs, select the first and press **Shift** when selecting the last.

   You can also submit jobs by selecting the jobs and pressing **Ctrl+Alt+M**.

To submit a preparation list, complete the following steps:

1. In the **Prep List** tab, right-click and select **Submit prep list** from the pop-up menu. The Open window is displayed.
2. Select the preparation list (DAT file) to load and click **Open**. AppBuilder adds the preparation jobs to the **Prep List** tab. If the items are already there it replaces them, but it does not clear the tab or remove other jobs from the list. The Submission Status changes to **Submitted**.

Submitting Preparation Jobs

You can resubmit any or all of the of the jobs in the list. Select the jobs to resubmit and right-click. Select **Submit**.

You can select jobs by clicking on them.

> To select multiple jobs, press **Ctrl** while selecting the jobs. To select a group of consecutive jobs, select the first and press **Shift** when selecting the last.

To unselect a job, click off the list but in the window or right-click and select **Clear Selected**.

Dragging and Dropping Preparable Objects

You can drag and drop preparable objects from the application hierarchy in the Hierarchy window to the **Prep List** tab to start preparation. For more information about the drag and drop feature, refer to Dragging and Dropping.

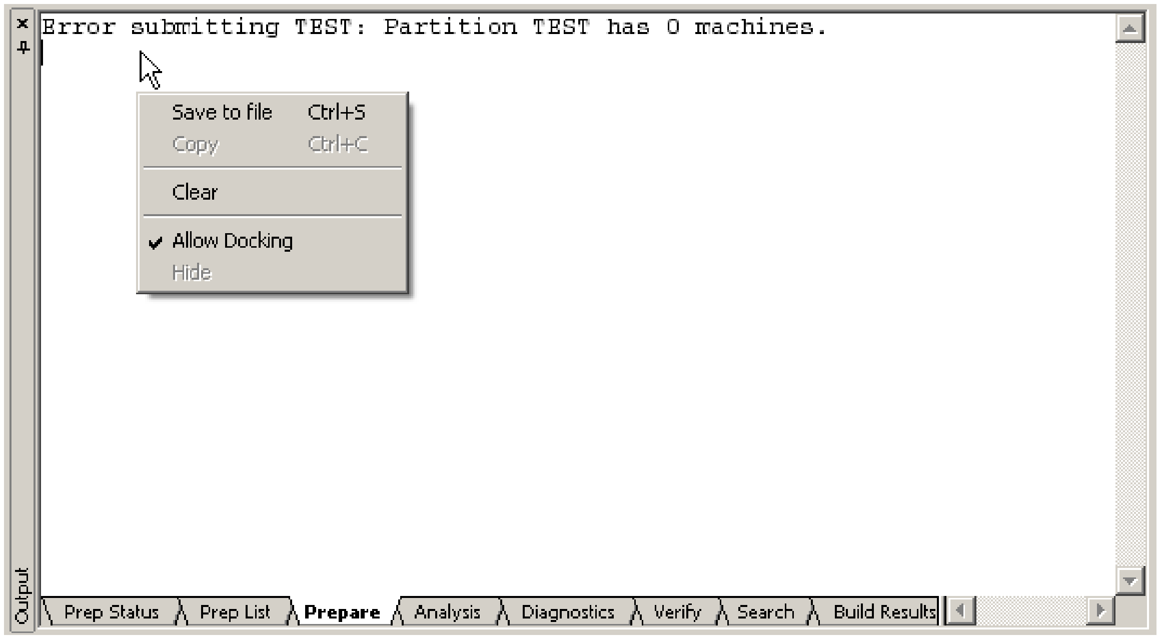
For more information about preparation in general, refer to the *Deploying Applications Guide*.

Prepare Tab

The **Prepare** tab displays any operational errors when preparing an object in the hierarchy. An example is shown in Sample Prepare tab. This tab is used as the place to display such errors as opposed to displaying an error message box for each error because there may be many such messages when a Super Prepare is performed. If there is a problem with the preparation, an error message is displayed in this tab.

To save the results, right-click in the Prepare tab, then select **Save to file**. The Save As dialog box appears. Select the location, then click **Save**. The default location is the AppBuilder directory, and the default file name is Result.txt.

Sample Prepare tab



```
Error submitting TEST: Partition TEST has 0 machines.
```

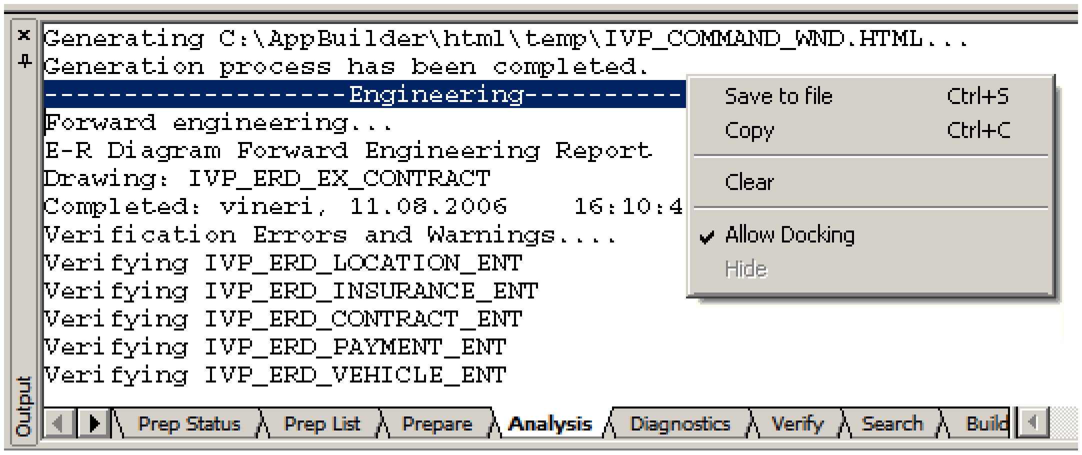
If the displayed message is longer than the window (which can happen if you have the Output window undecked), use the left and right arrow keys to navigate through the entire length of the message.

For more information about preparation, refer to the *Deploying Applications Guide*.

Analysis Tab

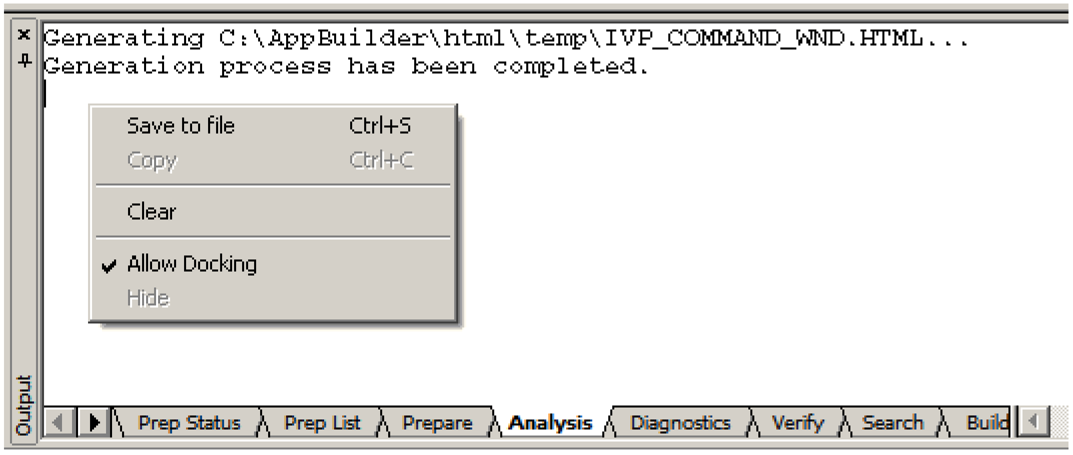
The **Analysis** tab displays the results of engineering tools (as shown in the sample in Sample engineering tools results in Analysis tab) and the results of HTML generation (as shown in the sample in Sample HTML generation results in Analysis tab).

Sample engineering tools results in Analysis tab



```
Generating C:\AppBuilder\html\temp\IVP_COMMAND_WND.HTML...
Generation process has been completed.
-------------------Engineering----------
Forward engineering...
E-R Diagram Forward Engineering Report
Drawing: IVP_ERD_EX_CONTRACT
Completed: vineri, 11.08.2006    16:10:4
Verification Errors and Warnings....
Verifying IVP_ERD_LOCATION_ENT
Verifying IVP_ERD_INSURANCE_ENT
Verifying IVP_ERD_CONTRACT_ENT
Verifying IVP_ERD_PAYMENT_ENT
Verifying IVP_ERD_VEHICLE_ENT
```

Sample HTML generation results in Analysis tab



```
Generating C:\AppBuilder\html\temp\IVP_COMMAND_WND.HTML...
Generation process has been completed.
```

To save the results, right-click in the Analysis tab, then select **Save to file**. The Save As dialog box appears. Select the location, then click **Save**. The default location is the AppBuilder directory, and the default file name is Result.txt.

Diagnostics Tab

The **Diagnostics** tab displays messages that relate to problems with the overall operation of the Construction Workbench. For example, if you are expanding a large hierarchy in the Hierarchy Diagram tool and a sequence number clash is detected, an error message is printed to the Output window Diagnostics Tab. The information in this tab helps Support troubleshoot problems. When contacting Magic Software Support, provide information that is displayed on this tab.

To save the results, right-click in the Diagnostics tab, then select **Save to file**. The Save As dialog box appears. Select the location, then click **Save**. The default location is the AppBuilder directory, and the default file name is Result.txt.

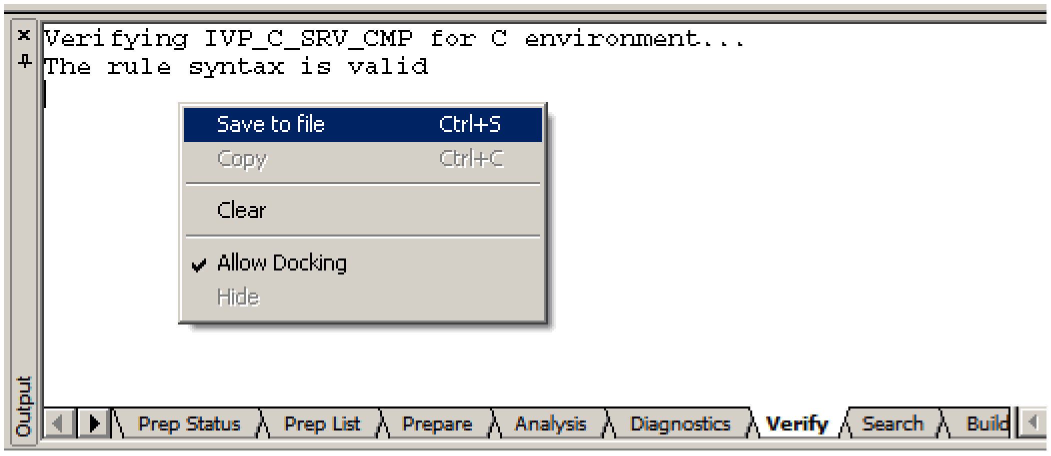
Verify Tab

The **Verify** tab displays reports from verifications:

- Engineering Relationship Diagram (ERD) Verification report
- Rule Verification report

An example is shown in Sample Verification report.

Sample Verification report



```
Verifying IVP_C_SRV_CMP for C environment...
The rule syntax is valid
```

If there are no errors or warnings, the system displays the following message:
`The rule syntax is valid.`

If there are errors or warnings when verifying a rule, they are listed in this tab. Double-click the error or warning in the **Verify** tab and the Rule Window (for the rule that contains the error) is displayed. If the verification report indicates errors or warnings in the rule, you can quickly locate the error within the rule.

To save the results, right-click in the Verify tab, then select **Save to file**. The Save As dialog box appears. Select the location, then click **Save**. The default location is the AppBuilder directory, and the default file name is Result.txt.

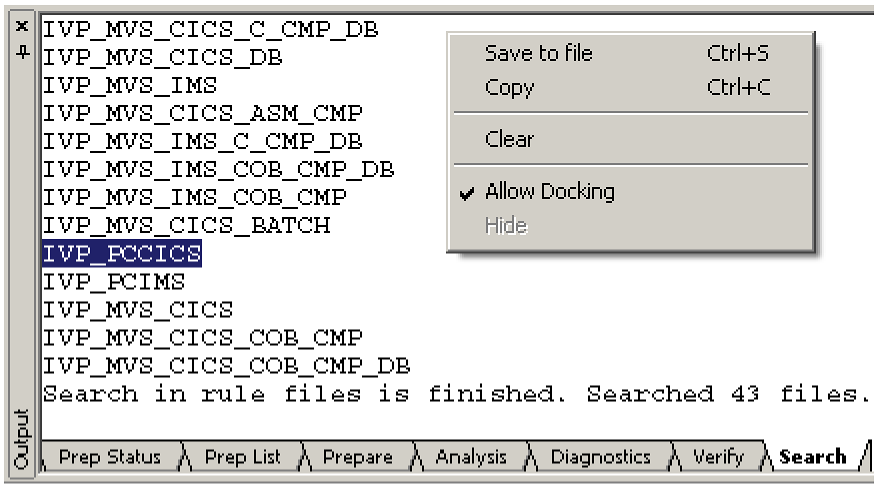
For more information about debugging, refer to the *Debugging Applications Guide*.

Search Tab

The Search tab displays the search result for the specified string in Rules, Components, or Windows. To search for a string in a Rule, Component, or Window, double-click the name of the Rule, Component, or Window, or double-click one of the found lines to bring up the Rule, Component, or Window that contains the text in the Rule Painter, the Component, or the Window Painter.

An example is shown in Sample search report.

Sample search report



```
IVP_MVS_CICS_C_CMP_DB
IVP_MVS_CICS_DB
IVP_MVS_IMS
IVP_MVS_CICS_ASM_CMP
IVP_MVS_IMS_C_CMP_DB
IVP_MVS_IMS_COB_CMP_DB
IVP_MVS_IMS_COB_CMP
IVP_MVS_CICS_BATCH
IVP_PCCICS
IVP_PCIMS
IVP_MVS_CICS
IVP_MVS_CICS_COB_CMP
IVP_MVS_CICS_COB_CMP_DB
Search in rule files is finished. Searched 43 files.
```

To save the search results, right-click in the Search tab, then select **Save to file**. The Save As dialog box appears. Select the location, then click **Save**. The default location is the AppBuilder directory, and the default file name is Result.txt.

For more information regarding the search function, see Searching for Objects and Text in Using the Hierarchy Window.

Build Results Tab

The Build Results tab displays the Codegen errors of the objects that are built. If there are errors or warnings when building an object, they are listed in this tab. Double-click the error or warning in the **Build Results** tab and the Class Editor (for the object that contains the error) is displayed with the cursor on the error line.

Rebuild Report Tab

This tab displays the list of objects that need to be rebuilt. The rebuild menu options are only enabled when "Rebuild" functionality is selected during the installation process with a workgroup repository.

Modeling Tools

Modeling tools help you design the application and perform some analysis before you begin development. Access the modeling tools from the Construction Workbench when you create a new diagram or drawing or when you open an existing diagram or drawing.
Each modeling tool has a toolbar associated with it. Diagrams are saved in the repository as instances of the diagram entity type. These diagrams graphically portray the entities and relationships between them. The position of boxes and lines relative to each other is stored in the diagram. The boxes and lines in the diagram also store pointers to the repository entities they represent.

Understanding Modeling Tools

The following modeling tools are available in AppBuilder:

- Entity Relationship Diagram
- Database Diagram
- Process Dependency Diagram
- State Transition Diagram
- Window Flow Diagram
- Class Diagram

Entity Relationship Diagram

Early in the software development process, you specify data requirements for the system you are building by using data modeling. One part of data modeling is creating an entity relationship diagram (ERD). Tasks include:

- Creating or Opening an Entity Relationship Diagram
- Inserting Objects into the Entity Relationship Diagram
- Analyzing an ERD

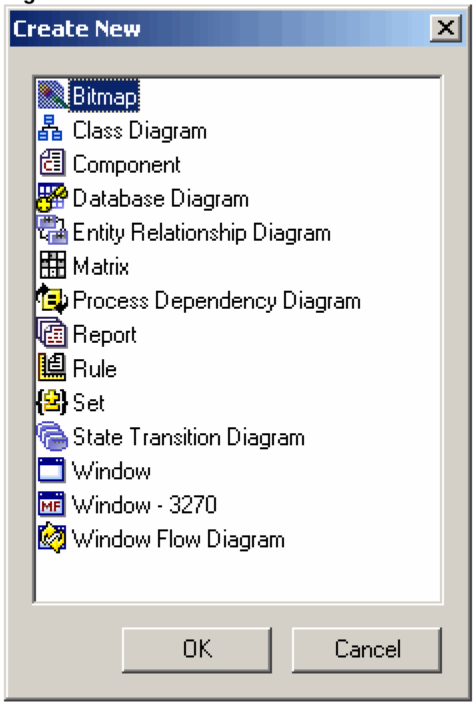
See Forward Engineering an ERD for how the forward engineering process translates logical entities (entities, identifiers, and attributes) into relational entities.

Creating or Opening an Entity Relationship Diagram

You can create a new entity relationship diagram or open one from the repository. When an ERD is created or opened, the diagram is displayed in the Work Area of the Construction Workbench.

Creating a New Entity Relationship Diagram

To create a new entity relationship diagram, complete the following steps:

1. From the Construction Workbench menu, click **File > New**.
   The Create New window displays.
   Figure 8-1 Create New window
   

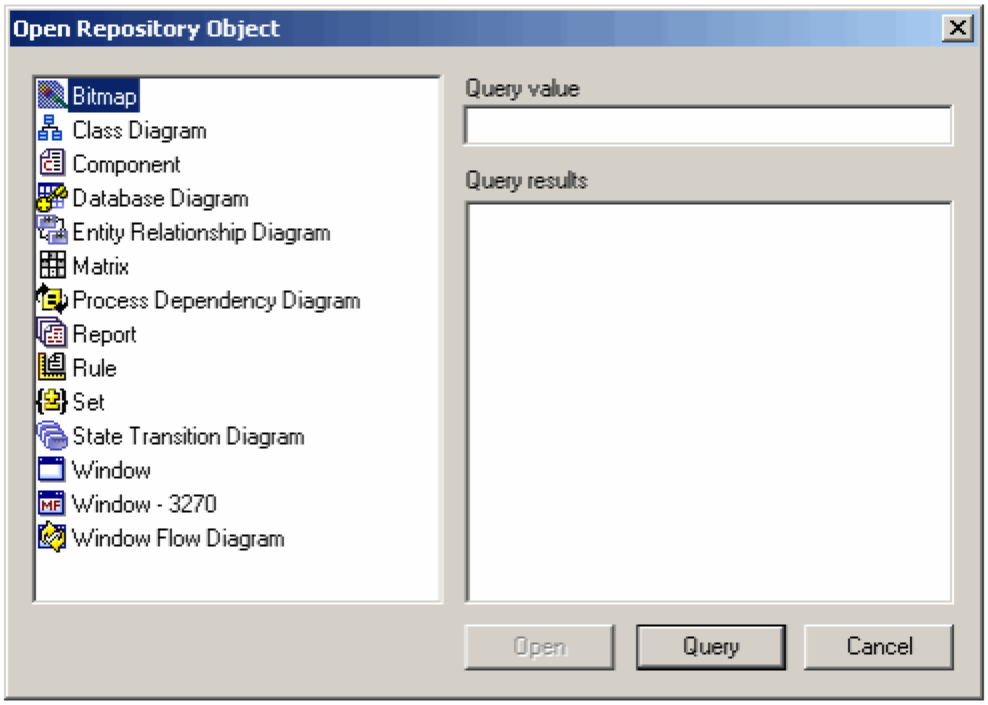
2. Select Entity **Relationship Diagram**.
3. Click **OK**.

An empty Entity Relationship Diagram displays in the Work Area.

Opening an Existing Entity Relationship Diagram

To use a diagram from the repository, complete the following steps:

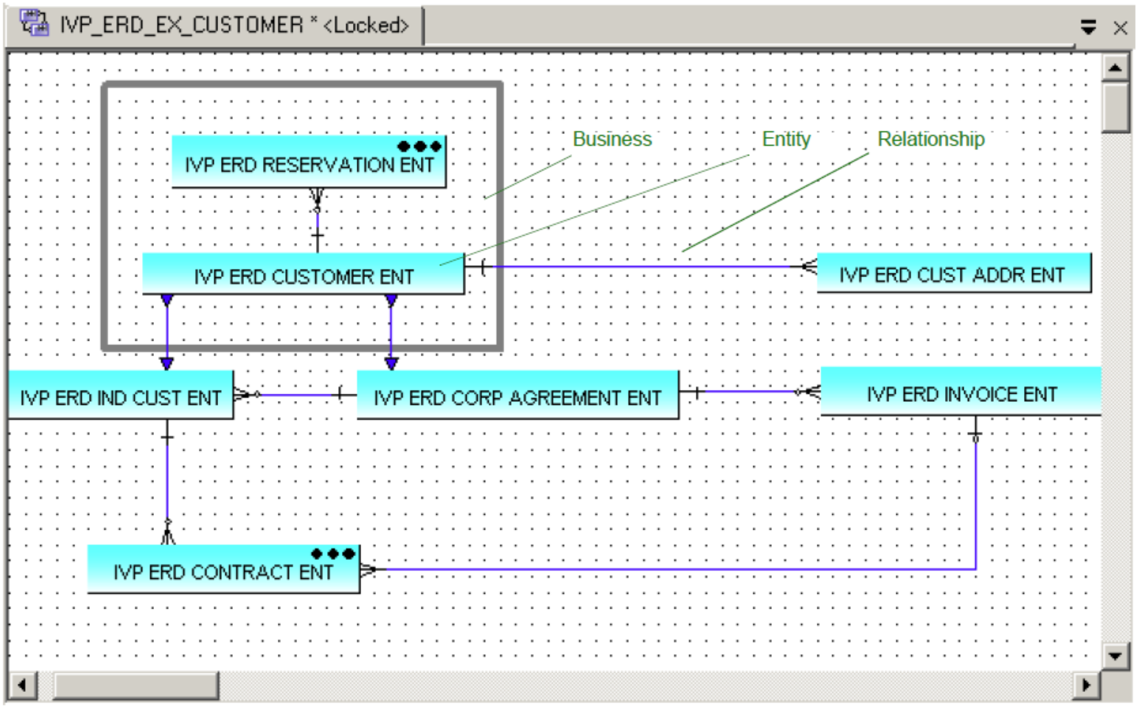
1. From the Construction Workbench menu, click **File > Open**.
   The Open Repository Object window displays.
   Figure 8-2 Open Repository Object window



2. Select **Entity Relationship Diagram**.
3. Click **Query** button to display a list of Entity Relationship Diagrams in the repository.
4. Click to highlight the diagram you want to use and click the **Open** button.
   The diagram displays in the work area.

The toolbar associated with the Entity Relationship Diagram is also displayed in the toolbar area. A sample Entity Relationship Diagram is shown in the following figure:

Figure 8-3 Entity Relationship Diagram sample



Inserting Objects into the Entity Relationship Diagram

When an ERD is displayed in the Construction Workbench Work area, you can insert objects, such as Entities, Business Objects, or Relationships. To add or insert an object to the ERD, use the Insert menu in the Construction Workbench or the Entity Relationship Diagram (ERD) toolbar (see Entity Relationship Diagram Toolbar).
You can insert the following objects into the ERD:

- Entity
- Business Object
- Relationship
- Subtype Relationship
- Cardinality (One mandatory, One optional, Many mandatory, Many optional, Unknown)
- Note (see Adding a Note in a Diagram)

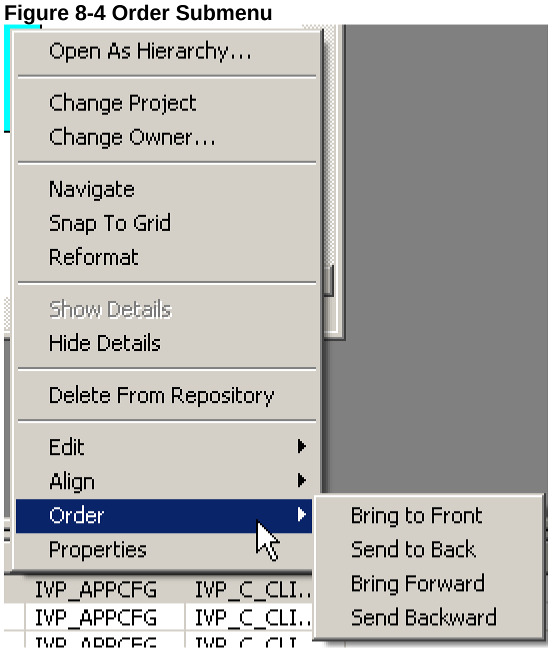
To insert an Entity, complete the following steps:

1. Click the **Entity** button in the ERD toolbar or select from the menu **Insert > Entity**.
2. Click inside the ERD where you want to place the Entity.
   The Entity is displayed as a gray box until you define the Entity.
3. Double-click the Entity.
   The **Insert Entity** dialog displays.
4. Click the **Query** button to display a list of entities in the repository or type a new entity name and click **Insert**.

For more information about adding objects to a diagram, refer to Adding an Object to a Diagram.

While you use the ERD to place the entities and relationships in a diagram, use the Hierarchy window to add attributes to an Entity. See Attribute in Using the Hierarchy Window for information about attributes. Attributes must be defined for each Entity to forward engineer the ERD. Refer to Creating Attribute Hierarchies in the Entity Relationship Diagram for instructions to create attributes.

You can also change where an Entity appears in the ERD with relation to other Entities. You can bring Entities forward in the ERD, or you can send Entities back in the ERD. To change where an Entity appears in the ERD, complete the following steps:

1. Right-click the Entity in the ERD.
   A popup menu displays.
   Figure 8-4 Order Submenu
   

2. Select one of the following:
   - Select **Order > Bring to Front** to put the selected Entity in front of all other Entities.
   - Select **Order > Send to Back** to put the selected Entity behind all other Entities.
   - Select **Order > Bring Forward** to move the selected Entity forward in front of one other Entity.
   - Select **Order > Send Backward** to move the selected Entity backward behind one other Entity.

Relations

A Relationship defines the interconnections between entities. In the Entity Relationship Diagram (ERD), a relationship can have two roles, "From" and "To". A recursive relationship relates an entity to itself. An Entity can have more than one relationship with itself.

To add a relationship between entities, complete the following steps:

1. Click the **Relationship** button in the ERD toolbar or select from the menu **Insert > Relation**.
2. Click inside the Entity from which the relationship starts, then click inside the Entity to which the relationship ends.
3. Click the relationship line to display the Object Property window for the relationship. Specify the roles of the relationship.
4. The ERD shows the role as a label for "From" relationship beside the cardinality it describes. You can display "From" only, or both "From" and "To", or no labels. See Modifying Aspects of a Diagram.

Cardinality

In the Entity Relationship Diagram (ERD), cardinalities are displayed at each end of a relationship line where the relationship meets an entity. They are also represented by the relationship label.
To specify the cardinality of a relationship, complete the following steps:

Select a cardinality type from the Insert menu or click one of the cardinality types on the ERD toolbar.

1. Click where the relationship line intersects with the Entity to which you want to apply the cardinality type.

The following table briefly describes the five cardinality types:

Table 8-1 Cardinality Types

Graphical Representation	**Cardinality Type**	**Description**
┼─	**One mandatory**	The one mandatory cardinality dictates that an entity must have one relationship with another entity. For example, a customer must have one and only one customer ID.
┼○─	**One optional**	The one optional cardinality means that an entity may have no relationship or one relationship with another entity.
>─	**Many mandatory**	The many mandatory cardinality dictates that an entity must have one or many relationships with another entity. For example, a customer must have an address but may have more than one address.
>○─	**Many optional**	The many optional cardinality dictates that an entity may have zero, one, or many relationships with another entity.
─	**Unknown**	Unknown cardinality type.

Subtype Relationship

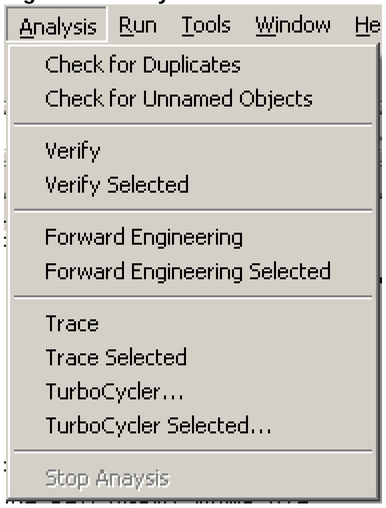
A Subtype relationship ↦ is a hierarchical relationship that shows shared aspects of entities.

Analyzing an ERD

When you have created a diagram, several functions are available through the *Analysis* menu (Figure 8-5) of the Construction Workbench. This section discusses the following functions:

- Check for Duplicates and Unnamed Objects in ERD
- Verify
- Forward Engineering
- Trace

Figure 8-5 Analysis menu



For information about using TurboCycler and TurboScripter, refer to the *Scripting Tools Reference Guide*.

Check for Duplicates and Unnamed Objects in ERD

AppBuilder can check for duplicate objects or unnamed objects when you are done with a diagram. This is helpful with large diagrams that have many objects. From the *Analysis* menu, select *Check for Duplicates* or *Check for Unnamed Objects*. Check for Duplicates searches for any objects used twice in the current diagram.

Verify

AppBuilder can verify the diagram when you are done with it. From the *Analysis* menu select *Verify* or *Verify Selected*.

Forward Engineering

Select *Forward Engineering* or *Forward Engineering Selected* to *forward engineer* your diagram. Read more about forward engineering under the section Forward Engineering an ERD.
See *Developing Applications Guide* for an example of forward engineering an ERD.

Trace

Use the *trace analysis* process in the Entity Relationship Diagram to see how forward engineering translated logical entities into relational entities. Trace analysis tracks and reports how the logical entities in an ERD correspond to the relational entities in the database model.
See *Developing Applications Guide* for an example of performing a trace analysis on an ERD.

Forward Engineering an ERD

You can forward engineer any entity. To forward engineer an entity, complete the following steps:

1. Right-click the entity in the ERD, then select **Open as Hierarchy** or select **Edit > Open as Hierarchy** to display it in the Hierarchy window. The selected entity is displayed in the Repository tab of the Hierarchy window.
2. Select **Build > Forward Engineering** from the menu bar. You can also right-click the entity in the Hierarchy window and select **Forward Engineering**.

The forward engineering process in the ERD translates logical entities (entities, identifiers, and attributes) into relational entities (tables, keys, and columns). Forward engineering creates a database diagram containing a table for each ERD entity, keys from their identifiers, and columns from their attributes. Forward engineering maintains inheritance information so you can do a trace analysis to see how the logical entities translated into relational entities.

The following sections describe how to complete the ERD you plan to forward engineer and the steps in the forward engineering process. These sections include explanations of the relationship patterns that the forward engineering process follows and a list of error messages.

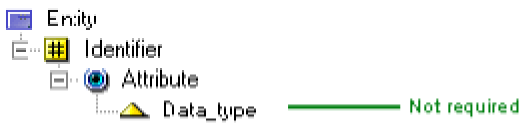
The following topics are discussed in this section:

- Creating Attribute Hierarchies in the Entity Relationship Diagram
- Forward Engineering an Entity
- Viewing the Forward Engineering Report
- Verifying the Entity Relationship Diagram
- Renaming Generated Names
- Understanding Relationship Patterns
- Handling Error Messages
- Performing Trace Analysis on an ERD

Creating Attribute Hierarchies in the Entity Relationship Diagram

Before you begin forward engineering, create an attribute structure for each entity you want to translate. An attribute structure can include attributes, data types, identifiers, and relations. Each kernel entity should have at least one unique identifier and one attribute, and each attribute should have a data type. Minimum attribute structure shows the minimum attribute structure you need for forward engineering a kernel entity. A data type attribute is not strictly required, because not having it in the attribute hierarchy generates only a warning - rather than an error - when you verify the Entity Relationship Diagram (ERD).

Figure 8-6 Minimum attribute structure



To create an attribute hierarchy, complete the following steps:

1. Click to select one or more entities in the ERD. Hold down **Ctrl** key while clicking to select multiple entities.

1. To create an attribute hierarchy, select **Edit > Open as Hierarchy** from the Construction Workbench menu or right-click the entity and select **Open as hierarchy** from the pop-up menu.
The selected entity is displayed in the Repository tab of the Hierarchy window.

1. Do one of the following to insert an Attribute or Identifier:
    - Click to select an Entity in the Repository tab of the Hierarchy window and from the Construction Workbench menu, select **Insert Child > Attribute** or **Identifier**.
    - Right-click an Entity in the Repository tab of the Hierarchy window, select **Insert Child** from the pop-up menu, and select **Attribute** or **Identifier**.
2. Do one of the following to create a Data Type for an Attribute:
    - Select the Attribute in the Repository tab of the Hierarchy window and select **Insert Child > Data Type** from the Construction Workbench menu.
    - Right-click the Attribute, select **Insert Child** from the pop-up menu, and select **Data Type**.
3. Display the Object Property window for the Data Type object and set data format if necessary.

See *Developing Applications Guide* for an example of attribute structures of a sample ERD.

Forward Engineering an Entity

Before you forward engineer an Entity Relationship Diagram (ERD), make sure that each kernel entity has at least one unique identifier and at least one attribute, and that all attributes have a data type.
You can Forward Engineer an Entity without open ERD drawing, directly through Entity objects from Hierarchy tree.
To forward engineer an Entity, complete the following steps:

Display the Entity objects in the Hierarchy window.

1. To make the Entity Relationship Diagram window active in the Construction Workbench, click the window or select the window from the Construction Workbench **Windows** menu.
2. Check each relationship in the diagram to ensure it has a cardinality symbol at each end. If any cardinality symbols are missing, place them.
3. Select *Edit* > *Show details* to show the hierarchy for each entity in the diagram and check each entity to ensure it has the attribute structure as described in Building Attribute Hierarchies.
4. Select **Build > Forward Engineering** from the Construction Workbench menu or right-click the Entity in the Hierarchy window and select **Forward Engineering**.

You can also forward engineer selected parts of an Entity by selecting specific entities or many-to-many relationships and selecting **Build > Forward Engineering Selected**.
As forward engineering progresses, the message line displays status messages at the bottom of the Analysis tab of the Output window.
When the process is complete, the system displays a status report on the *Analysis* tab of the Output window to display any errors or warnings. See Viewing the Forward Engineering Report.
The system displays a message in the **Analysis** tab of the Output window showing the status of the process. If you encounter errors, repeat steps 1-5.

1. When you have successfully created the table from the entity relationship diagram, create a new database diagram. From the Construction Workbench menu, select **File > New > Database Diagram**.
2. Click the Table button on the Database: Diagram toolbar to insert a Table object.
3. Double-click the table in the database diagram. The Insert Table dialog displays. Use the **Query** button to select one or more of the tables just created.

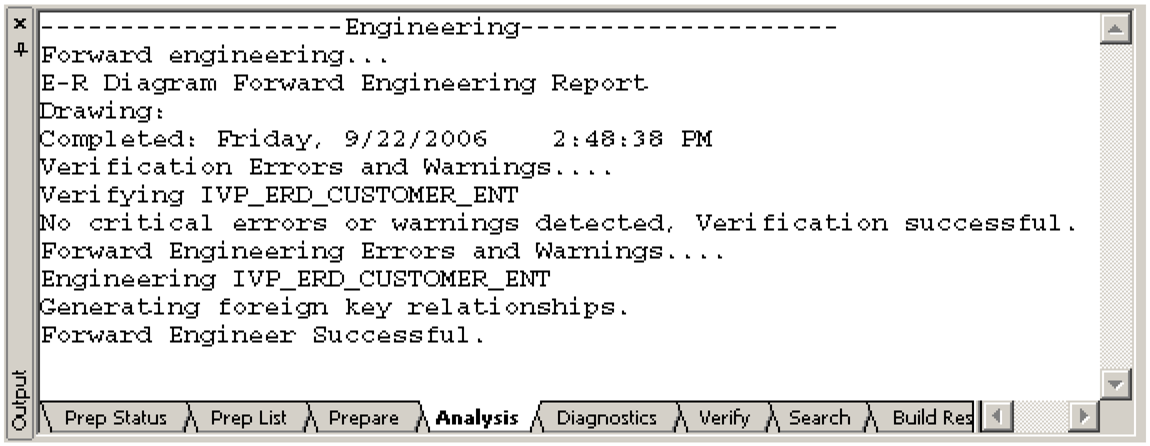
The selected table is inserted.

1. In the Database Diagram, you can display the tables and keys that forward engineering has created To expand the objects, use F8. This shows the entity and its keys. For details about the results of Forward Engineering, see Performing Trace Analysis on an ERD.
2. Select **File > Commit** from the Construction Workbench menu. If you have displayed the columns and keys in the Database Diagram, name and save the diagram.

Viewing the Forward Engineering Report

After completing forward engineering, the system displays the results in the *Analysis* tab of the Output window. This forward engineering report lists verification and forward engineering warnings and errors. The report in Forward Engineering report shows that both verification and forward engineering completed without any warnings or errors.

Figure 8-7 Forward Engineering report



```
-------------------Engineering-------------------
Forward engineering...
E-R Diagram Forward Engineering Report
Drawing:
Completed: Friday, 9/22/2006    2:48:38 PM
Verification Errors and Warnings....
Verifying IVP_ERD_CUSTOMER_ENT
No critical errors or warnings detected, Verification successful.
Forward Engineering Errors and Warnings....
Engineering IVP_ERD_CUSTOMER_ENT
Generating foreign key relationships.
Forward Engineer Successful.
```

Verifying the Entity Relationship Diagram

Verification ensures that the entities and relationships in the ERD are correct for forward engineering. The forward engineering process automatically verifies the diagram, so you do not have to verify it before you forward engineer. However, it is a good practice to verify the ERD before forward engineering to save time.
To verify an ERD, complete the following steps:

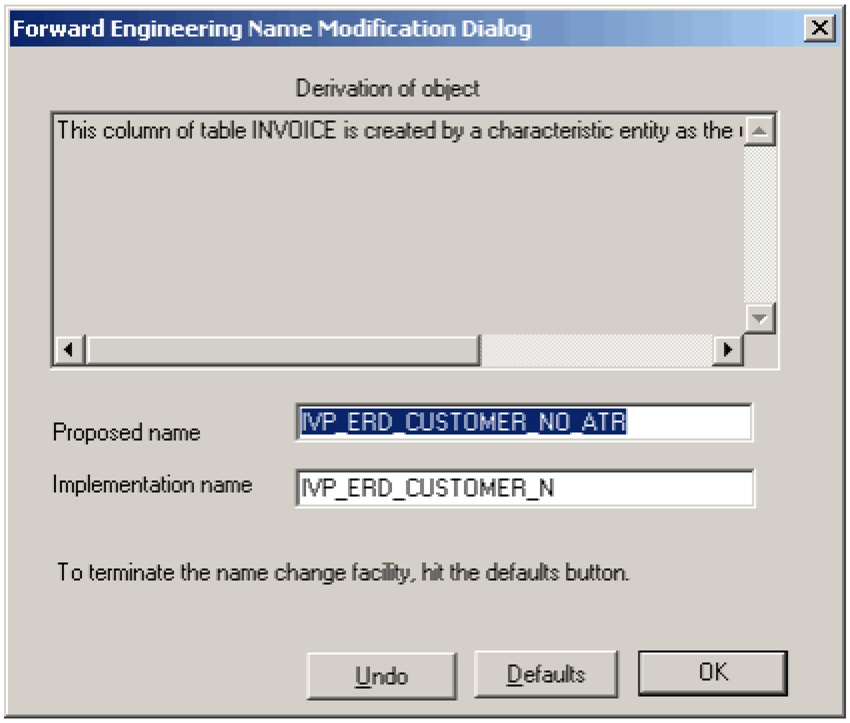
1. Display the Entity Relationship Diagram (ERD) tab in the Work area.
2. Do one of the following to verify selected parts of an ERD:
3. Select *Analysis* > *Verify* from the Construction Workbench menu.
4. Select specific entities and select **Analysis > Verify selected**.
5. When the process is complete, the system displays a status report on the **Analysis** tab of the Output window to display any errors or warnings.

The verification process also alerts you to potential problems by displaying warnings, which the forward engineering report explains. Unlike errors, warnings do not stop the forward engineering process.

Renaming Generated Names

The forward engineering process gives new objects system-created names based on the name of the original entity. For example, forward engineering the **CUSTOMER** entity creates a table named **CUSTOMER**.
You can develop a database schema that promotes view-to-view mapping by renaming each object as it is created. To create custom names for generated objects, select the **Query user** option on the **Engineering** tab of the Workbench Options window (as in Engineering options).
If the **Query user** option is checked here, when you create your ERD with forward engineering (select **Analysis > Forward Engineering**), the Name Modification window (Changing generated names) displays, and forward engineering creates each new object. You can specify a prefix or suffix to be used with the object name when forward engineering creates new objects and a default column type for any attribute that does not have a data type.
To stop displaying the Name Modification window and resume automatic object naming, click the *Defaults* button. Selecting the *Defaults* button is equivalent to unselecting the **Query user** option in the Engineering section of the Workbench Options window.

Figure 8-8 Changing generated names



Derivation of object

This column of table INVOICE is created by a characteristic entity as the

Proposed name IVP_ERD_CUSTOMER_NO_ATR

Implementation name IVP_ERD_CUSTOMER_N

To terminate the name change facility, hit the defaults button.

Understanding Relationship Patterns

The forward engineering process uses several patterns when translating logical entities in an ERD to relational entities in a Database Diagram (DBD). The pattern used depends on the entities and relationships in the original ERD and the attribute hierarchy of each entity.
This section discusses the following patterns:

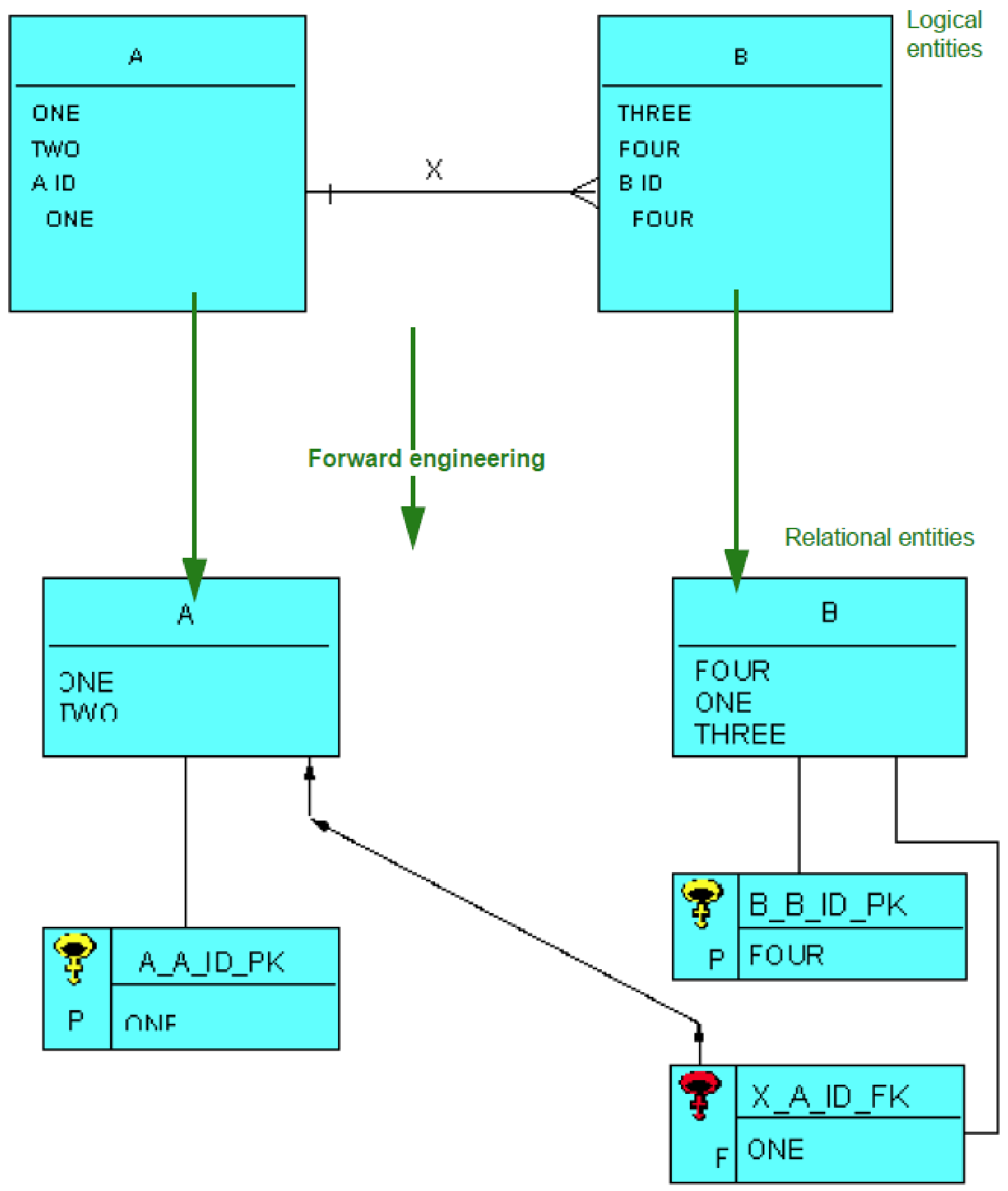
- One-to-One or One-to-Many
- Many-to-Many
- Supertype and Subtype
- Recursive

One-to-One or One-to-Many

The One-to-many relationship type is the most common relationship. A simple example is one sales person has many customers. Forward engineering creates two table entities, one for each entity in the original ERD. The primary identifier for each entity becomes the primary key attached to its table. Any candidate or associate identifiers become index keys. Finally, a foreign key is created to represent the relationship between the two entities.

For example, One-to-many relationship before and after forward engineering shows the tables, columns, and keys that forward engineering created for two entities with a one-to-many relationship. Entity **A** displays its attributes (**ONE** and **TWO**) and its identifier (**A_ID**). Forward engineering creates a table named **A** with columns **ONE** and **TWO**, and a primary key (**A_A_ID_PK**).
Similarly, entity **B** becomes a table **B**, its attributes (**THREE** and **FOUR**) become columns, and its identifier (**B_ID**) becomes the primary key (**B_B_ID_PK**).
Forward engineering creates a foreign key named **X_A_ID_FK** to reflect the relationship (named **X**) between **A** and **B**. This foreign key is related to table **B** through the Table **owns** Key relationship because the many cardinality is on the many side of the **X** relationship. The **X_A_ID_FK** foreign key is related to table **A** through the Table **refers-to** Key relationship. The column reflecting the primary key of entity **A** is inherited through the foreign key into table **B**.

Figure 8-9 One-to-many relationship before and after forward engineering



Many-to-Many

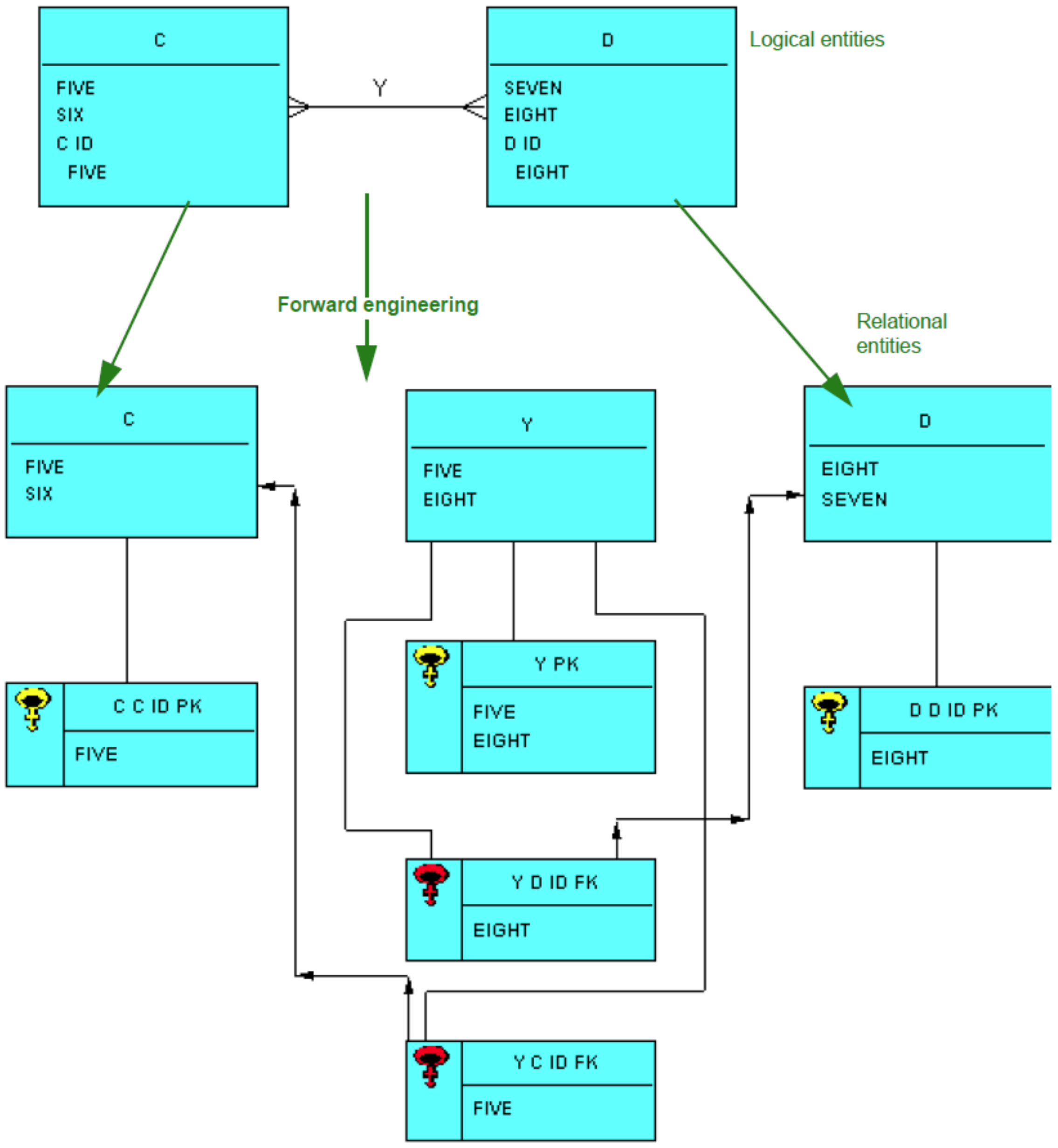
A simple example of a many-to-many relationship is many employees have many tasks. In a many-to-many relationship, forward engineering creates one table for each entity and an additional table containing intersection data. The name of the intersection table is based on the name of the relationship between the two entities.

The example in Many-to-many relationship before and after forward engineering shows that the entity **C** has two attributes (**FIVE** and **SIX**) and one identifier (**C_ID**). The **D** entity also has two attributes (**SEVEN** and **EIGHT**) and one identifier (**D_ID**), and is related to the **C** entity by a many-to-many relationship named **Y**.

As Many-to-many relationship before and after forward engineering shows, forward engineering creates three tables: **C**, **D**, and an intersection table (**Y**) named for the relationship between the **C** and **D** entities.

Table **Y** and its primary key (**Y_PK**) both have two columns, each of which reflects the attribute in their identifiers, **FIVE** and **EIGHT**, inherited through the foreign keys. **Y** has two related foreign keys (**Y_D_ID_FK** and **Y_C_ID_FK**) each reflecting the identifiers of the two entities. The Key **refers-to** Table relationship relates each foreign key to the table it refers to.

Figure 8-10 Many-to-many relationship before and after forward engineering



Supertype and Subtype

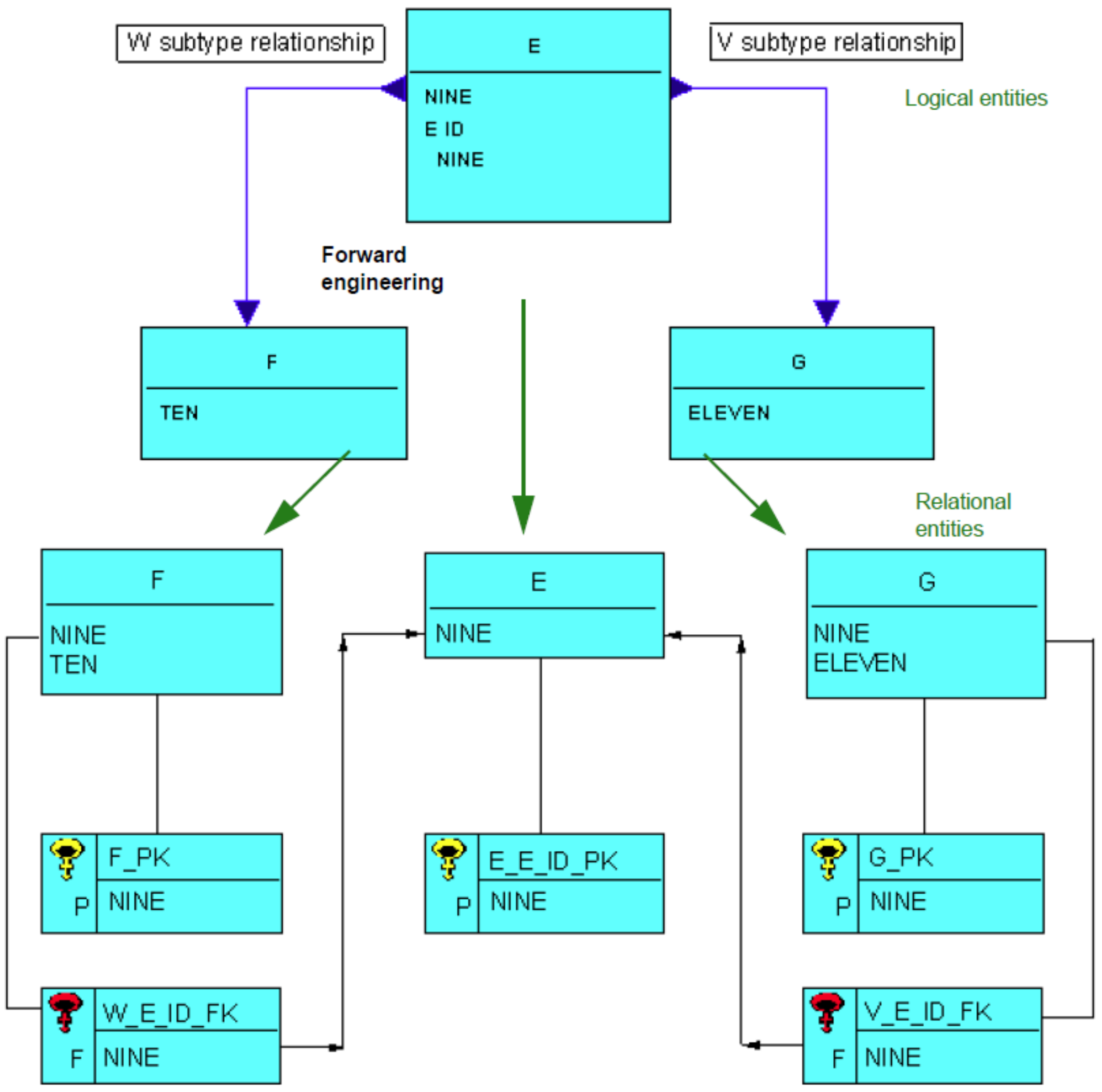
A supertype entity is an entity that can be broken down into smaller entities or subtypes. For example, a supertype entity EMPLOYEE can have subtype entities FULL-TIME and PART-TIME.

Forward engineering creates one table per supertype and one table per subtype. The supertype table contains columns for its attributes and a primary key for its identifier. The subtypes have columns for their own attributes and columns corresponding to all the supertype primary attributes.

The example in Entity and subtype entities before and after forward engineering shows that forward engineering creates a table for each entity: table **E**, table **F**, and table **G**. The table created from the **E** supertype has one column because it has one attribute (**NINE**). Tables that implement subtype entities inherit from the table that implements their supertype. Forward engineering creates subtype tables that have their own primary key and a foreign key. Both keys contain a column corresponding to every column of the supertype table primary key. The columns for both the primary and foreign keys appear as the table children.

So in this example, forward engineering creates primary keys that reflect the identifiers for each entity: **E_E_ID_PK** for table **E**, **F_PK** for table **F**, and **G_PK** for table **G**, based on the relationship between them and E. Finally, as Entity and subtype entities before and after forward engineering shows, forward engineering creates two foreign keys related to tables **F** and **G**, which refer back to the **E** supertype through their columns and the Key *refers-to* Table relationship.

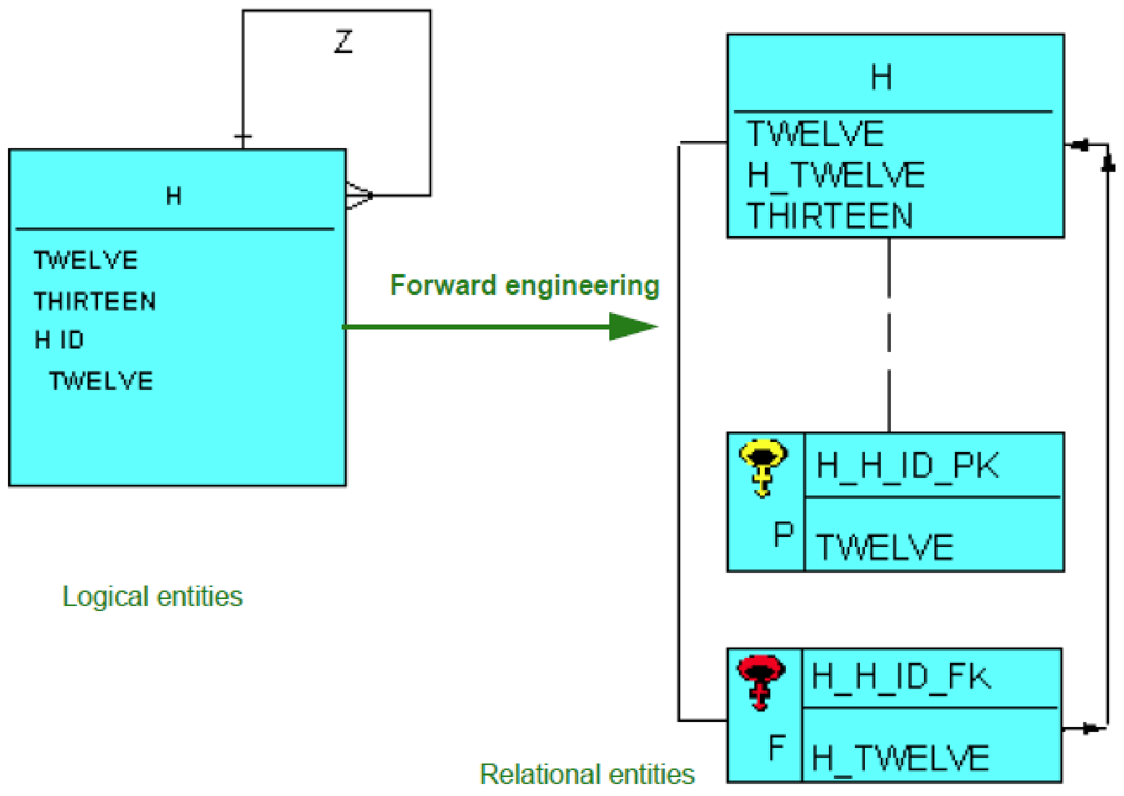
Figure 8-11 Entity and subtype entities before and after forward engineering



Recursive

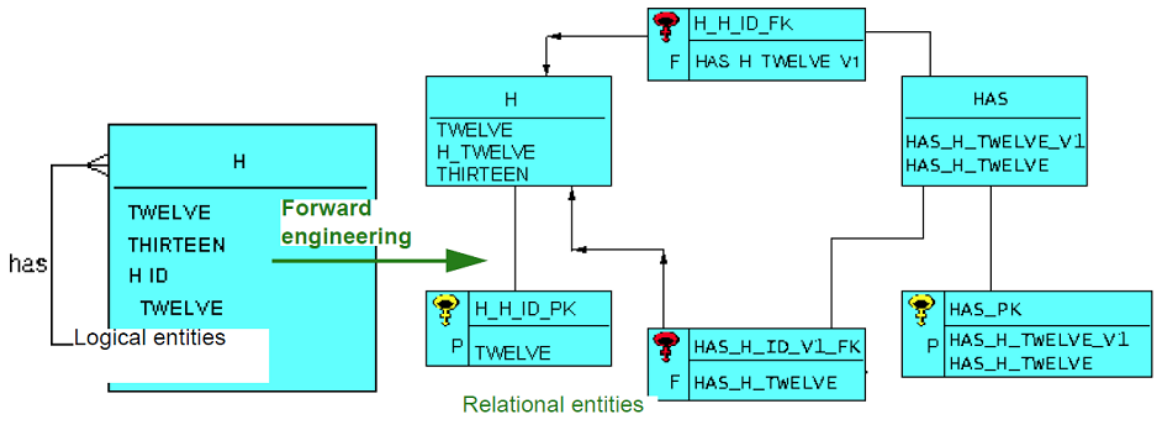
A recursive relationship is a relationship between an entity and itself. A simple example is a female person entity is a mother of a person and a child of a person. An entity with a one-to-many recursive relationship converts according to the one-to-many relationship, except an additional foreign key is added to reflect the relationship the entity can have with itself. The example in Recursive one-to-many relationship before and after forward engineering shows that during forward engineering, the recursive **Z** relationship that relates **H** to itself produces a table named **H** with attributes of entity **H** as columns, a primary key that reflects the entity identifier of entity **H**, and a foreign key named for the recursive relationship (**Z**) between **H** and itself.

Figure 8-12 Recursive one-to-many relationship before and after forward engineering



An entity with a many-to-many recursive relationship converts with two foreign keys reflecting the relationships the entity can have with itself. The example in Recursive many-to-many relationship before and after forward engineering shows that during forward engineering, the recursive **has** relationship that relates **H** to itself produces a table named **H** with entity **H** attributes as columns, a primary key that reflects the entity **H** identifier, and two foreign keys named for the recursive **has** relationship between **H** and itself. (Versioning, indicated with a _**V1** suffix, is used to resolve conflicting names.)

Figure 8-13 Recursive many-to-many relationship before and after forward engineering



Handling Error Messages

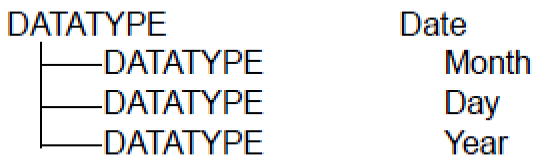
Error messages you can encounter during forward engineering (and thus are displayed in the Forward Engineering Report window) are explained in the following table.

Error Messages in forward Engineering

Error Message	Description
Associative Entity Is Missing an Attribute	Each associative entity must have at least one attached Attribute.

Characteristic Entity is missing an attribute	Each characteristic entity must have at least one attached Attribute.
Entity has more than one Primary Identifier	An entity can have only one primary identifier, because the primary identifier serves as the index into the generated table.
Identifier is missing an Attribute or a Relationship	Each identifier must have at least one attached relationship or Attribute.
Kernel Entity is missing an Attribute	Each kernel entity must have at least one Attribute attached to it or its identifier.
Kernel Entity is missing a Primary Identifier	Each non-subtype kernel entity must have one attached primary identifier.
Leaf Attribute defined with more than one data type	Each attribute can have only one data type, which defines the type of information the attribute contains. However the data type can be complex, such as the one that Complex data type shows.
Non-leaf Attribute defined with a data type	Only leaf attributes can have data types.
Relationship defined with unknown cardinality for Entity	Both ends of all relationships in an entity relationship diagram must have a cardinality.
Subtype Entity defined with a Primary Identifier	Primary identifiers cannot attach directly to subtype entities because subtypes inherit their primary identifiers from the relationship to the supertype.

Figure 8-14 Complex data type



Performing Trace Analysis on an ERD

Trace analysis tracks how forward engineering converted entities to tables and produces reports that you can view.

To perform a trace analysis on an ERD, complete the following steps:

1. Display the Entity Relationship Diagram in the Work area. Click the diagram tab or select it from the Construction Workbench **Windows** menu.
2. Do one of the following to verify selected parts of an ERD:
   - Select *Analysis* > *Trace* from the Construction Workbench menu.
   - Select specific entities and select **Analysis > Trace Selected**.
     The system displays the status of the process in the *Analysis* tab of the Output window.
     When the process is complete, the system displays a status report on the *Analysis* tab of the Output window to display a list of the entities you traced along with the corresponding tables, columns, and keys in the repository.
     If the process concludes with errors, make sure you downloaded the diagram correctly. If necessary, delete the columns the previous forward engineering process created, forward engineer the ERD again, and then do another trace analysis.
3. Select **File > Print** from the Construction Workbench to print the report to a printer or file.

Refer to the example *Performing Trace Analysis* on an ERD in *Developing Applications Guide*, which shows an example traceability report.

Database Diagram

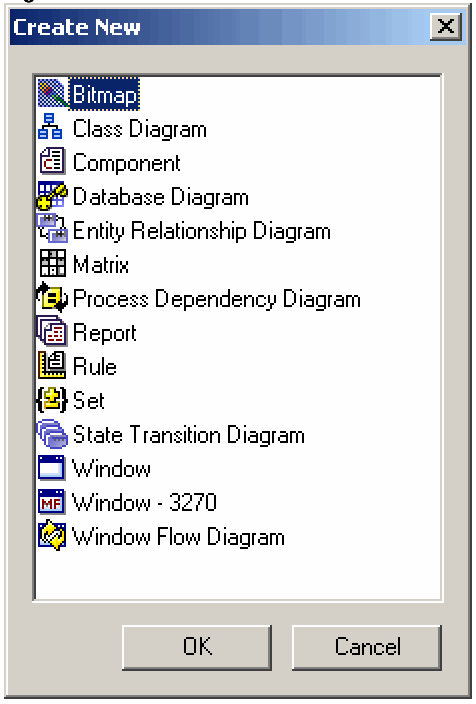
Early in the software development process, you specify data requirements for the system you are building by using data modeling. One part of data modeling is creating a database diagram. The tool that helps you create a database diagram in the Construction Workbench is also called the Database Diagram (DBD).
Tasks include:

- Creating or Opening a Database Diagram
- Inserting Database Objects
- Adding Columns to Tables and Keys
- Denormalizing a Database

Creating or Opening a Database Diagram

Creating a New Database Diagram

To create a new database diagram, complete the following steps:

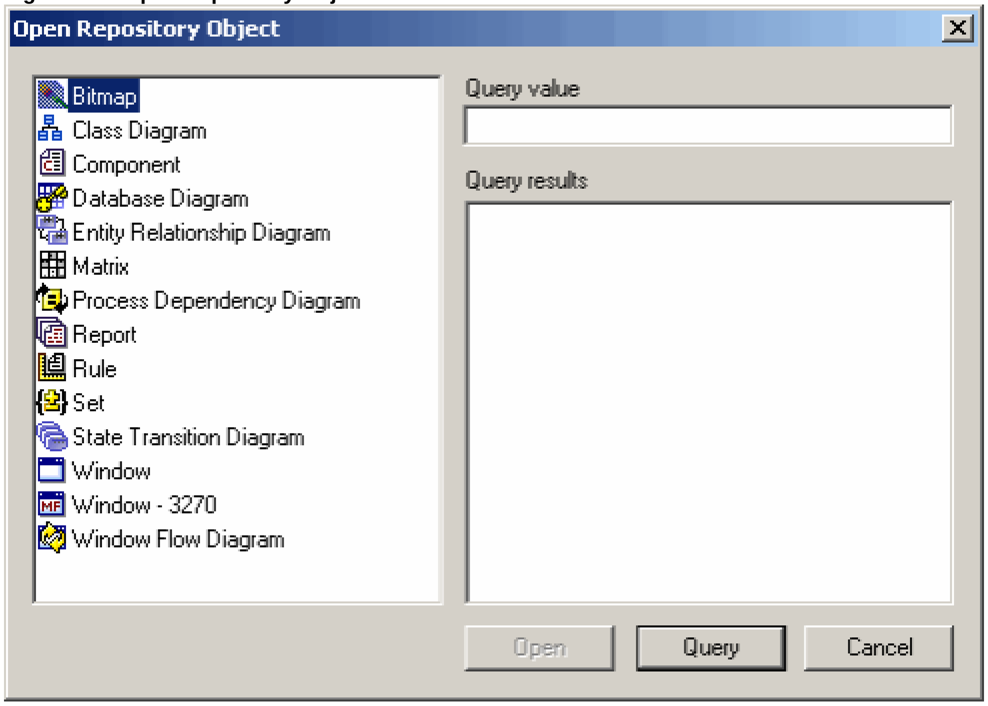
1. From the Construction Workbench menu, click **File > New**.
   The Create New window displays.
   Figure 8-15 Create New window
   

2. Select **Database Diagram**.
3. Click **OK**.

An empty Database Diagram displays in the work area.

Opening an Existing Database Diagram

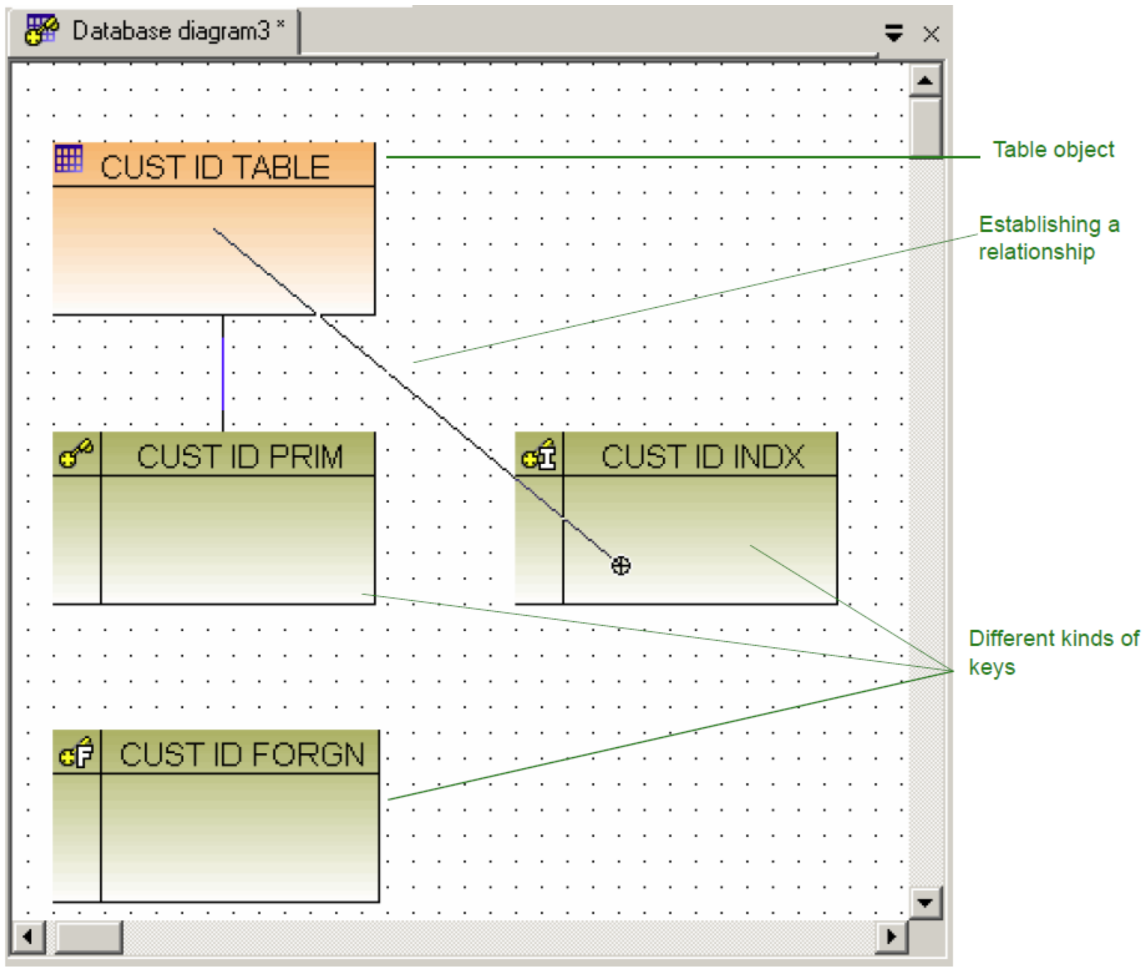
To use an existing diagram from the repository, complete the following steps:

1. From the Construction Workbench menu, click **File > Open**.
   The Open Repository Object window displays.
   Figure 8-16 Open Repository Object window
   

2. Select **Database Diagram**.
3. Click the **Query** button to display a list of Database Diagrams in the repository.
4. Click to highlight the diagram you want to use, and click **Open** button.

The diagram displays in the work area.
The toolbar associated with the Database Diagram also displays in the toolbar area. A sample Database Diagram is shown in Database diagram.

Figure 8-17 Database diagram



Inserting Database Objects

You can add or insert an object using the *Insert* menu in the Construction Workbench or the Database Diagram (DBD) toolbar (see Project Toolbar) when the DBD is displayed in the Construction Workbench Work Area.
You can insert the following objects into the Database Diagram:

- Table
- Key (primary, index, and foreign)
- Table has Key
- Key refers-to Table
- Note (see Adding a Note in a Diagram)

If you have created tables by forward engineering an Entity Relationship Diagram (ERD), complete the following steps below to view the tables in the database diagram:

1. Click the **Table** button in the DBD toolbar or select **Insert > Table** from the Workstation Workbench menu.
2. Click inside the DBD where you want to place the Table object.
   The Table displays as a gray box until you define the Table.
3. Double-click the Table.
   The **Insert Table** dialog displays.
4. Click the **Query** button to display a list of tables in the repository, select the table name and click **Insert**.
5. Repeat steps 1 through 3 to insert all the tables created by forward engineering an ERD.
6. Press F8 to display the keys and relationships between tables that forward engineering has created.

For more information about adding objects to a diagram, refer to Adding an Object to a Diagram.
While you use the DBD to place the tables, keys, and relationships in a diagram, you use the Hierarchy window to add columns to a Table or a Key. Refer to Adding Columns to Tables and Keys for how to create columns.

Table has Key

The Table has Key relationship is a line object that attaches a key to a table.

Key refers-to Table

The Key refers-to Table relationship is a line object that relates a key to a table. This relationship can be used to enforce referential integrity. All columns in a foreign key should have corresponding columns in the primary key of the table to which the foreign key refers.

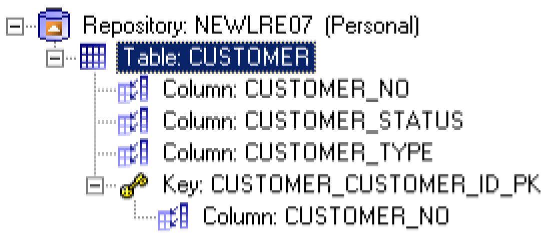
Adding Columns to Tables and Keys

While you use the Database Diagram to place the tables, keys and relationships in a diagram, you use the Hierarchy window for adding columns to those tables and keys. If you forward engineered an ERD to create tables and keys, the columns are created by the forward engineering process.

To add a Column to a Table or a Key, complete the following steps:

1. Click to select one or more objects in the DBD. Hold down Ctrl key while clicking to select multiple objects.
2. Select **Edit > Open as Hierarchy** from the Construction Workbench menu or right-click the object and select **Open as hierarchy** from the pop-up menu.
   The selected object displays in the Repository tab of the Hierarchy window (sample hierarchy is shown in Hierarchy of database tables).
3. To insert a Column, do one of the following:
   - Click to select the table or the key in the Repository tab of the Hierarchy window and from the Construction Workbench menu, select **Insert Child > Column**.
4. Right-click the object in the Repository tab of the Hierarchy window, select **Insert Child** from the pop-up menu, and select **Column**.
5. Display the Object Property window for the Column object and set properties information if necessary.

Figure 8-18 Hierarchy of database tables


Repository: NEWLRE07 (Personal)
Table: CUSTOMER
Column: CUSTOMER_NO
Column: CUSTOMER_STATUS
Column: CUSTOMER_TYPE
Key: CUSTOMER_CUSTOMER_ID_PK
Column: CUSTOMER_NO

Denormalizing a Database

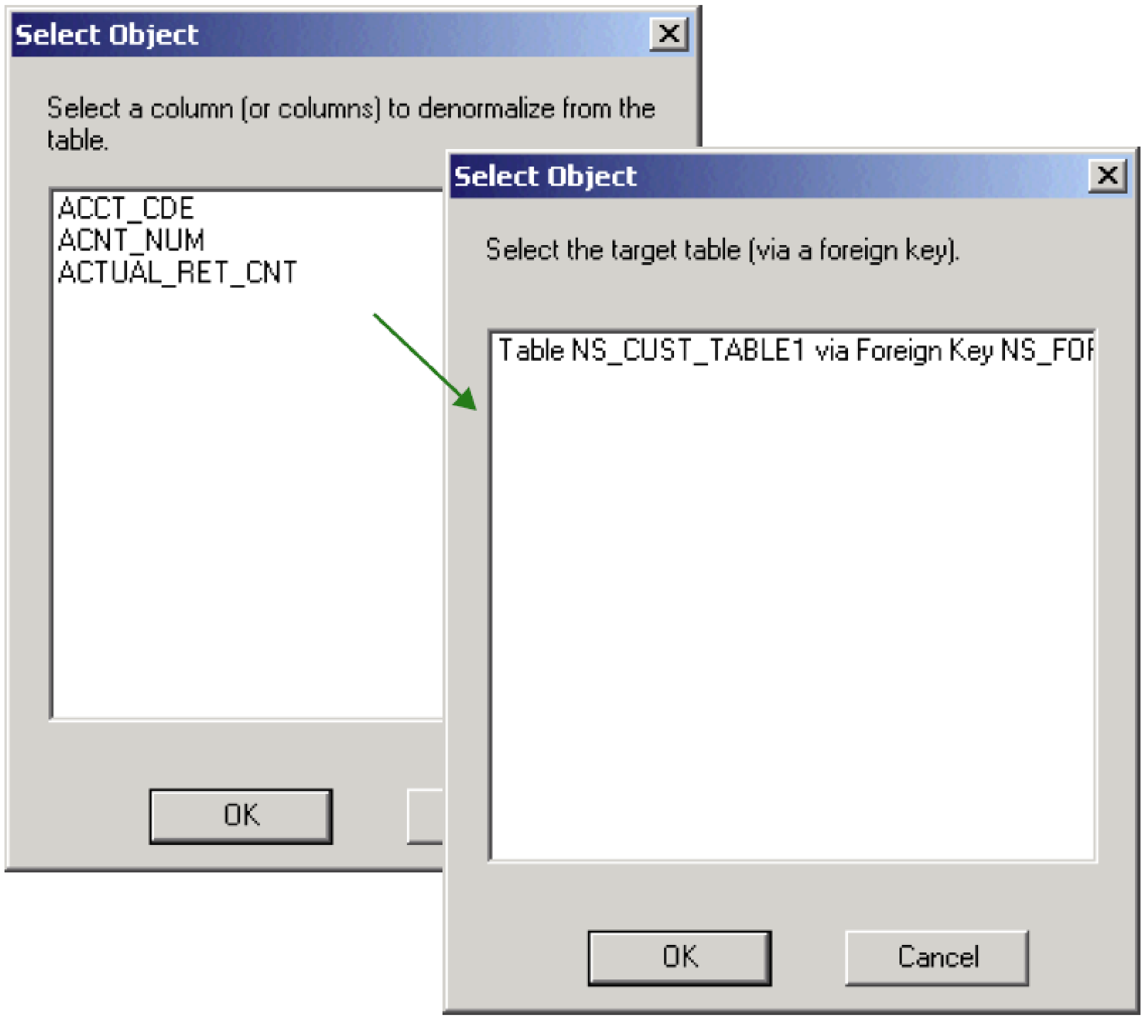
You can use denormalization to copy a column from one table to another through a foreign key. This improves performance by eliminating the need for table joins while executing SQL queries. You can use the denormalizing process in a database diagram to maintain traceability information and create physical columns in a table through inheritance from other tables.

For example, assume a logical model where an employee works for a department. When the ERD is forward engineered, the Department Number is propagated into the Employee table. However, the database administrator (DBA) might recognize that the Department Name should also be included with the Department Number.

Obtaining the name of the employee's department requires a table join unless the database is denormalized. To avoid the table join and thus speed database operations, the DBA can denormalize the model by including the Department Name column in the Employee table. Denormalization copies the Department Name column from the Department table into the Employee table and associates them with the Works_For foreign key.
To denormalize a table (or tables), complete the following steps:

1. In a database diagram, select the table (or tables) from which you want to denormalize.
2. Select **Analysis > Denormalize**.
   A Select Object dialog displays.

Figure 8-19 Select object for denormalize


Select Object
Select a column (or columns) to denormalize from the table.
ACCT_CDE
ACNT_NUM
ACTUAL_RET_CNT
OK
Select Object
Select the target table (via a foreign key).
Table NS_CUST_TABLE1 via Foreign Key NS_FOF
OK
Cancel

3. Select a column or columns to denormalize and click **OK**.
   The same window asks you to select the target table.
4. Select the target table and click **OK**.
   A message tells you the denormalize process completed.

Denormalizing might take some time because the process updates traceability information to reflect the columns copied to another table. Suppose you want to see who made a reservation for a customer. The RESERVATION table currently has only the agent's ID number as a foreign key. To have the agent's last name appear in the RESERVATION table without joining it with the AGENT table, denormalize the database by copying the AGENT_LAST_NAME column to the RESERVATION table.

Select the AGENT table and then run the denormalize function as described above. Select the AGENT_LAST_NAME column from the list that displays the source table's column names. The list displays only the columns the source table does not already have in common with the target table. Another window displays the names of the tables that have a foreign key from the source table. You can select the RESERVATION table and then choose **Analysis > Reverse trace**.

Process Dependency Diagram

In systems development, a **process** is one or more tasks that a system performs. Processes must often be performed in a certain order because the output of one process is the input to another. This is described as dependency. As you design the processes in an application, you reach a point where you must identify the processes and how they relate to each other. You must specify when in the application they are performed and what are the prerequisite tasks. You must also identify the data used. Use the Process Dependency Diagram (PDD) to specify the logical structure of the processes in an application.

This section discusses the following:

- Creating or Opening the Process Dependency Diagram
- Inserting Objects into the Process Dependency Diagram

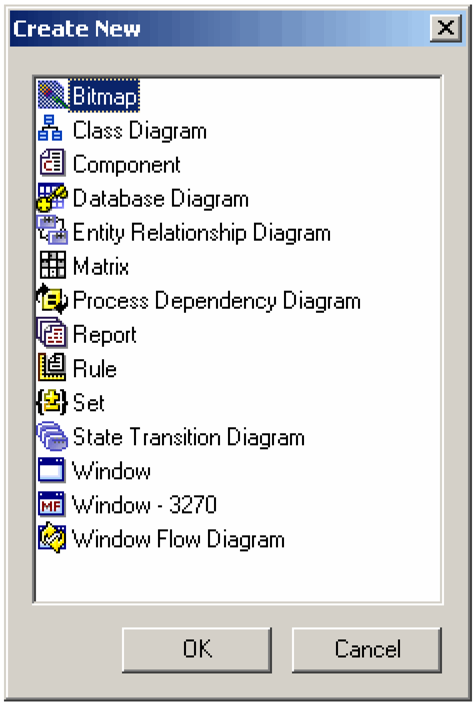
Creating or Opening the Process Dependency Diagram

Creating a New Process Dependency Diagram

To create a new Process Dependency Diagram, complete the following steps:

1. From the Construction Workbench menu, click **File > New**.
   The Create New window displays.

Figure 8-20 Create New window



2. Select **Process Dependency Diagram**.
3. Click **OK**.

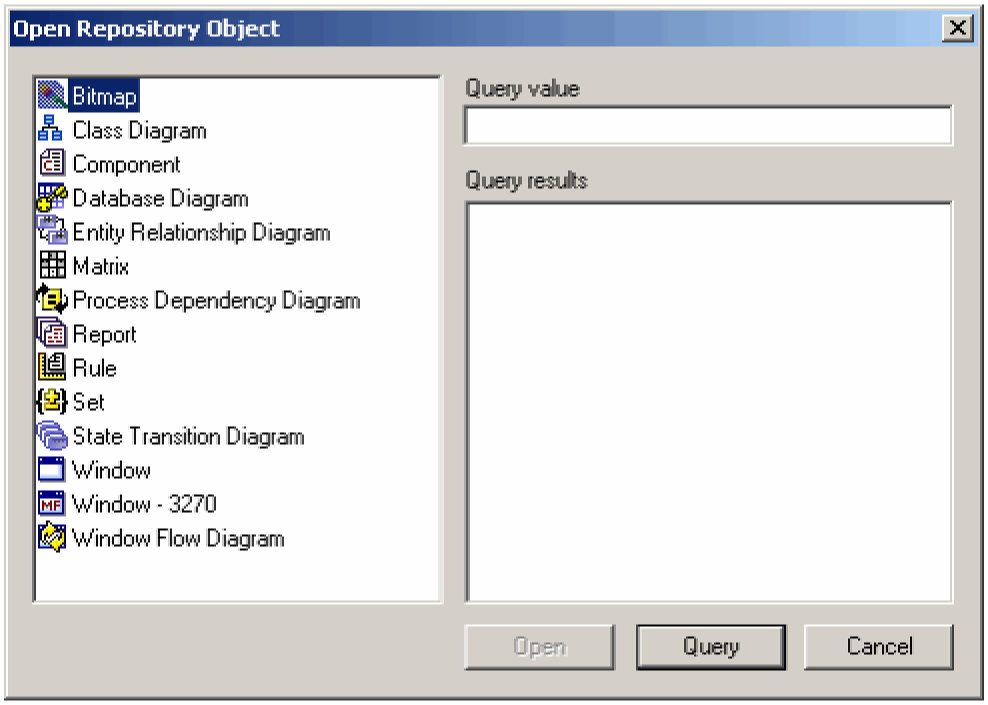
An empty Process Dependency Diagram is displayed in the Work Area.

Opening an Existing Process Dependency Diagram

To use a diagram from the repository, complete the following steps:

1. From the Construction Workbench menu, click **File > Open**.
   The Open Repository Object window displays.

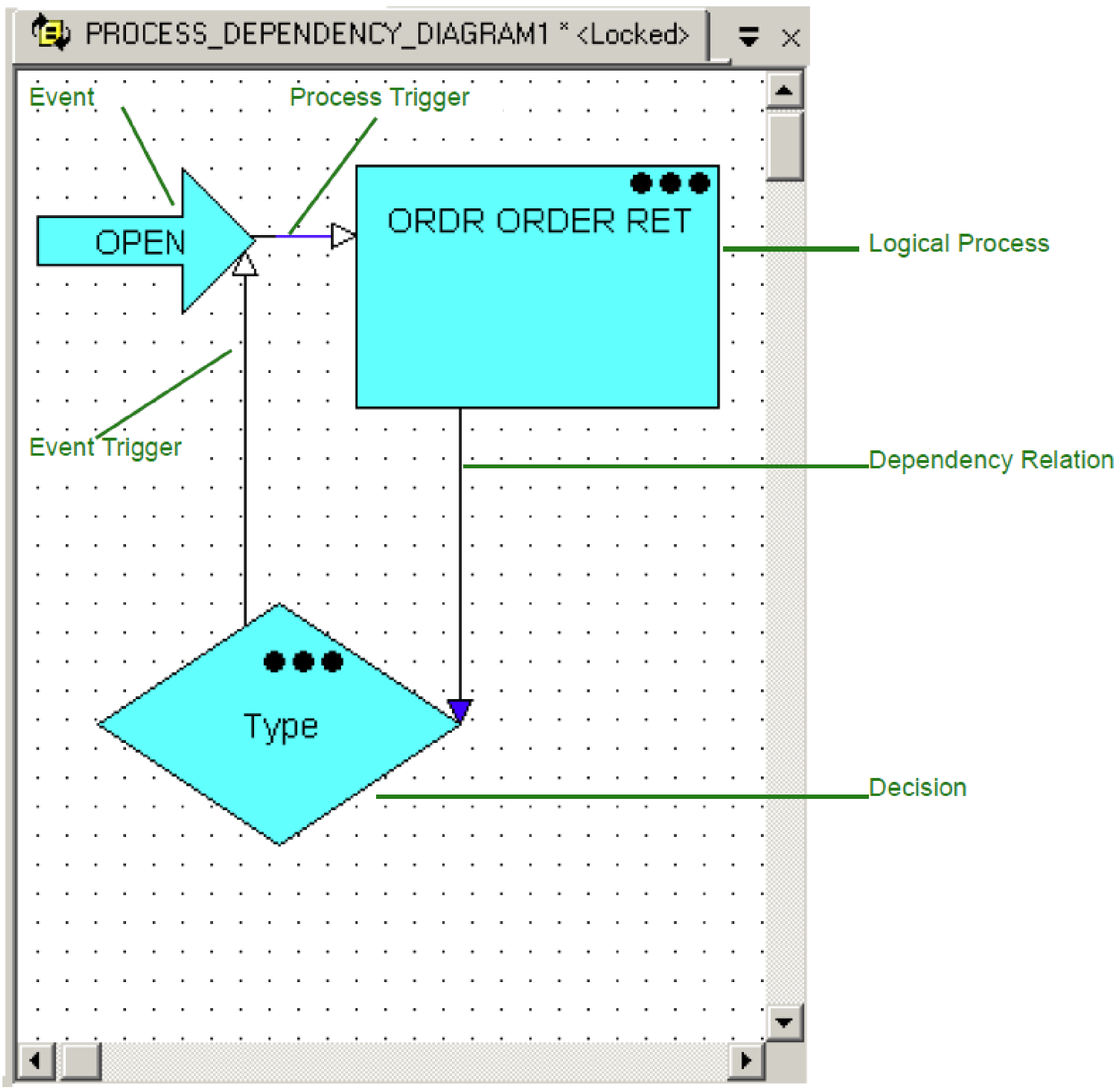
Figure 8-21 Open Repository Object window



2. Select **Process Dependency Diagram**.
3. Click the **Query** button to display a list of Process Dependency Diagrams in the repository.
4. Click to highlight the diagram you want to use and click **Open** button.

The diagram displays in the work area.
The toolbar associated with the Process Dependency Diagram also displays in the toolbar area. A sample Process Dependency Diagram is shown in Process Dependency Diagram Example.

Figure 8-22 Process Dependency Diagram Example



The label **Type** displayed for the Decision is the description of the decision, not the name of the Decision. To view the name of the Decision, right-click the Decision object and Select **Properties** or select **Open As Hierarchy**. The name of the Decision is displayed in the hierarchy.

Inserting Objects into the Process Dependency Diagram

You can add or insert an object using the *Insert* menu in the Construction Workbench or the Process Dependency Diagram toolbar (see Process Dependency Diagram Toolbar) when the Process Dependency Diagram (PDD) displays in the Construction Workbench Work Area.
You can insert the following objects into the PDD:

- Event
- Process
- Decision
- Connector (Process Trigger, Dependency, Event Trigger)
- Note (see Adding a Note in a Diagram)

To insert objects, complete the following steps:

1. Click the **Event**, **Process**, or **Decision** button in the Process Dependency Diagram toolbar or select the option from the **Insert** menu in the Construction Workbench.
2. Click inside the Process Dependency Diagram where you want to place the object.
   An object is displayed as a gray box until you define the object.
3. Double-click the object.
   The **Insert** dialog displays.
4. Click the **Query** button to display a list of objects in the repository or type a new object name and click **Insert**.

For more information about adding objects to a diagram, refer to Adding an Object to a Diagram.

Connector

A Connector is a line object used in the Process Dependency Diagram whose meaning is determined by the objects it connects. It represents a **Dependency** when it connects logical or decision processes. It represents an **Event Trigger** when it connects a process and an event. It represents a **Process Trigger** when it connects an event and a process or decision.

State Transition Diagram

A State Transition Diagram identifies the internal and external events to which an application must respond and the various states of the data that result. Transitions between states help outline the business rules an application needs.

This section discusses the following:

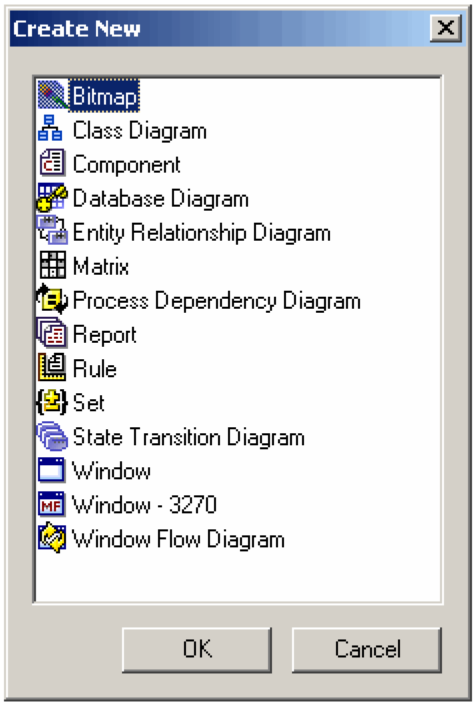
- Creating or Opening the State Transition Diagram
- Inserting Objects into the State Transition Diagram

Creating or Opening the State Transition Diagram

Creating a New State Transition Diagram

To create a new diagram, complete the following steps:
#From the Construction Workbench menu, click **File > New**.
The Create New window displays.

Figure 8-23 Create New window



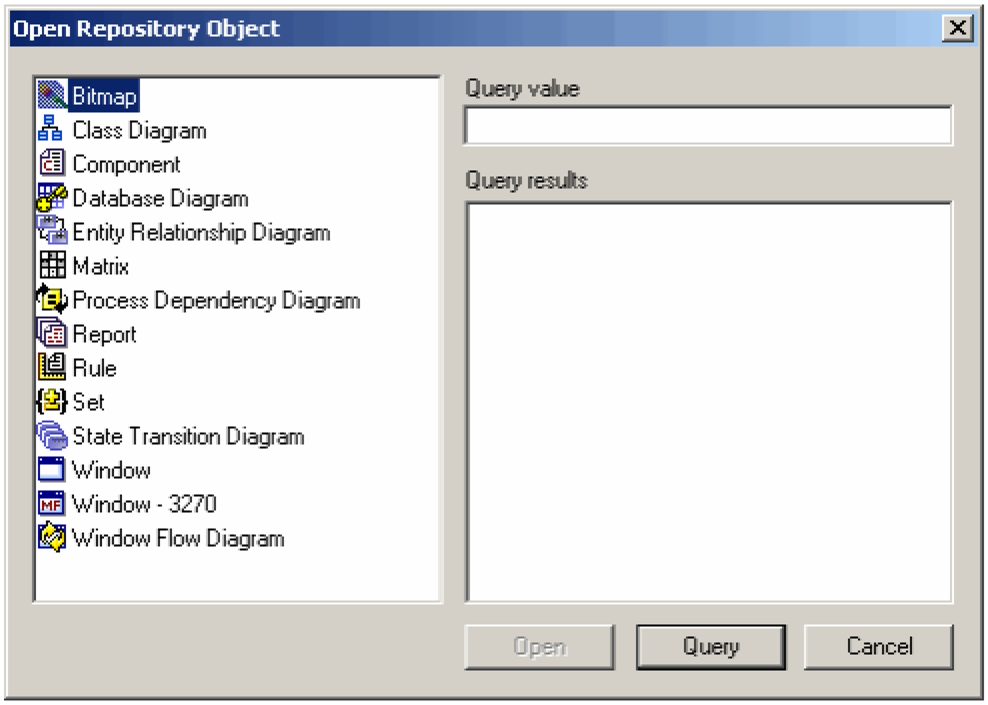
1. Select **State Transition Diagram** and click **OK**.
   An empty State Transition Diagram displays in the work area.

Opening an Existing State Transition Diagram

To use a diagram from the repository, complete the following steps:

1. From the Construction Workbench menu, click **File > Open**.
   The Open Repository Object window displays.

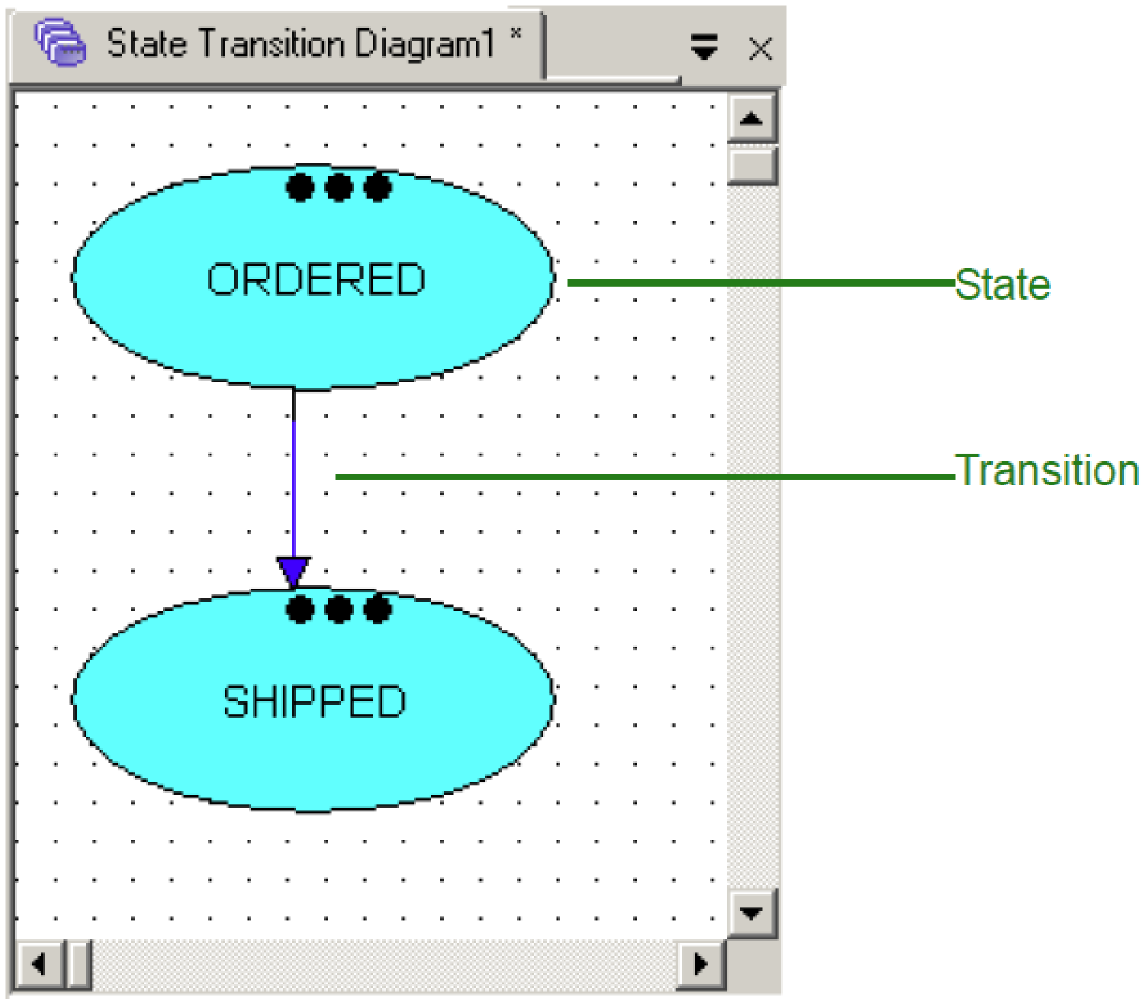
   Figure 8-24 Open Repository Object window



2. Select **State Transition Diagram**.
3. Click the **Query** button to display a list of State Transition Diagrams in the repository.
4. Click to highlight the diagram you want to use and click **Open** button. The diagram displays in the work area.

The toolbar associated with the State Transition Diagram is also displayed in the toolbar area. A sample State Transition Diagram is shown in State Transition Diagram window Example.

Figure 8-25 State Transition Diagram window Example



Inserting Objects into the State Transition Diagram

You can add or insert an object using the *Insert* menu in the Construction Workbench or the State Transition Diagram toolbar (see State Transition Diagram Toolbar) once the State Transition Diagram is displayed in the Construction Workbench Work Area.

You can insert the following objects into the State Transition Diagram:

- State
- Event
- Transition
- Note (see Adding a Note in a Diagram)

To insert objects, complete the following steps:

1. Click the **State** or **Event** button in the State Transition Diagram toolbar or select from the menu **Insert** in the Construction Workbench.
2. Click inside the State Transition Diagram where you want to place the object.
   An object is displayed as a gray box until you define the object.
3. Double-click the object.
   The **Insert** dialog displays.
4. Click the **Query** button to display a list of objects in the repository or type a new object name and click **Insert**.
5. Add a Transition connector object to connect the State objects.
   You can connect an Event object to the Transition.

For more information about adding objects to a diagram, refer to Adding an Object to a Diagram.

Window Flow Diagram

The Window Flow Diagram is a tool that can be used to prototype and simulate the sequence or flow of windows.

This section discusses the following:

- Creating or Opening the Window Flow Diagram

- Inserting Objects into the Window Flow Diagram

Refer to the *Developing Applications Guide* for more information about how to use the Window Flow Diagram during the application design phase.

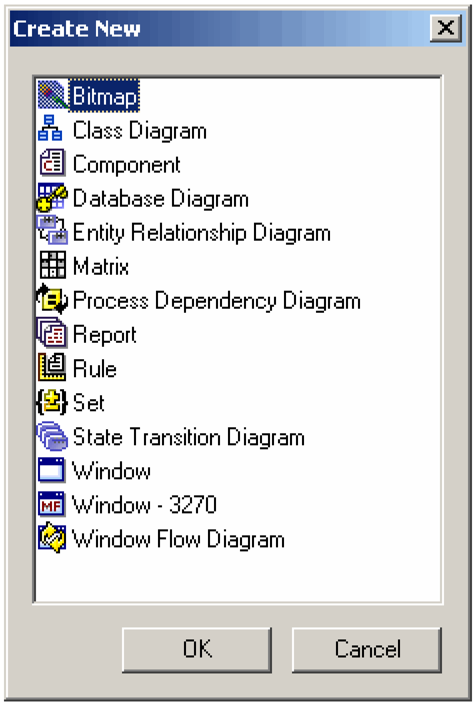
Creating or Opening the Window Flow Diagram

Creating a New Window Flow Diagram

To create a new diagram, complete the following steps:

1. From the Construction Workbench menu, click **File > New**.
   The Create New window displays.

Figure 8-26 Create New window



2. Select **Window Flow Diagram**.
3. Click **OK**.

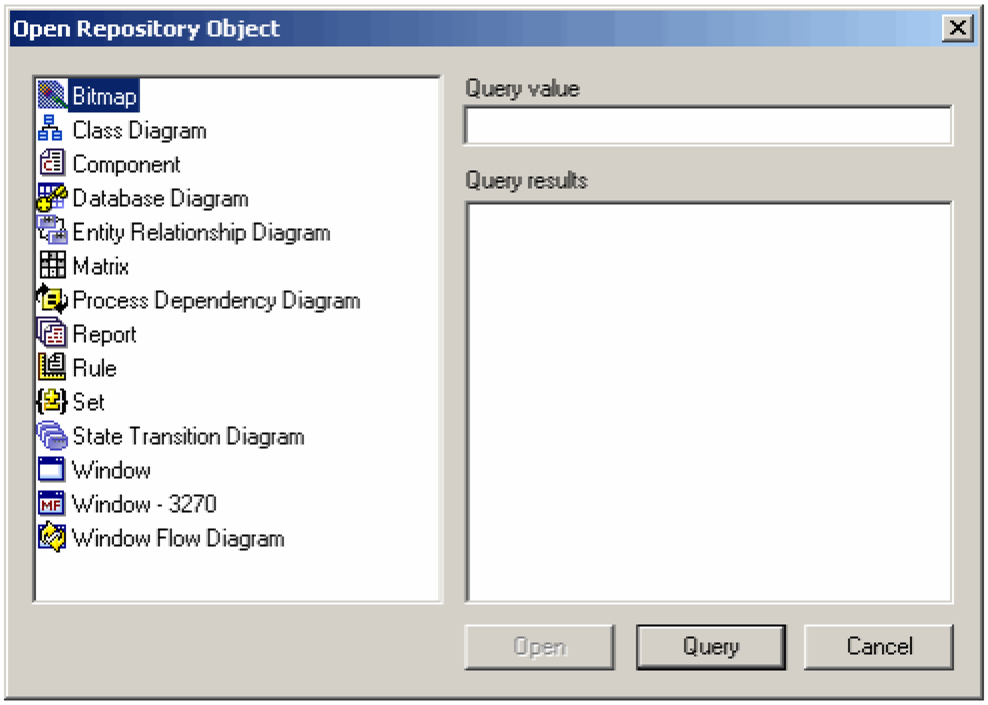
An empty Window Flow Diagram displays in the work area.

Opening an Existing Window Flow Diagram

To use a diagram from the repository, complete the following steps:

1. From the Construction Workbench menu, click **File > Open**.
   The Open Repository Object window displays.

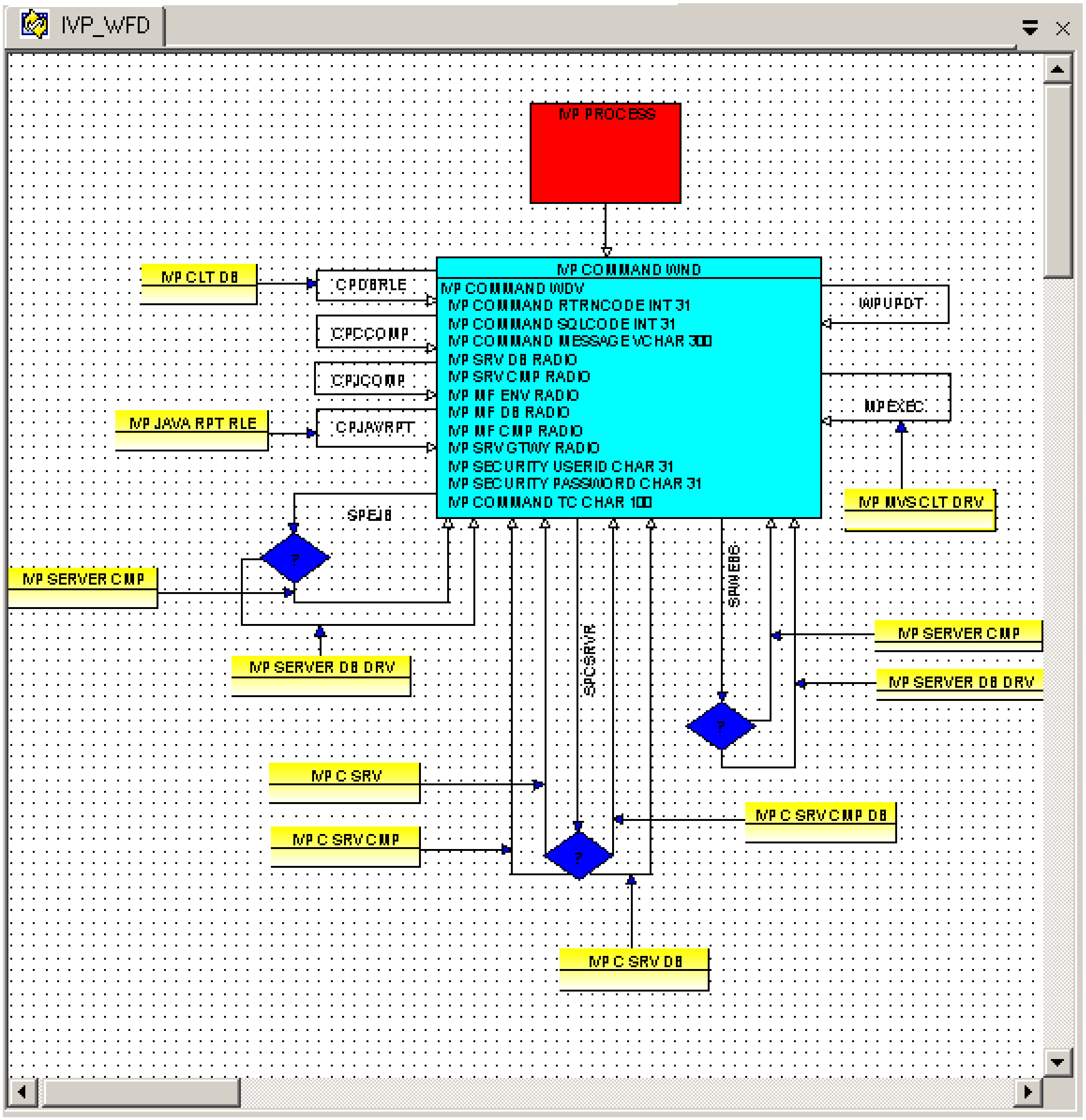
Figure 8-27 Open Repository Object window



2. Select **Window Flow Diagram**.
3. Click the **Query** button to display a list of Window Flow Diagrams in the repository.
4. Click to highlight the diagram you want to use and click the **Open** button.
   The diagram is displayed in the work area.

The toolbar associated with the Window Flow Diagram is also displayed in the toolbar area. A sample Window Flow Diagram is shown in Window Flow Diagram example.

Figure 8-28 Window Flow Diagram example



Inserting Objects into the Window Flow Diagram

You can add or insert an object using the *Insert* menu in the Construction Workbench or the Window Flow Diagram toolbar (see Window Flow Diagram Toolbar) once the Window Flow Diagram (WFD) is displayed in the Construction Workbench Work Area.

You can insert the following objects into the Window Flow Diagram:

- Process (in drawing tools)
- Window (in Window Flow Diagram)
- Decision
- Terminal
- Entry Point
- Dialog Unit
- Rule (in Window Flow Diagram)
- Flow (normal, nested, or detached)
- Note (see Adding a Note in a Diagram)

To insert objects, complete the following steps:

1. Click one of the objects button in the WFD toolbar or select from the menu *Insert* in the Construction Workbench.
2. Click inside the WFD where you want to place the object.
   An object is displayed as a gray box until you define the object.
3. Double-click the object.
   The **Insert** dialog displays.
4. Click the **Query** button to display a list of objects in the repository or type a new object name and click **Insert**.

5. Add a flow object to connect different objects.

For more information about adding objects to a diagram, refer to Adding an Object to a Diagram. For an example tutorial for Window Flow Diagram, refer to the *Developing Applications Guide*.

Process (in drawing tools)

A Process is a box indicating an activity that comprises logical units of work. Each process represents a single application (leaf process) or a set of applications.

Window (in Window Flow Diagram)

A Window is a logical representation in a repository of a window the user sees on a workstation or 3270 terminal display.

Decision

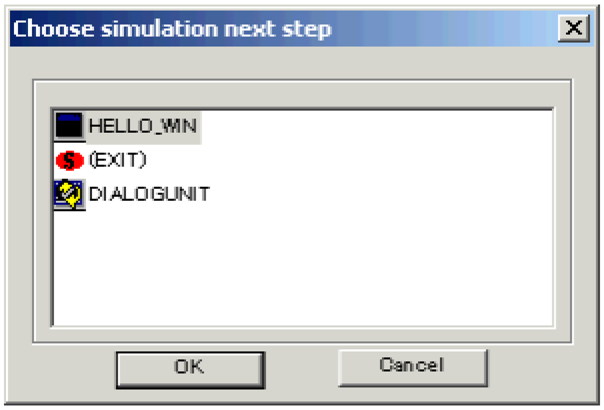
A Decision is a representation of a logical decision that should be made after the execution passes a flow.

A Decision can have an incoming **Flow** connector and might have multiple exit points. The connector types that can be used out from the Decision are **Flow**, **Nested Flow** or **Detached Flow**. The Decision in a Window Flow Diagram drawing is only a graphical representation with no corresponding object in the repository and is displayed as a diamond shape with a question mark inside. On the other hand, you can type text in the **Description** attribute on the Decision's properties dialog to be displayed inside the Decision in a Process Dependency Diagram drawing.

In the window flow simulation, if a Decision is selected when starting a window flow simulation, the decision becomes the starting point for the simulation.

- If the Decision has no exit point, the current part of the simulation terminates.
- If the Decision has only one exit point, the flow continues automatically to that point.
- If the Decision has multiple exit points, a dialog is displayed to choose the next step as shown in Window flow simulation next step dialog.

Figure 8-29 Window flow simulation next step dialog


Choose simulation next step

HELLO_WIN
(EXIT)
DIALOGUNIT

Terminal

A Terminal indicates an exit from the application or process.

Entry Point

An Entry Point represents the starting rule for a Dialog Unit. It becomes the starting point of the window flow simulation if no Window, Dialog Unit, or Decision is selected. The Entry Point is linked to a Rule object in the repository, which has the value of the Description property as "DIALOG UNIT". The Rule can be used to start simulation from another Window Flow Diagram or a Dialog Unit. You can only exit from an Entry Point with a *Flow* connector that can be connected to a Window, Dialog Unit, Decision or Exit point.

Dialog Unit

A Dialog Unit represents a way of executing a set of windows that performs a specific function. You can use a Dialog Unit to modularize an application so that you can reuse the same windows and flows in different drawings. A Dialog Unit is similar to any other Window Flow Diagram, except that it must be started with a Rule as the entry point, instead of a process.

A Dialog Unit is represented in the repository as a RULE object with Description property as "DIALOG UNIT" (same as for Entry Point). When querying for existing Dialog Unit in the repository, the query box displays only the rules that have a "DIALOG UNIT" description.

After inserting the Dialog Unit symbol onto the drawing, double-click the symbol to choose the starting Rule. Then, right-click the symbol and choose **Navigate** to select a drawing. When the window flow simulation reaches the Dialog Unit object, it loads the drawing that is associated with the object and starts its simulation.

Rule (in Window Flow Diagram)

A Rule in the Window Flow Diagram is only a representation of an execution of a Rule along with a Flow (normal, nested or detached), and does not affect the window flow simulation. When you insert a Rule in the Window Flow Diagram drawing, a line that follows the mouse movement displays. Choose a flow to be associated with the current Rule by clicking on the flow. Then, double-click the Rule object to link to a Rule in the repository. You can only associate one Rule to a single flow. The Rule is deleted when the associated flow is deleted.

Flow

A Flow line indicates that an action taken on the source window opens the target window and closes the source window. You can label a Flow to indicate what action causes the transition between windows, and you can attach a Rule to a flow to indicate the rule that governs the transition.

A Nested Flow line indicates that an action taken on the source window opens the target window on top of the source window. You can label a Nested Flow to indicate what action causes the transition between windows, and you can attach a Rule to a flow to indicate the rule that governs the transition. When you close a nested window, you automatically return to its source window.

In a Window Flow Diagram, the user cannot add or edit labels to links from decisions. However, when the text for a label on a link from a window is other than an HPSID, then the simulation will no longer work. The label text must be changed back to an HPSID for simulation to work.

Class Diagram

A Class Diagram is the most common diagram for modeling an object-oriented system. It represents the static design view of the system.
The class diagram shows a set of classes, interfaces, structures, exceptions, services and their relationships which model the data and the behavior of the system.

Use the class diagrams to:

- Model the vocabulary of a system, specifying the abstractions and their responsibilities
- Model simple collaborations, like collections of classes and their relationships
- Model database schema.

Class diagrams cannot be forward engineered or used to generate other artifacts in the process of application development.
Tasks include:

- Creating or Opening a Class Diagram
- Inserting Objects into the Class Diagram

Creating or Opening a Class Diagram

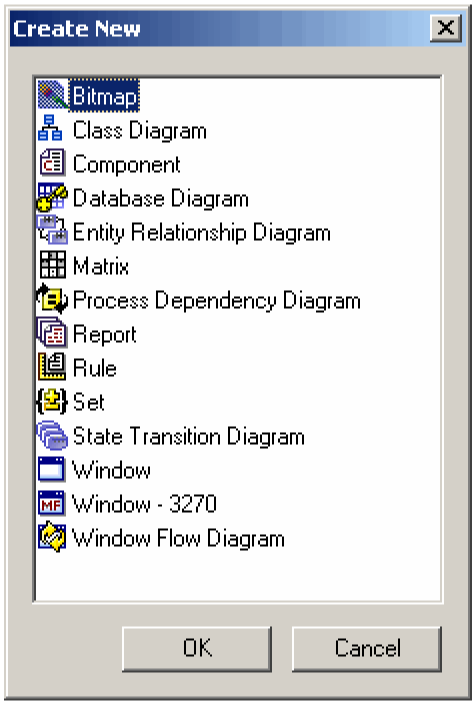
You can create a new class diagram or open one from the repository. When a class diagram is created or opened, the diagram is displayed in the Work Area of the Construction Workbench.

Creating a New Class Diagram

To create a new class diagram, complete the following steps:

1. From the Construction Workbench menu, click **File > New**.
   The Create New window displays.

   Figure 8-30 Create New window



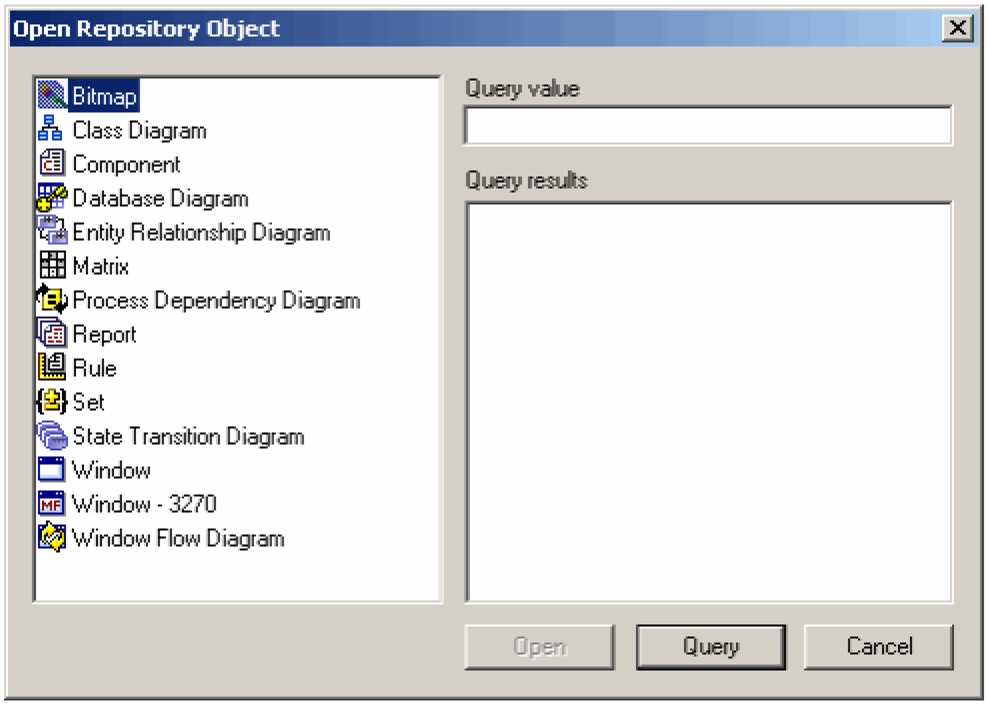
2. Select **Class Diagram**.
3. Click **OK**.
   An empty Class Diagram displays in the Work Area.

Opening an Existing Class Diagram

To use a class diagram from the repository, complete the following steps:

1. From the Construction Workbench menu, click **File > Open**.
   The Open Repository Object window displays.

Figure 8-31 Open Repository Object window

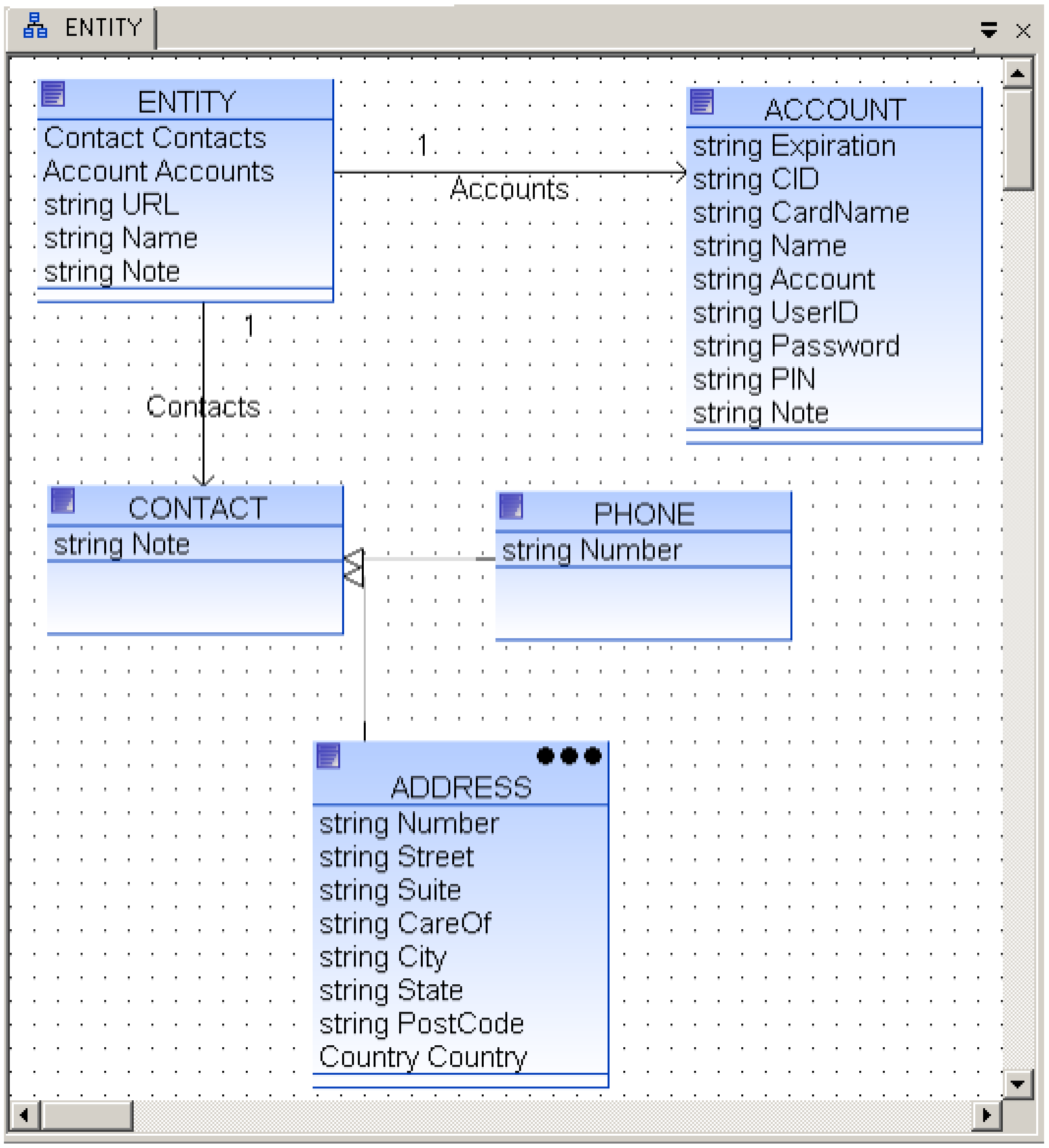


2. Select **Class Diagram**.
3. Click **Query** button to display a list of Class Diagrams in the repository.

4. Click to highlight the diagram you want to use and click the **Open** button.

The diagram displays in the work area.
The toolbar associated with the Class Diagram is also displayed in the toolbar area. A sample Class Diagram is shown in Class Diagram sample:

Figure 8-32 Class Diagram sample



Inserting Objects into the Class Diagram

When a class diagram is displayed in the Construction Workbench Work area, you can insert classifiers, such as classes, interfaces, structures, exceptions, services, maps, enumerations, tables, rules, or relations or collaboration, such as Generalize Class, Realization, Generalize Interface, Uses Service, Uses, Association, Uses Table, Uses Rule, Array, No Array. To add or insert a classifier into the class diagram, use the *Insert* menu in the Construction Workbench or the Class Diagram toolbar (see Class Diagram Toolbar). You can also drag and drop from the hierarchy to the class diagram.

You can insert the following items into the class diagram:

- Classifiers:
  - Class
  - Interface
  - Structure
  - Exception
  - Service
  - Map
  - Table
  - Rule

- Collaborations (relationships)
- Complex Associations
- Notes (see Adding a Note in a Diagram)

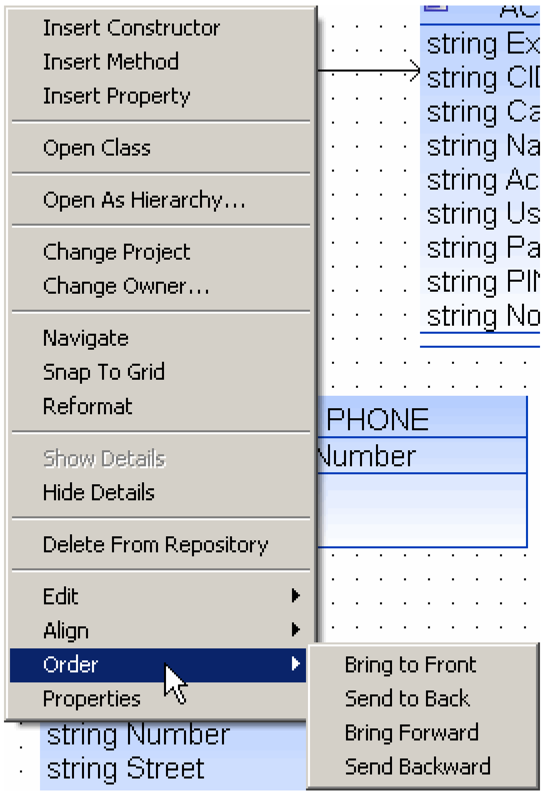
To insert an object, complete the following steps:

1. Click the corresponding object button in the Class Diagram toolbar or select it from the menu **Insert > Object**.
2. Click inside the class diagram where you want to place the object.
   The object is displayed as an empty box until you define it.
3. Double-click the object in the class diagram.
   The **Insert Object** dialog displays.
4. Click the **Query** button to display a list of objects in the repository or type a new object name and click **Insert**.

For more information about adding objects to a diagram, refer to Adding an Object to a Diagram.
You can also change where an item appears in the class diagram with relation to other items. You can bring items forward in the class diagram, or you can send them back in the class diagram. To change where an item appears in the class diagram, complete the following steps:

1. Right-click the item in the class diagram.
   A popup menu displays.

Figure 8-33 Order Submenu when working with a class diagram



2. Select one of the following:
   - Select **Order > Bring to Front** to put the selected item in front of all other items.
   - Select **Order > Send to Back** to put the selected item behind all other items.
   - Select **Order > Bring Forward** to move the selected item forward in front of one other item.
   - Select **Order > Send Backward** to move the selected item backward behind one other item.

Collaborations (relationships)

A Collaboration defines the interconnections between two classifiers. In the class Diagram, a collaboration can have two roles, "From" and "To". A recursive collaboration relates a classifier to itself. A classifier can have more than one collaboration with itself.

You can add the following collaborations:

- Extends Class
- Implements Interface
- Extends Interface
- Uses Service
- Uses
- Association

- Uses Table
- Uses Rule

To add a collaboration between classifiers, complete the following steps:

1. Click the corresponding collaboration button in the Class Diagram toolbar or select the collaboration from the **Insert** menu.
2. Click inside the classifier from which the collaboration starts, then click inside the classifier to which the collaboration ends.
3. Click the collaboration line to display the Object Property window for the collaboration. Specify the roles of the relationship.
4. The class diagram shows the role as a label for "From" relationship beside the cardinality it describes. You can display "From" only, or both "From" and "To", or no labels. See Modifying Aspects of a Diagram.

Cardinality

In the Class Diagram, cardinalities are displayed at each end of a collaboration line where the relationship meets an entity. They are also represented by the collaboration label.

To specify the cardinality of a collaboration, complete the following steps:

1. Select a cardinality type from the **Insert** menu or click one of the cardinality types on the Class Diagram toolbar.
2. Click where the relationship line intersects with the Entity to which you want to apply the cardinality type.

Complex Associations

Complex association can be:

- Array: it is used for an association between two classes. This will make the association become a array.
- No Array: if you apply it to the association, the Array will be changed to single association.

Working with Diagrams

Diagrams are essential outlines for successful projects. This section discusses controls and functions that are common to all drawing tools and diagrams.
The following topics provide information that applies to all the drawing tools:

- Display Options for Diagrams

Common tasks with diagrams include the following:

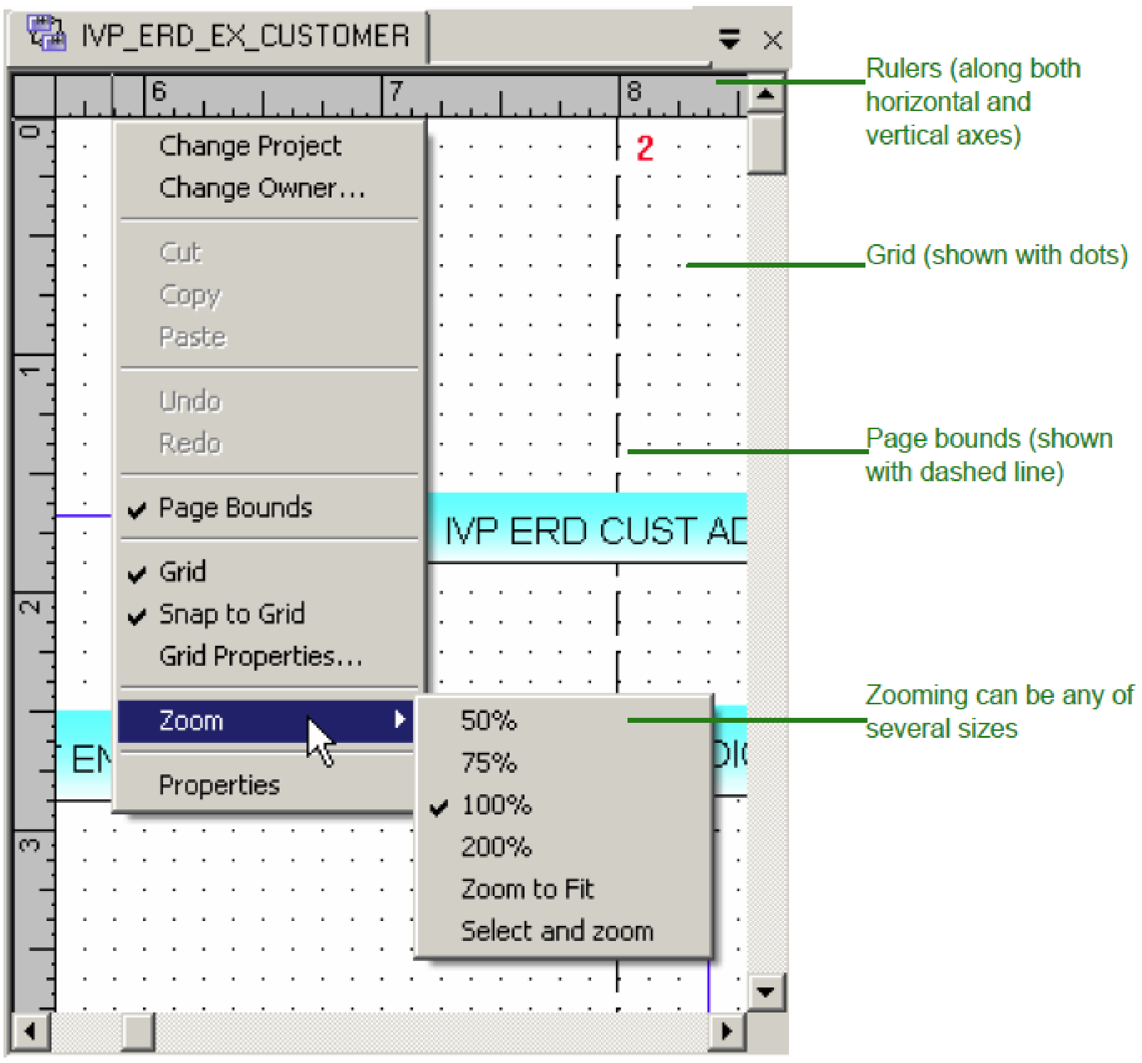
- Working with Objects in Diagrams
- Specifying Header or Footer
- Navigating to Another Diagram
- Modifying Aspects of a Diagram
- Showing More Information
- Changing Affiliation and Owner

Display Options for Diagrams

The following options to change the display of the diagram are available to all of the drawing tools. Refer to the following figure for a summary of the options, which include the following:

- Displaying a Grid
- Displaying Rulers
- Displaying Page Bounds
- Zooming In and Out

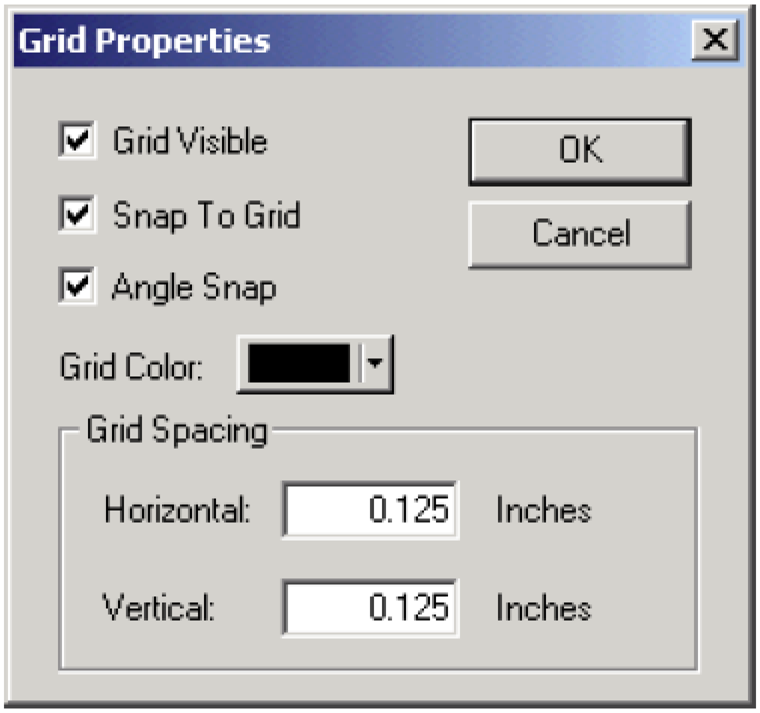
Figure 8-34 Display options



Displaying a Grid

Right-click in the diagram window (see Display options) to display a menu that includes the grid options. From this menu, check **Grid** to show the grid pattern and uncheck to hide the grid. The grid is intended to aid you in placing and arranging objects in the diagram.
You can also align the objects you place in the diagram with the grid. From the menu shown in Display options, select **Snap to Grid** to place objects only at grid points.
To change the appearance of the grid, select **Grid Properties**. The dialog shown in Grid Properties dialog displays.

Figure 8-35 Grid Properties dialog



☑ Grid Visible

☑ Snap To Grid

☑ Angle Snap

Grid Color: ■

Grid Spacing

Horizontal: 0.125 Inches

Vertical: 0.125 Inches

- **Grid Visible** - turn on or off the display of the grid.
- **Snap To Grid** - places objects only at grid points.
- **Angle Snap** - allow angles in the lines (relationship objects) in the diagram.

Select the color of the dots in the grid with the color selection pull-down. Light colors, like yellow, barely display because the dots are small.
Type the values in the Grid Spacing area to set the horizontal and vertical spacing of the grid dots.

Displaying Rulers

As shown in Display options, you can display rulers (both horizontal and vertical) around the outside of the diagram in the diagram window. From the Construction Workbench, click **View > Diagram Rulers** to turn on or off the display of the rulers. When the check mark is displayed in the menu, both rulers display.
The rulers display with the units of measurement specified in the Windows operating system for that workstation. The units are country code dependent; they might either be **U.S.** or **Metric** depending on what country or locale is set in the Control Panel. To check or change that value, look in the **Start > Control Panel > Regional and Language Options** (for Windows XP) or **Start > Settings > Control Panel > Regional Options** (for Windows 2000), the **Measurement system** value. For the U.S. setting, the rulers display in units of inches; for the Metric setting, the rulers display in units of centimeters.

Displaying Page Bounds

From the right-click menu shown in Display options, you can display the place in the diagram where the printer ends one page and begins another. These page bounds are shown with a dashed line that runs the length and height of the diagram. The page bounds are helpful if you have a multi-page diagram and you want to organize objects in such a way that they print clearly. From the right-click menu in the diagram window, select **Page Bounds** to turn on or off the display of the dashed boundary lines. When the check mark is displayed in the menu, the page bounds are displayed.
For other information regarding page printing, refer to Working with Objects in Diagrams.

Zooming In and Out

From the right-click menu shown in Display options, you can zoom in or zoom out on your diagram. Click **Zoom** and then select the zoom percentage you want. The **Zoom to Fit** option fits your diagram in the diagram window regardless of the size of your window. The Select and Zoom option allows you to zoom the area that you select.
You can also access zoom properties from the **View** menu of the Construction Workbench menu:

- **Zoom In** - Scales the diagram larger.
- **Zoom Out** - Scales the diagram smaller.
- **Show Normal Size** - Sets the zoom factor to 100%.
- **Show All** - Fits the diagram in the window regardless of the size of your window. **Show All** is the same as **Zoom to Fit** from the right-click menu.
- **Zoom to Selected Area** - Select this option and select a piece of your diagram. The tool zooms in on that selected portion of the diagram.

Working with Objects in Diagrams

This section discusses common things you do when working with diagrams and drawings:

- Adding an Object to a Diagram
- Adding a Note in a Diagram
- Moving an Object in a Diagram
- Resizing an Object in a Diagram
- Finding an Object in a Diagram
- Deleting an Object from the Diagram
- Copying and Pasting an Object in a Diagram
- Arranging Objects in a Diagram

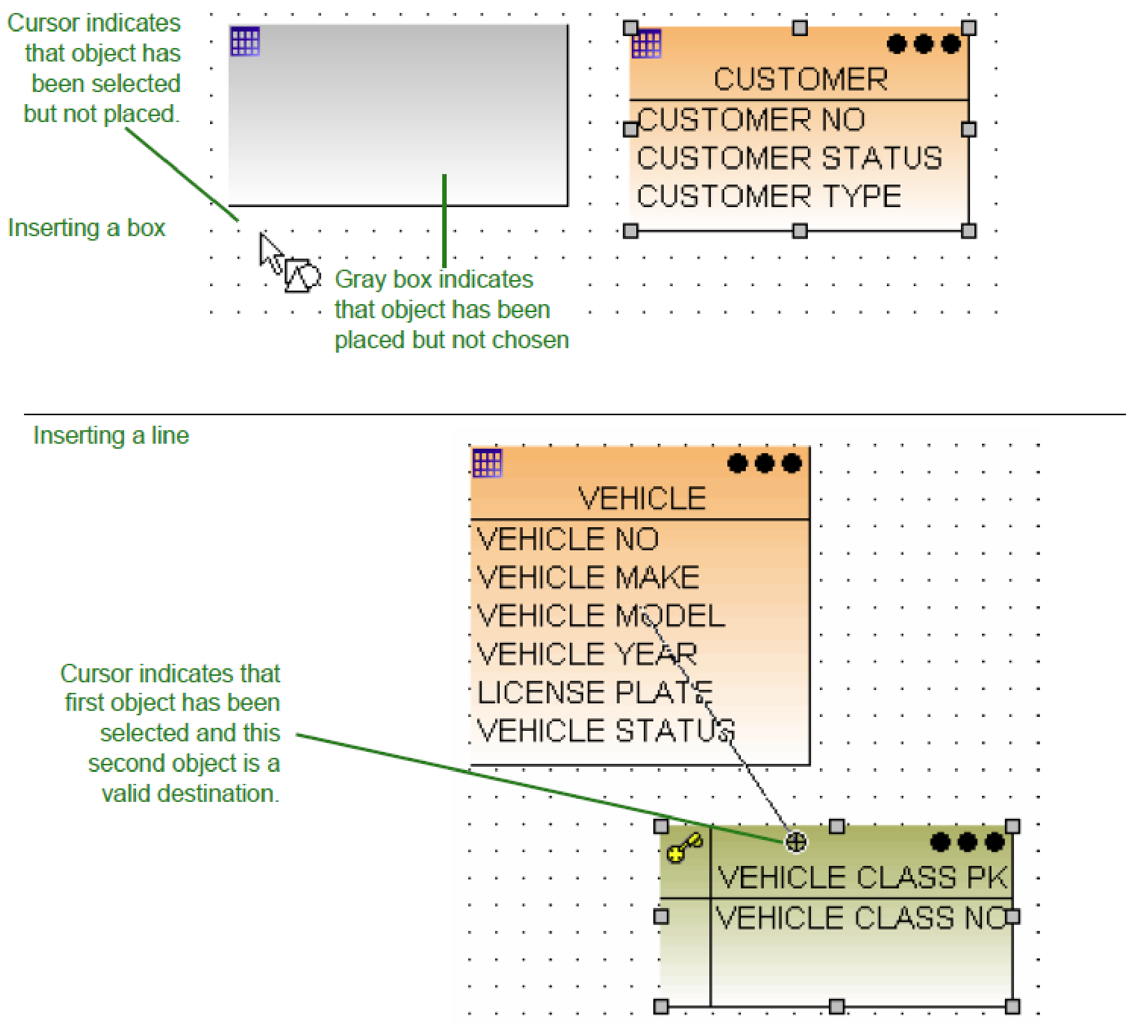
Adding an Object to a Diagram

To add an object to a diagram, do the following:

1. Open the diagram window in the Construction Workbench.
2. From the Diagram Toolbar, select an object to insert, or from the Construction Workbench menu bar, select **Insert**, then select the type of object to insert.

When inserting an object, the pointer changes until you click somewhere in the diagram window. Then a gray version of the entity is shown in the diagram until you either define that entity or select one from the repository. For a line, select the first object and the destination object and the system draws the line between them. Adding object (box and line) shows the stages of adding an object, for both a box and a line object. When done adding several objects, you might want to try Reformatting Objects.

Figure 8-36 Adding object (box and line)

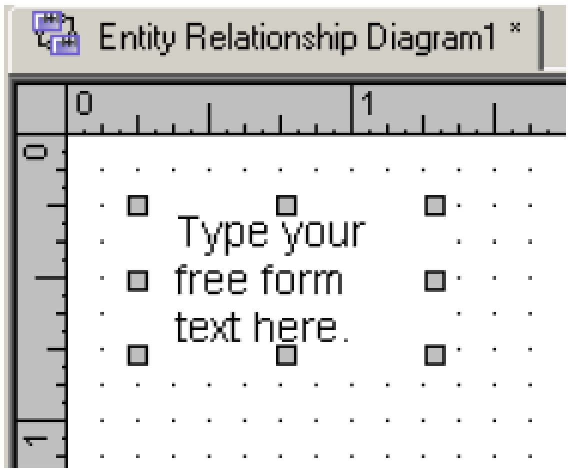

Cursor indicates that object has been selected but not placed.
Inserting a box
Gray box indicates that object has been placed but not chosen
CUSTOMER
CUSTOMER NO
CUSTOMER STATUS
CUSTOMER TYPE
Inserting a line
VEHICLE
VEHICLE NO
VEHICLE MAKE
VEHICLE MODEL
VEHICLE YEAR
LICENSE PLATE
VEHICLE STATUS
Cursor indicates that first object has been selected and this second object is a valid destination.
VEHICLE CLASS PK
VEHICLE CLASS NO

Select any object and drag a handle to a new location to resize the objects.

Adding a Note in a Diagram

To add a note of text in a diagram, from the Construction Workbench menu bar with the diagram window open, select **Insert > Note** or select the **Note** button from the **Diagram** toolbar and place it on the diagram. Click in the diagram; a text block that you can edit displays. You can turn on or off the border of the note in the Drawing Tools Options section of the Workbench Options. To get to the Drawing Tools Options, from the Construction Workbench menu bar, click **Tools > Workbench Options > Tools > Drawing Tools**.
Insert note text shows the placement of a note in a diagram. In this example, the border is on. The note is not an object in the repository; it is associated with the diagram to serve as a reminder or a label. It is displayed when printed.

Figure 8-37 Insert note text


Entity Relationship Diagram1 *
0
1
Type your free form text here.

Select the note and drag a handle to a new location to resize a note. Double-click the note to add some text, then enter the text in the Note Editor window and click **OK**.

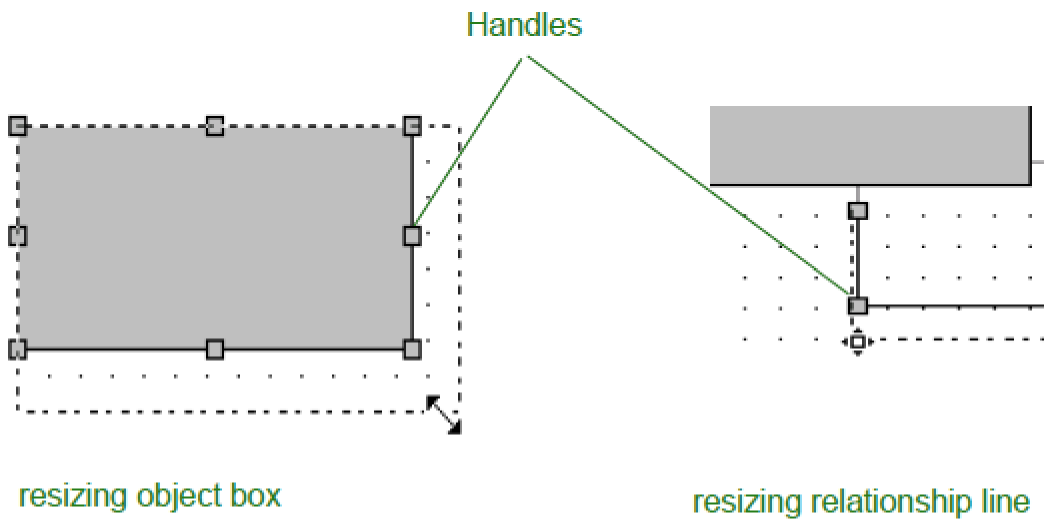
Moving an Object in a Diagram

To move an object in a diagram, select the object and drag it to its new location. You can use the mouse, or you can press (or hold) the arrow keys. As soon as you release the mouse button, the object snaps to the nearest grid point if snap-to-grid is set or to the location where it was released.

Resizing an Object in a Diagram

To change the size of an object in a diagram, select the object and drag one of the handles to a new location. When you select an object in a diagram, the handles are displayed on all sides and corners of a rectangle (for an object) or a line (for a relationship). When the cursor is over a handle, the pointer changes, as shown in Moving handles. This indicates the directions you can drag and move the handle to resize the object or line. To resize both height and width of an object, drag the corner.

Figure 8-38 Moving handles



Finding an Object in a Diagram

If your diagram is large, you can use this option to find an object in the diagram. From the Construction Workbench menu bar, click **Edit > Find** or press (**Ctrl+F**). The Find window displays. You can either search for the entire name or just a string of any named object in the diagram. The substring match finds all occurrences of the text string including the text as part of another word.

Deleting an Object from the Diagram

To delete an object from the diagram, select the object and press the **Delete** key. Optionally, select the object, and from the Construction Workbench menu bar, click **Edit > Clear**. This action deletes the object from the diagram but keeps the object in the repository. To remove the object from the repository, select **Ctrl+Delete** or **Edit > Delete from repository**.

Copying and Pasting an Object in a Diagram

To copy, cut, or paste an object, do one of the following:

- Right-click the object and select **Edit > Copy** or **Cut** or **Paste**
- Select the object, and from the Construction Workbench menu bar, select **Edit > Copy** or **Cut** or **Paste**.

Cut removes the selected object or objects from the diagram but not from the repository. The objects move to the clipboard so you can paste them elsewhere in the diagram or another diagram. **Copy** copies the selected object or objects to the clipboard so you can paste them elsewhere in the diagram or another diagram. **Paste** places any cut or copied object or objects from the clipboard into the display area and selects them.

Arranging Objects in a Diagram

You can perform the following with objects in a diagram using the right-click menu:

- Aligning Objects
- Reformatting Objects

Aligning Objects

You can align the selected objects along their top, bottom, left, or right edges. To align two or more selected objects, do one of the following:

- From the Construction Workbench menu bar, click **Edit > Align**.
- Right-click the second object and click **Edit > Align**.

Reformatting Objects

You can redraw any selected objects to make the diagram more readable. Do one of the following:

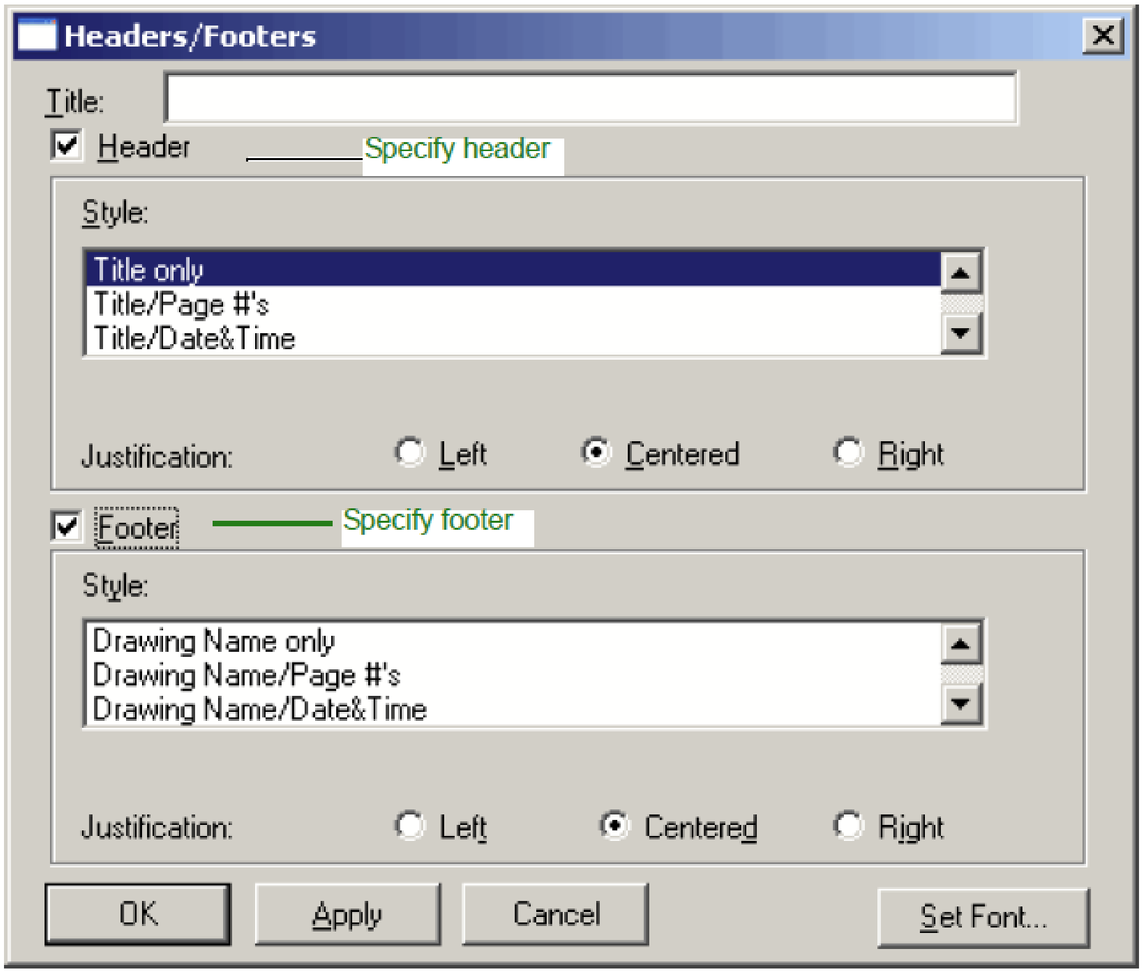
- From the Construction Workbench menu bar click **Edit > Reformat**.
- Right-click the second object and click **Reformat**.

For boxes, the diagram is redrawn with each box long enough to display the name. For lines, the diagram is redrawn with the lines in the best orthogonal path between its endpoints, avoiding boxes and other lines.

Specifying Header or Footer

You can select which information is displayed when you print the diagram. You can select to print a header at the top of every page or a footer at the bottom of every page or both. To specify headers, footers, or both, from the **View** menu, select **Headers/Footers**.

Figure 8-39 Headers and footers example window


Headers/Footers

Title:

[✓] Header — Specify header

Style:
- [✓ Title only]
- Title/Page #'s
- Title/Date&Time

Justification: Left [✓ Centered] Right

[✓] Footer — Specify footer

Style:
- Drawing Name only
- Drawing Name/Page #'s
- Drawing Name/Date&Time

Justification: Left [✓ Centered] Right

If you select the **Header** check box, a header is printed at the top of every page with the information you select from the scroll list (for example, the title and page numbers). You select whether to set it to appear on the right, center, or left of the page. The title is the information you type in the **Title** field. The title is printed on its own line at the top, and the other information is printed on the next line below it.
If you select the **Footer** check box, a footer is printed at the bottom of every page with the information you select from the scroll list (for example, date and time). You select whether to set it to appear on the right, center, or left of the page.
To select the size and color of the font, complete the following steps:

1. Click **Set Font** and change the settings in the **Font window**.
2. Click **OK**.
   The settings apply to both headers and footers.
3. When done with the headers and footers, click **Apply** or **OK**.

Navigating to Another Diagram

A large system has multiple diagrams. Using Navigate, you can move to another diagram that is stored in the repository or create a new diagram. First, select an entity in your diagram. Subsequent navigations from that entity return to the same diagram without further prompting. You can navigate to more than one diagram from a single entity. If you have navigated to more than one diagram from a single entity, each time you attempt to navigate from that entity, the system prompts you with a list of diagrams where you have set navigation paths. The navigate function is not available for all diagrams.

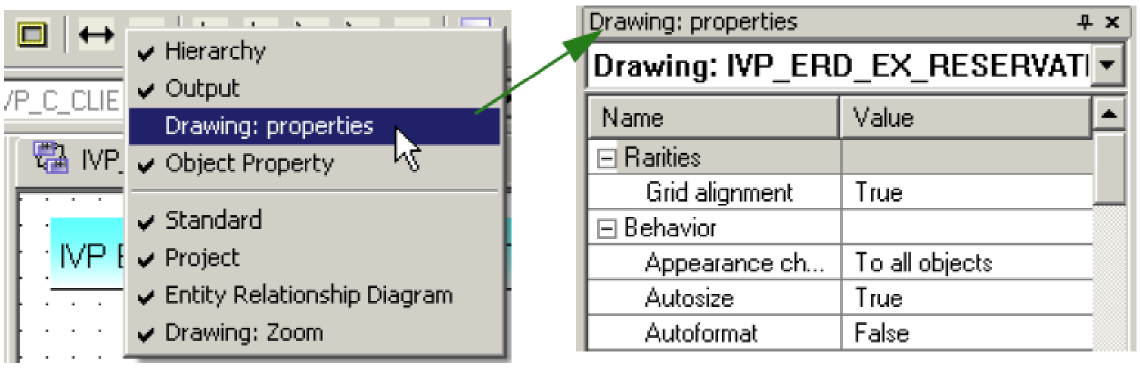
To navigate do the following:

1. With an entity of your diagram selected, click **Edit > Navigate** or right-click on the entity in the diagram and select **Navigate**.
   The Navigate window is displayed.
2. Select the type of diagram that you want to navigate to and click **OK**.
3. In the next window, type the name of the diagram that you want to navigate to or query the repository to find it. If you click **Query** to query the repository, the Insert Drawing window is displayed.
4. From the Insert Drawing window, click **Query**.
5. Select the drawing that you want to navigate to and click **Insert**.

Modifying Aspects of a Diagram

Right-click the **Construction Workbench** toolbar or menu and select **Drawing: Properties** as shown in Accessing diagram properties. These settings are not the same as the diagram properties that are available from the Object Property window of the diagram. The Drawing Properties are more operational settings that apply to the diagram or the objects in the diagram. These settings control object appearance and help you create a properly sized and formatted diagram. The settings are described in Drawing Properties. The Drawing Properties are saved when a diagram is saved.

Figure 8-40 Accessing diagram properties



Drawing: IVP_ERD_EX_RESERVATI

Name	Value
Rarities	
Grid alignment	True
Behavior	
Appearance ch...	To all objects
Autosize	True
Autoformat	False

Table 8-3 Drawing Properties

Name	Description
Rarities	Miscellaneous properties.
Grid Alignment	True or false. Toggles on/off the grid alignment. Default is true.
Behavior	Set various aspects of diagram behavior.
Appearance change applies	When you change one of the settings (properties) under Appearance, AppBuilder applies that change to objects in the diagram according to this setting. Possible settings are as follows: • To all objects - AppBuilder applies that change to all the objects in the diagram. • To new objects - AppBuilder applies that change only to any new objects added to the diagram but not to objects already in the diagram (default setting). Default value is To all objects. This setting only applies to the specific properties listed under Appearance. This setting does not affect other properties, such as Font.
Autosize	True or False. When set to True, the box objects automatically size to the length of their names. Box objects with short names become the default size. Resizes objects in the diagram to display all details. Default is True.
Autoformat	True or False. Set this to True to ensure that a new or moved box object is just long enough to display the object's name or that a new or moved line object is drawn in the best orthogonal path between its endpoints, avoiding box objects and other line objects. Draws any line in as few horizontal and vertical line segments as possible, regardless of how many you draw with the mouse. Default is False.
Multicreate (Repeated Insert)	True or False. If True, you can perform repeated inserts. When you insert an object using the toolbar or Insert menu, the cursor remains the same so that you can insert multiple objects of this type. You must click the arrow cursor or press Esc to set the cursor back to the normal pointer. Default is False.
Font	Set the font for the display.
Font	Select the font, style, and size for the display. The fonts available depend on the fonts you have installed on your system.
Confirmation	Set which actions require confirmation by the user.
Confirm Delete	True or False. True has AppBuilder display a confirmation window before deleting an object in the diagram. Default is True.
Confirm Use	True or False. If True, AppBuilder displays a confirmation window before inserting an object that already exists in the diagram. Default is False.
Confirm Create	True or False. True has AppBuilder display a confirmation window before creating a new object in the diagram. Default is False.
Name Style	Set the style of the names in the diagram.

Name Style	Specify how entity names are displayed in the diagram area. Select either Upper Case, Lower Case, or Mixed Case. Upper Case displays name in all capital letters. Lower Case displays name in all lower-case letters. Mixed Case displays name in lower-case letters except for the first letter in every name and any letter immediately after an underscore. Default is Upper Case. **Note**: Regardless of the name style you choose here for display, the entity's name is stored in the repository as all upper case and must have underscores instead of spaces.
Show Underscore	True or False. True shows the underscore in entity names in the diagram area. False replaces underscores with spaces. Default is False.
Appearance	Set the appearance of various parts of the display.
Show Note borders	True or False. True displays a thin rectangle around Note objects. Default is False.
Hierarchy Display	True or False. True displays the associated properties within each box object. In Entity Relationship Diagram, Database Diagram, Window Flow Diagram, set this to True to display the property, column, or window hierarchy for all entities in the diagram. The hierarchy is displayed inside the objects box in the diagram tool; it does not navigate to the Hierarchy diagram. Default is True.
Label Rotation	True or False. True rotates a diagram's labels so they are parallel to the line or to a segment of a line object that has multiple segments. For example, it aligns roles in the ERD and dependency names in the PDD so they are parallel. Controls how labels align with the lines they designate. When True, labels align on the same slope as the lines and move with them if you move the lines. Default is False.
From Label Display	True or False. True sets the ERD to display the From cardinality label of relationship lines. You can select both the From and To label to make both sets of labels visible. This option shows the label associated with the "from" end of a relationship line. Default is True. Available for ERD drawings only.
To Label Display	True or False. True sets the ERD to display the To cardinality label of relationship lines. You can select both From and To label to make both sets of labels visible. This option shows the label associated with the "to" end of a relationship line. Default is False. Available for ERD drawings only.
Orthogonal Lines	True or False. True ensures that all new line objects are drawn horizontally or vertically. A line object remains black until you name both box objects it connects, then it changes color. Allows you to manually draw line segments rather than the usual, automatic single line. Default is True.
Printing	Set various options for printing.
Print Zoom Percentage	Specify the percentage of print zoom for the display. Must be an integer value. Default is 100.
Pages per Row	Specify the number of pages per row. Default is one (1).
Pages per Column	Specify the number of pages per column. Default is one (1).
Print Zoom Method	Specify the print zoom method. Available values are Direct and Fit to Pages. Default is Direct.
Page Orientation	Specify the page orientation, either Landscape (long dimension is horizontal) or Portrait (short dimension is horizontal). Default is Portrait.

⚠ The drawing properties option from the View menu shows individual properties for each drawing tool that you use, and it differs from the Drawing Tools property page you find in Workbench > Options > Drawing Tools. The drawing properties window overrides any global settings previously made in the Drawing Tools page.

Showing More Information

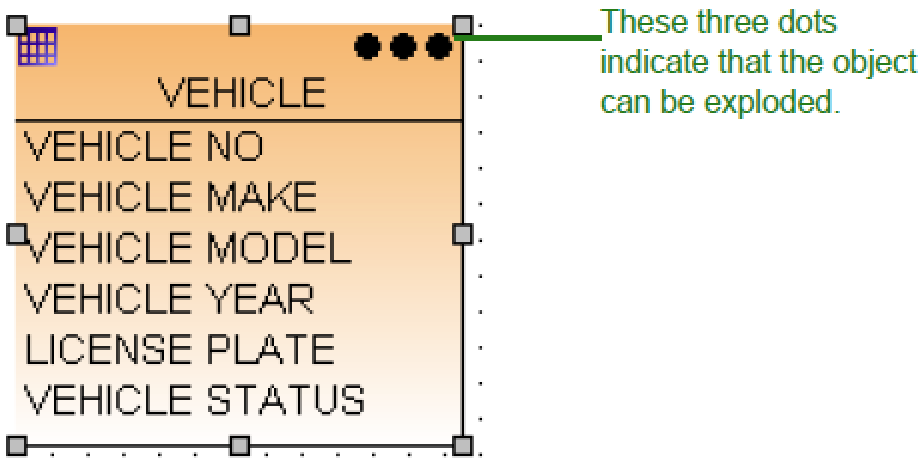
This section discusses how you can see more detail in your drawings and diagrams:

- Exploding Diagram Objects and Relations
- Showing and Hiding Details.

Exploding Diagram Objects and Relations

A diagram object can have more than one object associated with it. When an object has at least one more object attached to it (but not shown), the object shows three dots (...). This indicates that you can expand or explode it to see the other objects.
To explode an object, from the Construction Workbench menu, click **Edit > Explode**. To explode the relationships between objects, select the relationship object and click **Edit > Explode Relations**.

Figure 8-41 Collapsed entity (can be exploded)



With the object selected in the diagram, select **Edit > Explode** or press **F8** to explode the object in the diagram. This expands the next level of objects under it in the diagram.
To explode the relations, with the object selected in the diagram, select **Edit > Explode Relations** or press **Ctrl-F8**.
Do not confuse these with the Expand and Collapse commands in the Hierarchy. The Explode command explodes entities in the diagram but does not change the appearance of the hierarchy; the Expand command in the hierarchy does not change the appearance of the diagram.

⚠ You cannot explode objects and relations in a Window Flow Diagram.

Showing and Hiding Details

Show or hide the details of one or more selected objects including the labels for lines and the hierarchy for boxes. From the **Edit** menu select **Show Details** or **Hide Details**.

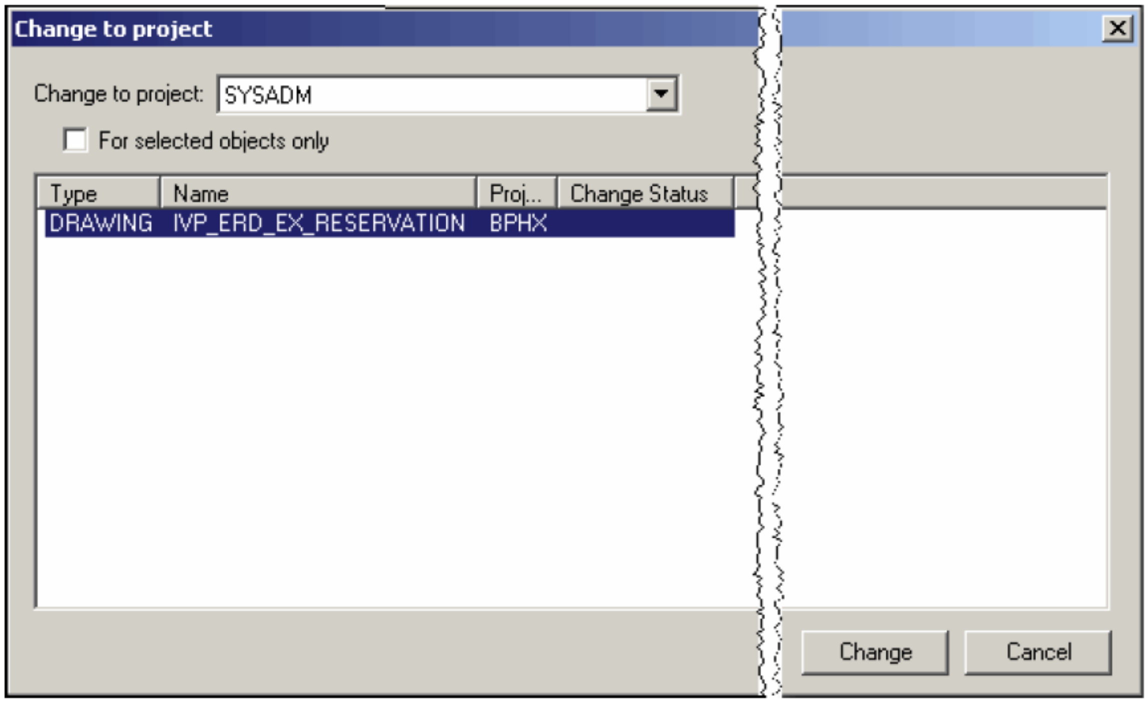
Changing Affiliation and Owner

For a diagram, you can change affiliation with a project (Changing Project Affiliation) or change the owner of the diagram (Changing Diagram Owner). These options are available only for a saved diagram.

Changing Project Affiliation

In a workgroup repository, you can change the project to which a diagram belongs. Refer to the **Repository Administration Guide for Workgroup and Personal Repositories** for security and authorization restrictions. To associate a diagram with a different project, complete the following steps:

1. In the diagram window, right-click and select **Change Project** or select **Edit > Change Project**.
   The Change to Project window displays.
   In the Change to project window, the current diagram object whose project you are changing is displayed in the list.
2. Select the projects to which you want to transfer the object in the **Change to project** drop-down list.
   Figure 8-42 Change project window example



Type	Name	Proj...	Change Status
DRAWING	IVP_ERD_EX_RESERVATION	BPHX	

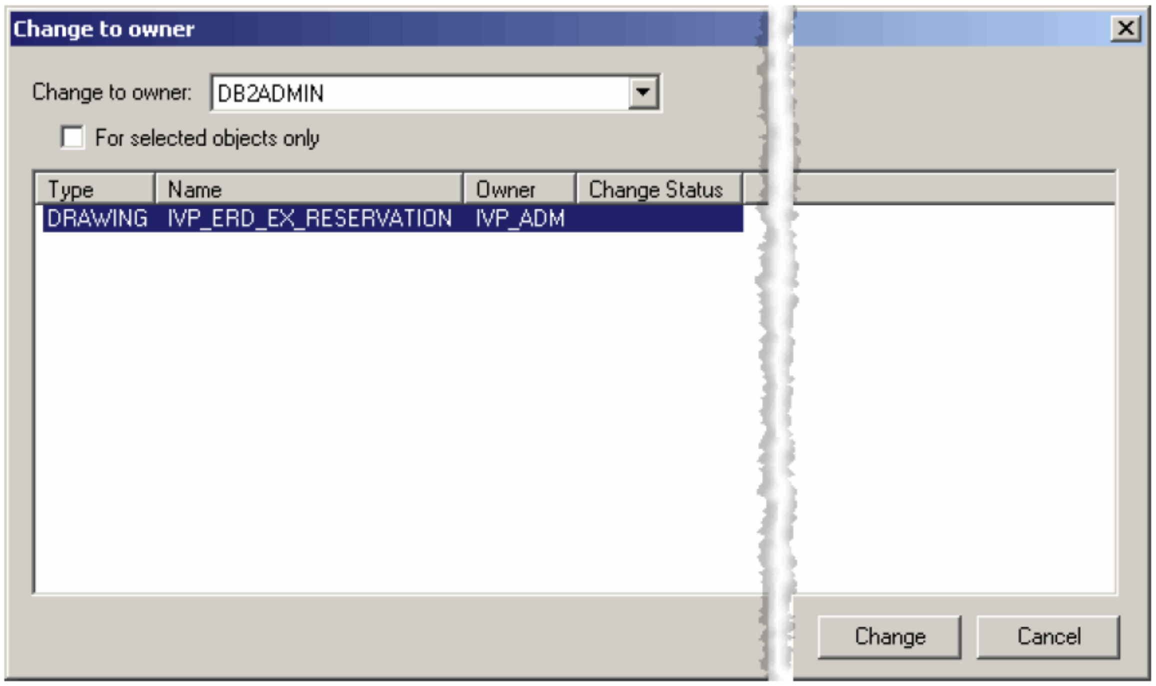
3. Check **For selected objects only** checkbox to change only the affiliation of the selected objects. If this check box is unchecked, all the objects listed in this dialog changes.
4. Click **Change** to accept the changes.

The message "Action successful" displays under the Change Status column.
If you do not have the required authorization, an error message displays that indicates you do not have update authority for the project to which you are changing the object.

Changing Diagram Owner

In a workgroup repository, you can change the owner of a diagram. Refer to the **Repository Administration Guide for Workgroup and Personal Repositories** for security and authorizations restrictions.
To change the owner for a diagram, complete the following steps:

1. In the diagram window, right-click and select **Change Owner** or select **Edit > Change Owner**.
   The **Change to Owner** window displays.
   In the Change to owner window, the current diagram object whose project you are changing displays in the list.
2. Select the project to which you want to transfer the object in the **Change to owner** drop-down list.
   Figure 8-43 Change owner window example


Change to owner

Change to owner: DB2ADMIN

For selected objects only

Type	Name	Owner	Change Status
DRAWING	IVP_ERD_EX_RESERVATION	IVP_ADM	

3. Check **For selected objects only** checkbox to change only the affiliation of the selected objects. If this check box is unchecked, all the objects listed in this dialog change.
4. Click **Change** to accept the changes. The message "Action successful" displays under the Change Status column.

If you do not have the required authorization, then an error message is displayed that indicates you do not have update authority for the diagram.

Construction Tools

Construction Tools

Use the construction tools once you have finished creating a design and you are ready to develop the application. Construction tools include the following:

- Rule Painter
- Component Painter
- Macros
- Bitmap Viewer
- Set Builder
- Matrix Builder

Component Painter

AppBuilder recognizes two kinds of components. System components come with AppBuilder and can be added to a project by locating them and dropping them into the hierarchy. User components are small programs that you write yourself to add to the project.
Use the Component Painter to construct user components. The Component Painter is similar to the Rule Painter, and in fact has a subset of the functionality of the Rule Painter, with a specialized text editor that is displayed in the work area and specialized toolbars and menus. You can perform the following tasks with Component Painter:

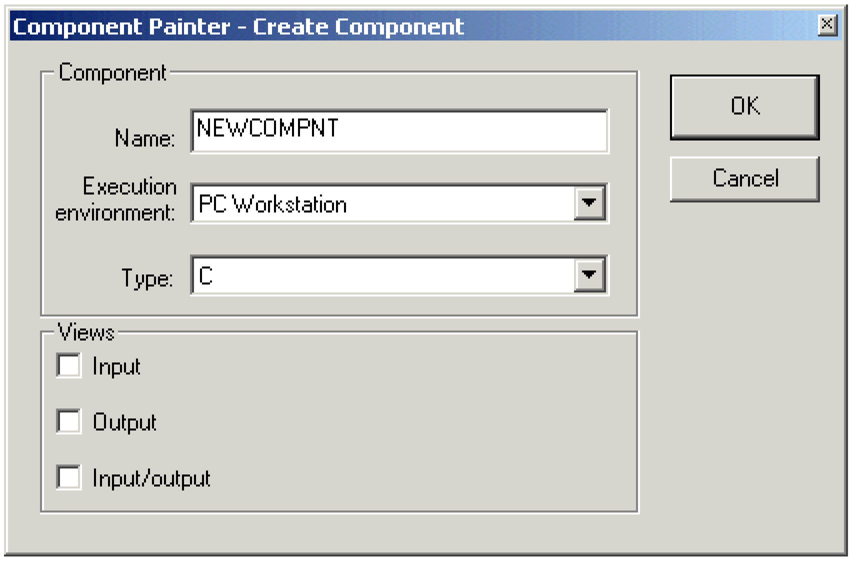
- Creating a User Component
- Using a Macro in a Component
- Using Text Editor Features

Creating a User Component

To create a user component, complete the following steps:

1. From the* **File** menu, select **New**. In the Create New dialog, select **Component**. The Component Painter - Create Component dialog displays, as shown in* **Create Component dialog.**

Create Component dialog



Name: NEWCOMPNT
Execution environment: PC Workstation
Type: C
Views: ☐ Input ☐ Output ☐ Input/output

Component Painter interface	
Field	**Descriptions**
Name	Specify the unique name of the component. Follow the guidelines you have for naming components.
Execution environment	Select from the drop-down list.
Type	Select the language of the component.
Views	Select the type of view.
Input	May be Input or Input and Output.
Output	May be Output or Input and Output.
Input/Output	Mutually exclusive to the other choices.

The Component Painter tab opens in the Work area and you can begin to type the code for the component. The Tools menu changes to allow you to work with macros.
See the *Developing Applications Guide* for more details about user components.

Using a Macro in a Component

From the Component Painter tool, you can use a macro to speed development. Refer to Macros for a description of macros and quick macros. You can use **c_component_skeleton** to create a skeleton for a C component or **JavaUserComponent** to create a skeleton for a Java component.

Using Text Editor Features

From the Component Painter tool, you can use these additional text editor features:

- Set Text Editor options in *Tools* > *Workbench Options* > *Tools* . See Text Editor Options.
- Use *Text Editor -Tools* toolbar to access the macros and other tools. See Text Editor – Tools Toolbar.
- Use *Text Editor - Bookmarks* toolbar, to find text and navigate with bookmarks. See Text Editor – Bookmarks Toolbar.
- Use *Text Editor - Event Wizard* toolbar, to apply events and event procedures. See Text Editor - Event Wizard Toolbar.
- Navigate with bookmarks using *View* > *Bookmarks* . See Text Editor – Bookmarks Toolbar.

Macros0

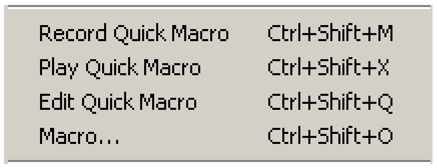
Macros

AppBuilder gives you the ability to create and use macros (text files of Microsoft Visual Basic Scripting commands or VBScript) to speed development by automating routine tasks when creating rules in Rule Painter or creating components in Searching for Regular Expressions. You can create a macro by recording it or by editing an existing macro. When you record a macro, the system remembers your actions, converts them into commands, and inserts the commands into a text file known as a macro file. There are two types of development macros:

- Quick Macros
- Named Macros

Use *Quick macros* to quickly record your actions from a particular session, without having to name a macro, provide a description for it, or review the resulting macro file. This also saves you some keystrokes whenever you want to run the quick macro, as it remains available from the menu until you record another quick macro. Use *named macros* to name, save, and edit the actions you record.
You must use the keyboard when recording macro actions. The system does not record mouse actions when selecting text or moving an insertion point. However, you can use the mouse to click commands and select options. If you are recording a named macro, examine the macro carefully when you are done recording to make sure that it is complete. If you find something missing, add the appropriate code before using it.
From the *Tools* menu, as shown in Macro menu, you have options for using the quick macro or the Macros dialog for named macros.

Macro menu


Record Quick Macro Ctrl+Shift+M
Play Quick Macro Ctrl+Shift+X
Edit Quick Macro Ctrl+Shift+Q
Macro... Ctrl+Shift+O

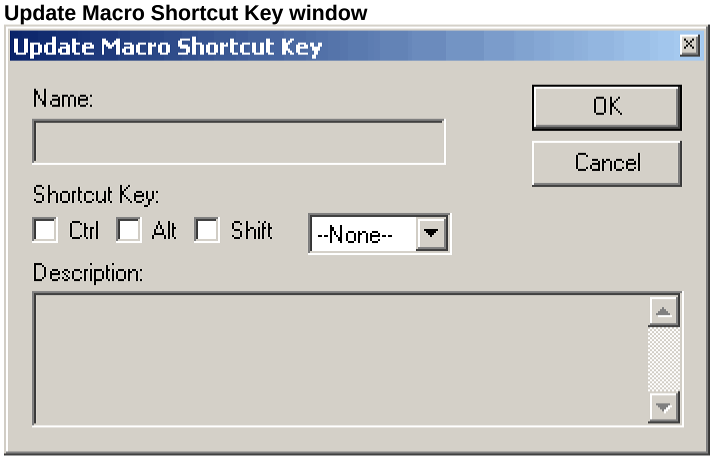
This type of macro is completely different from the type of macro available in the Rules Language. These are development environment macros in VBScript while Rules macros are Rules Language constructs.

Quick Macros

For those routine steps that you would like to record and use without saving to the repository, use the quick macro. AppBuilder provides the ability to record and run quick macros for use in developing rules or components.

Recording a Quick Macro

To record or create a quick macro, complete the following steps:

1. From a Rule Painter window, click the *Record Macro* button in the Text Editor - Tools toolbar.
   The Update Macro Shortcut Key window displays.
   Update Macro Shortcut Key window
   

2. If you want to assign shortcut keys to the macro, select **Ctrl** or **Alt** or **Shift** , or two of them or all of them. Then select a key from the drop-down list.
3. Perform the actions to include in the macro. The system records each action.
4. Complete the necessary actions and click the **Record Macro** button again to stop recording. The system displays the macro information in a quick_macro.qm tab in the Work Area.

You can also record macros by selecting *Tools* > *Record Quick Macro* from the Construction Workbench toolbar or by pressing **Ctrl+Shift+M**.

Playing a Quick Macro

To run the macro, click the **Play Macro** button in the Text Editor - Tools toolbar.

You can also play quick macros by selecting **Tools > Play Quick Macro** from the Construction Workbench toolbar or by pressing **Ctrl+Shift+X**.

Editing a Quick Macro

To edit the macro, select **Tools > Edit Quick Macro** from the Construction Workbench toolbar. The quick_macro.qm tab displays. Edit the macro as necessary.

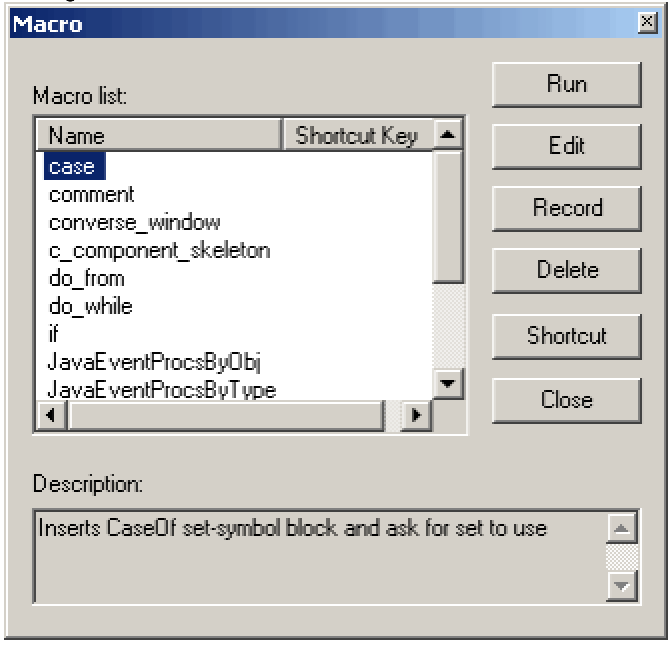
Named Macros

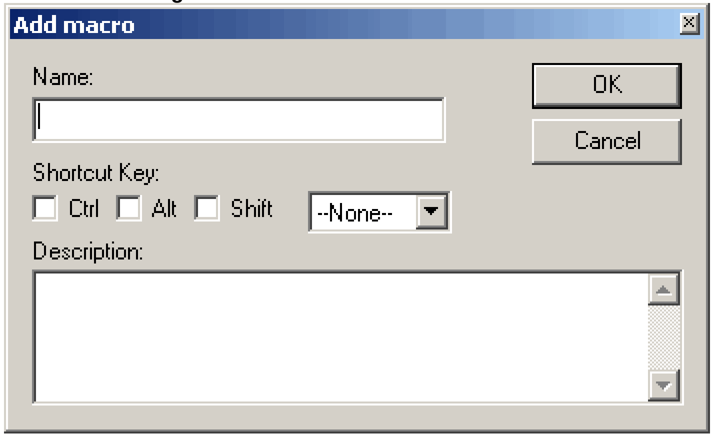
If you use some macros many times during development or across projects, it makes sense to create these macros and name them. AppBuilder provides some macros with the product, but you are not limited to these. Use the Macro dialog to record, edit, and name macros for developing rules or components.

- Recording a Named Macro
- Playing a Named Macro
- Editing a Named Macro

Recording a Named Macro

You can record (that is, create) macros that are named for use, regardless of which quick macro is available. To record a named macro, complete the following steps:

1. From a Rule Painter window, select **Tools > Macro** from the Construction Workbench menu bar.
   The Macro dialog displays, as shown in Adding a macro, displaying the list of saved macros.
   Adding a macro
   

   case
   comment
   converse_window
   c_component_skeleton
   do_from
   do_while
   if
   JavaEventProcsByObj
   JavaEventProcsByType
   Inserts CaseOf set-symbol block and ask for set to use
2. Click **Record**.
   The **Add Macro** dialog is displayed, also shown in Add macro dialog.
   Add macro dialog
   

3. Type a **Name** and **Description** for the macro and click **OK**.
   If you want to assign shortcut keys to the macro, select **Ctrl** or **Alt** or **Shift** , or two of them or all of them. Then select a key from the drop-down list.

The Rule Painter tab displays in the Work Area.
4. Perform the actions to include in the macro.
   The system records each action.
5. Do one of the following to stop recording:

   - Complete the necessary actions and click the **Record Macro** button in the Text Editor - Tools toolbar to stop recording.
   - Select **Tools > Stop Recording** or press **Ctrl+Shift+M** .

The system displays the macro information in a text editor tab in the Work Area.

Playing a Named Macro

To run a macro, select **Tools > Macro** from the Construction Workbench. The Macro dialog displays, showing all the system macros. Select the macro and click **Run** or double-click the name of the macro.

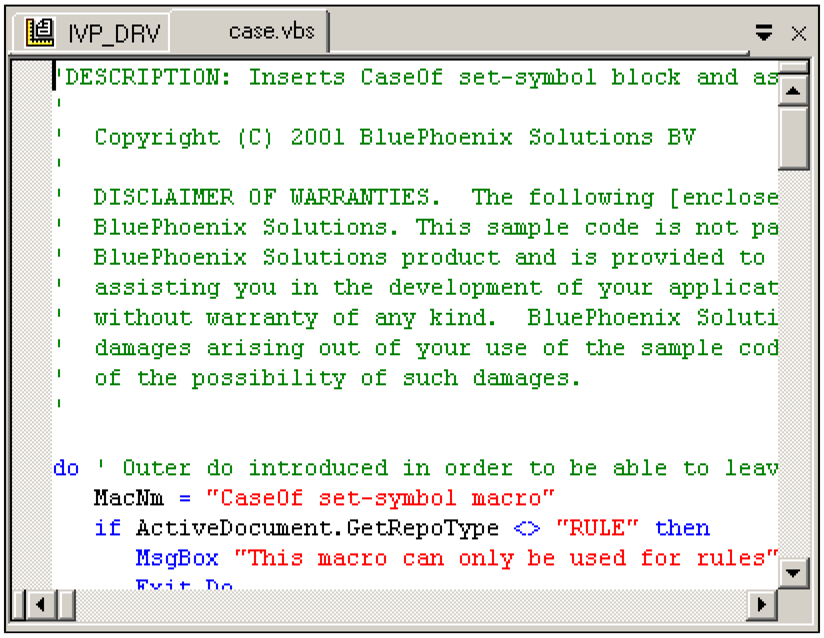
Editing a Named Macro

You can edit macros that are named. These include any of the macros delivered with the product as well as ones you have created. Macros delivered with the product include such helpful automated steps as:

- **c_component_skeleton** - inserts skeleton for a C user component
- **JavaUserComponent** - inserts skeleton for Java user component
- **JavaEventProcByObj** - inserts event procedures for objects for a display rule with window attached
- **JavaEventProcByType** - inserts event procedures for objects for a display rule with window attached

To edit the macro, select **Tools > Macro** from the Construction Workbench and click **Edit**. The Rule Painter tab displays in the Work Area with the text of the macro as shown in Editing a macro. Edit the macro as necessary. When finished, close the window. You can also use the Text Editor - Macros toolbar to select the macro from the Macro drop-down list, then click the **Open Macro** button .

Editing a macro



```
'DESCRIPTION: Inserts CaseOf set-symbol block and as
'
'  Copyright (C) 2001 BluePhoenix Solutions BV
'
'  DISCLAIMER OF WARRANTIES.  The following [enclose
'  BluePhoenix Solutions. This sample code is not pa
'  BluePhoenix Solutions product and is provided to
'  assisting you in the development of your applicat
'  without warranty of any kind.  BluePhoenix Soluti
'  damages arising out of your use of the sample cod
'  of the possibility of such damages.
'

do ' Outer do introduced in order to be able to leav
   MacNm = "CaseOf set-symbol macro"
   if ActiveDocument.GetRepoType <> "RULE" then
      MsgBox "This macro can only be used for rules"
      Exit Do
```

Bitmap Viewer

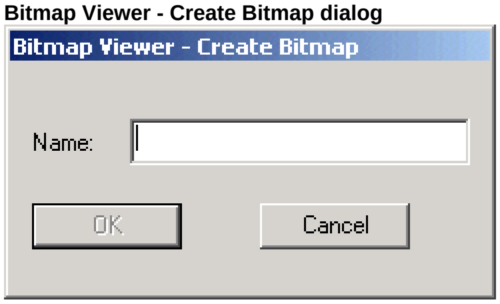
The Bitmap Viewer is a tool for moving a graphic image in and out of a bitmap object and for viewing the image. The image can be a JPEG file, BMP file, or ICO file (icon).Typically the Bitmap Viewer is used to create an association between a Bitmap object in Window Painter and the file providing the image (though such associations can also be created with system components). Because bitmaps are used only to add visual interest, they are not linked to repository fields. See Using the Hierarchy Window.
Tasks for inserting a graphic image include:

- Opening the Bitmap Viewer
- Importing a Bitmap Image
- Adding a Bitmap to the Hierarchy
- Viewing a Bitmap Image

- Exporting a Bitmap Image

Opening the Bitmap Viewer

To open the Bitmap Viewer, complete the following steps:

1. Select*File > New.*
2. In the **Create New** dialog box, select **Bitmap** and click **OK**. The Create Bitmap dialog box is displayed.
   Bitmap Viewer - Create Bitmap dialog
   

3. Type the Name of the Bitmap and click **OK**. The Bitmap Viewer tab displays in the Work Area. This tab is empty until you import a graphic file into it.

Although the AppBuilder terminology is "Bitmap," AppBuilder does support the following graphic file types: JPEG (.jpeg,.jpg,.jpe), BMP (.bmp), or ICON (.ico).

Importing a Bitmap Image

To import an image, complete the following steps:

1. Right-click in the Bitmap Viewer window and select **Import** or select **File > Import**.
2. In the Import Bitmap window, select the drive and directory from which you want to import the bitmap. Select the type of image. Valid image types are JPEG (.jpeg,.jpg,.jpe), BMP (.bmp), or ICON (.ico).
3. Select the graphic file or type in the file name to add the file to the repository and click **Open** . When this dialog is closed, the image is displayed in the Bitmap Viewer.
   You can also drag the bitmap, icon, or JPEG file from Windows Explorer and drop it in the Bitmap Viewer.

After importing the bitmap graphic, the system displays the following information in the status bar. See Status bar with image type and size for an example of a status bar with bitmap graphic information.

Status Bar information

Field	Description
Type	Type of graphic
X Res	Width (X-axis) of the graphic. XRes and YRes are displayed in XRes * YRes format.
Y Res	Height (Y-axis) of the graphic. XRes and YRes are displayed in XRes * YRes format.

The name of the bitmap graphic is displayed on the title bar, and the image of the graphic is displayed in the document area of the Bitmap Viewer.

Adding a Bitmap to the Hierarchy

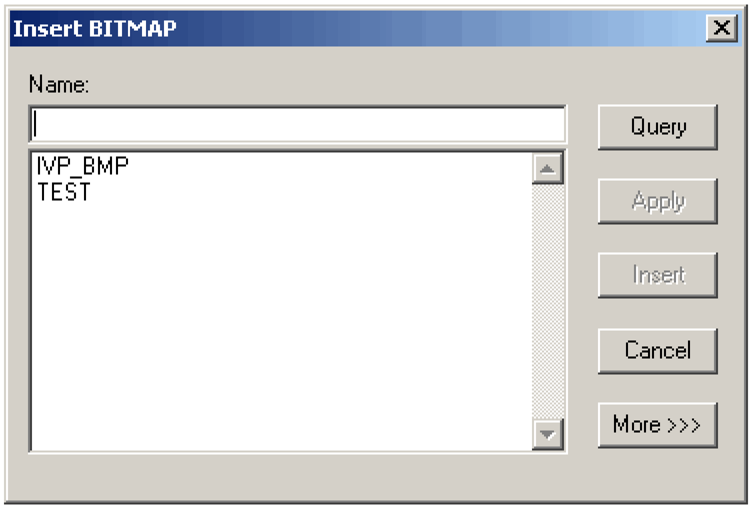
Right-click in the Bitmap Viewer window and select **Navigate to Hierarchy** or select **View > Navigate to Hierarchy**. The bitmap name is displayed in the Hierarchy Window, Repository tab.

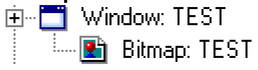
To insert the bitmap to a project's hierarchy, use the Insert menu options. See Adding a Bitmap to a Window for an example.

Adding a Bitmap to a Window

To add a bitmap to a Window, complete the following steps:

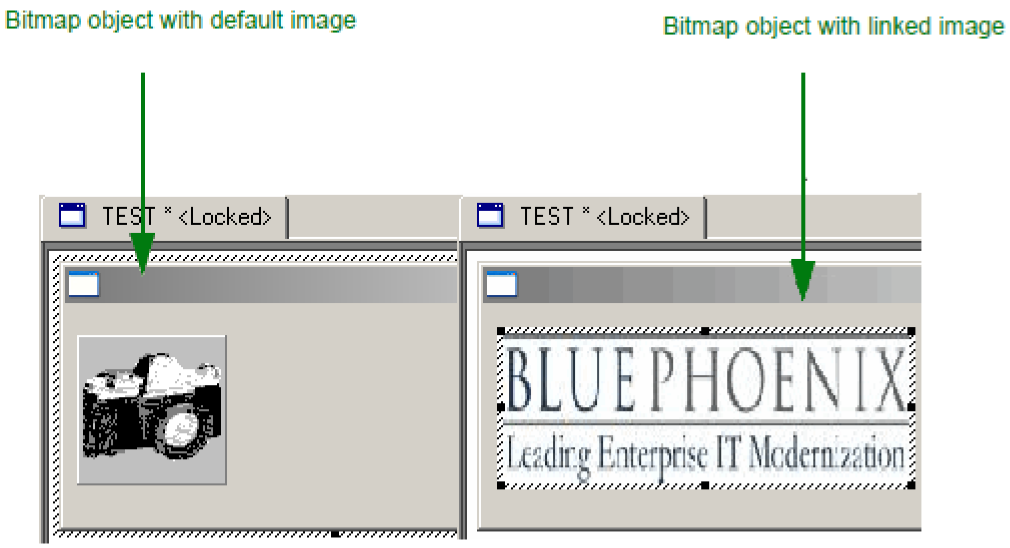
1. Right-click the Window, then select **Insert Child > Bitmap**.
   The Insert Bitmap dialog displays (see Insert Bitmap dialog sample).
   Insert Bitmap dialog sample



2. Click **Query** to display the list of available bitmaps, then select one of them.
   The selected bitmap is displayed as a child object of the window in Project tab (see Bitmap inserted as a child object of a window).
   Bitmap inserted as a child object of a window
   

3. Drag the bitmap name from the Hierarchy to the associated Window opened in the Work Area. The bitmap object is displayed in the Window. If you have already imported the graphic image (see Importing a Bitmap Image), the image is displayed within the Bitmap Object.

Bitmap Viewer with images


Bitmap object with default image
Bitmap object with linked image
TEST * <Locked>
TEST * <Locked>
BLUE PHOENIX
Leading Enterprise IT Modernization

To import a graphic, complete the following steps:

1. Double-click the bitmap name in the Hierarchy. The Bitmap Viewer displays in the Work Area.
2. Repeat steps in Importing a Bitmap Image to import a graphic.

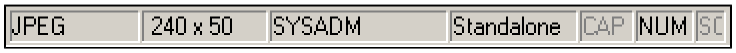
If the image is in the repository, you can display a graphic image in a bitmap object using the Link property in the Properties window for the bitmap. Click the bitmap object; and, from the drop-down list of possible Bitmap objects, select the image.

Viewing a Bitmap Image

To view an existing bitmap and information about it, select *File > Open > Bitmap* and browse for the file. Or in the Hierarchy window, double-click the name of the bitmap. The Bitmap Viewer tab is displayed in the Work Area, as shown below. The status bar at the bottom of the Construction Workbench displays the type, height, and width of the image, as shown in Status bar with image type and size.

Bitmap Viewer window

Status bar with image type and size



Exporting a Bitmap Image

You can export the image associated with a bitmap object to a file. You can edit the BMP, JPEG or ICO file in a graphics editor and re-import it into the repository.
To export an image, complete the following steps:

1. Right-click in the Bitmap Viewer and select **Export**, or select **File > Export{_}**.
2. In the Export Bitmap window, select the drive and directory where you want to store the bitmap. In the **File name:** field, type the name of the file. Click **Save**.
3. To Save the bitmap information to the repository, click **Commit**.

Rule Painter

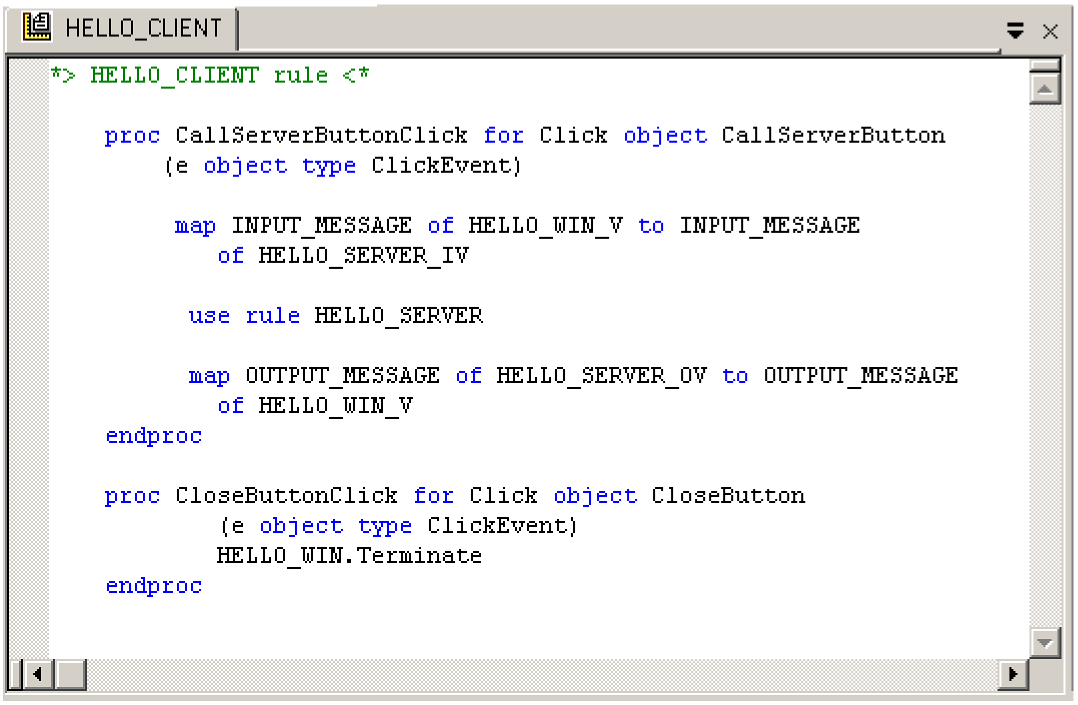
Rule Painter

Use the Rule Painter to create and edit rules in Rules Language source code and to verify the syntax of that code before preparing it. The Rule Painter is similar to a standard text editor with added functionality. You can enter, change, delete, and move text around the work area, define macros, and drag and drop objects from the hierarchy. See Rule Painter window for a simple example developed more fully in the *Getting Started Guide* .
In any Rule Painter window, you can perform the following tasks:

- Creating and Editing a Rule
- Inserting Object Names
- Inserting Reserved Words
- Using a Macro
- Using Text Editor Features
- Using the Mapping/Setting Wizard
- Using the SQL Builder
- Verifying Rules
- Searching for Regular Expressions

A rule is a series of programming statements that define the logic of an application?how the entities that comprise your application interact. At execution time, each statement performs a unique action. Use the Rule Painter in the Construction Workbench to create rules.

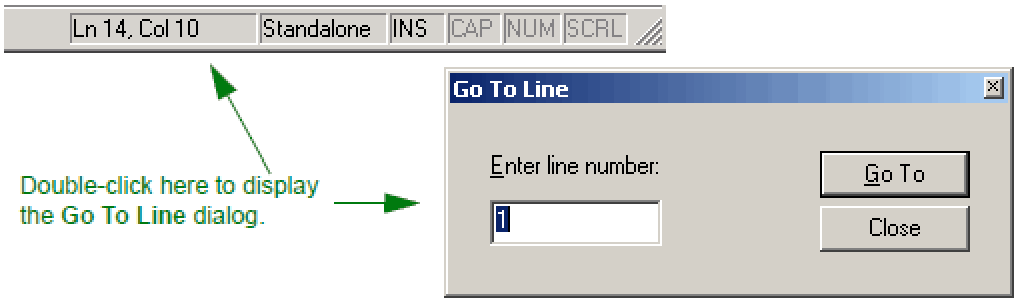
Rule Painter window



```
*> HELLO_CLIENT rule <*

   proc CallServerButtonClick for Click object CallServerButton
       (e object type ClickEvent)

        map INPUT_MESSAGE of HELLO_WIN_V to INPUT_MESSAGE
           of HELLO_SERVER_IV

         use rule HELLO_SERVER

         map OUTPUT_MESSAGE of HELLO_SERVER_OV to OUTPUT_MESSAGE
           of HELLO_WIN_V
   endproc

   proc CloseButtonClick for Click object CloseButton
           (e object type ClickEvent)
           HELLO_WIN.Terminate
   endproc
```

The line number and character position of your cursor in the Rule Painter is shown in the status bar at the bottom of the Construction Workbench window. Double-click a position pane in the status bar to bring up the *Go To Line* dialog.

Line and column numbers in the status bar and the Go To Line dialog



Creating a rule is similar to a creating a programming procedure in a computer language. The Rules Language consists of the statements, keywords, and arguments that comprise each rule. The Rules Language is both powerful and easy to use because it is portable, modular, usable, and reusable:

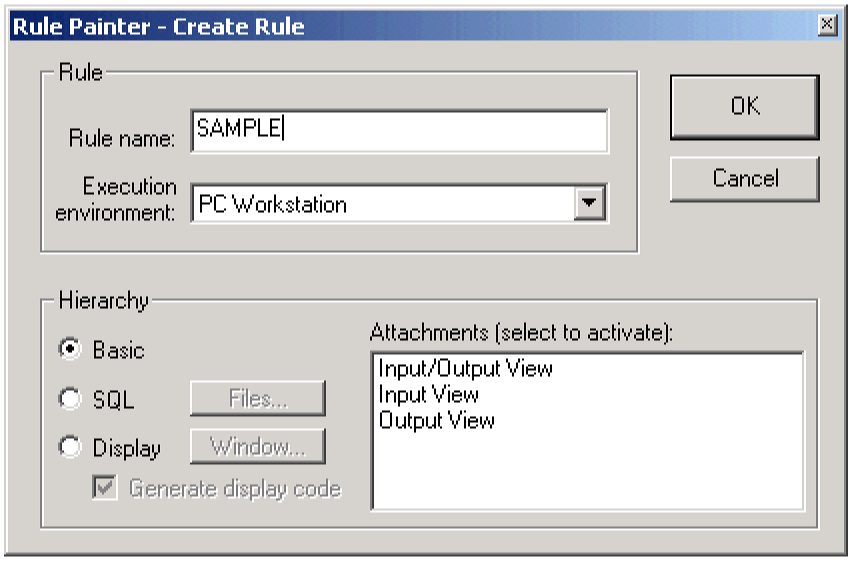
- *Platform-independent generation* ? The same Rules Language code can generate platform-specific executable files for supported hardware environments from Windows machines to UNIX machines and application servers to mainframes.
- *Modular* ? You define the non-local data structures for an application as views and fields in the repository outside its rules. Thus, you can change your data structures without necessarily having to change your code. Each rule performs one distinct task, such as retrieving data or drawing a window, so that you can easily understand any action of a rule.
- *Usable* ? The Rules Language is especially easy to learn and use because it contains a small set of statements, each with just a few clauses, keywords, and arguments.
- *Reusable* ? Because it is modular and all rules are stored in the repository, you can reuse a rule that you or someone else has written.

See the *Developing Applications Guide* for a discussion of the general use of rules. See the *Rules Language Reference Guide* for complete syntax information about individual rules.

Creating and Editing a Rule

To edit an existing rule, open the Rule Painter for that rule and modify the rule as you would edit any code in a source code or text editor.
To create a new rule, complete the following tasks:

1. From the Construction Workbench menu bar, click *File* > *New* .
2. In the Create New dialog, select *Rule* .
   The Create Rule dialog box displays.
   Create Rule dialog


Rule Painter - Create Rule

Rule

Rule name: SAMPLE

Execution environment: PC Workstation

Hierarchy

(●) Basic

( ) SQL

( ) Display

[✓] Generate display code

Attachments (select to activate):

Input/Output View
Input View
Output View

You can place three types of rules in a hierarchy. See the corresponding sections for more information:

- Creating a Basic Rule
- Creating an SQL Rule
- Creating a Display Rule

3. Click *OK* .

A new rule is created in the repository, and a Rule Painter window opens in the work area of the Construction Workbench.

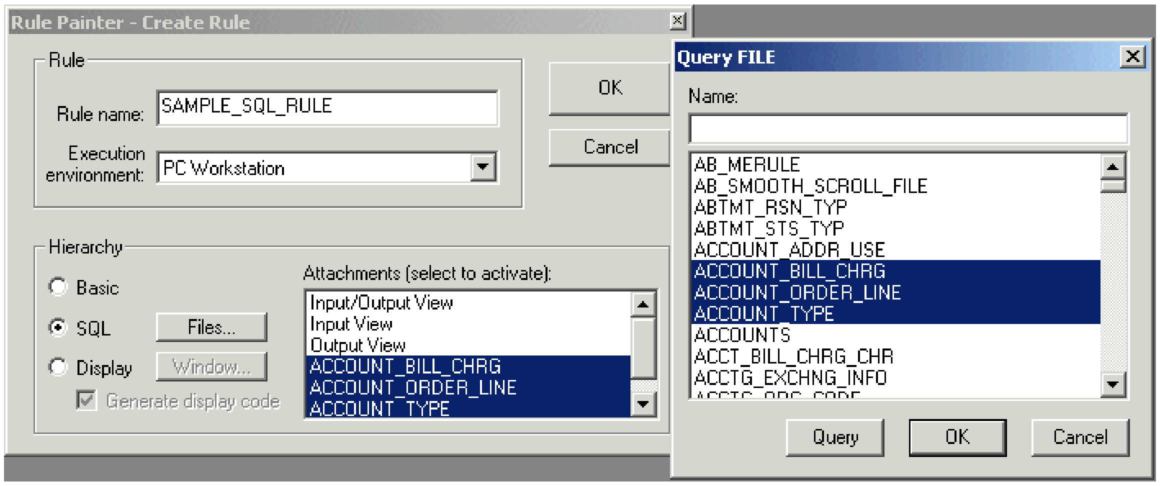
Creating a Basic Rule

When you check the *Basic* button, the Attachments List shows a list of types of views typically attached to a rule, as shown in Create Rule dialog. Select one or more views and click *OK* . A blank Rule Painter window opens.

Creating an SQL Rule

When you check the *SQL* button, the Attachments List shows a list of types of views typically attached to a rule, as shown in SQL Rule Option. Click the *Files* button to add file objects to the Attachments List. Select one or more views or files to attach and click *OK* . A blank Rule Painter window opens.

SQL Rule Option


Rule Painter - Create Rule

Rule

Rule name: SAMPLE_SQL_RULE

Execution environment: PC Workstation

Hierarchy

( ) Basic

(●) SQL

( ) Display

[✓] Generate display code

Attachments (select to activate):

Input/Output View
Input View
Output View
ACCOUNT_BILL_CHRG
ACCOUNT_ORDER_LINE
ACCOUNT_TYPE

Query FILE

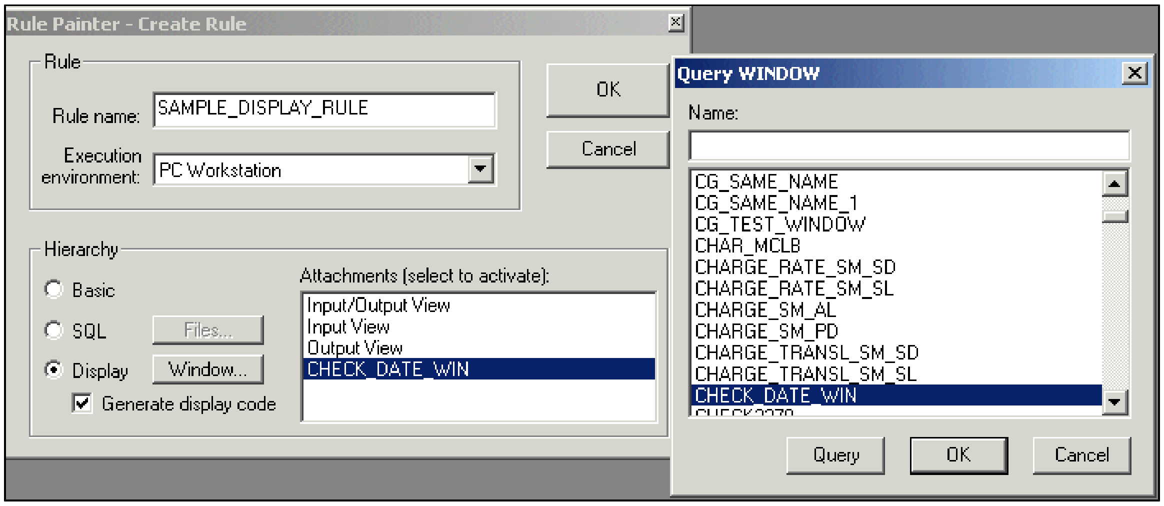
Name:

AB_MERULE
AB_SMOOTH_SCROLL_FILE
ABTMT_RSN_TYP
ABTMT_STS_TYP
ACCOUNT_ADDR_USE
ACCOUNT_BILL_CHRG
ACCOUNT_ORDER_LINE
ACCOUNT_TYPE
ACCOUNTS
ACCT_BILL_CHRG_CHR
ACCTG_EXCHNG_INFO

Creating a Display Rule

When you check the *Display* button, the Attachments List shows a list of types of views typically attached to a rule, as shown in Display Rule Option. Click the *Window* button to add a window object. Select one or more views or a window and click *OK* .

Display Rule Option


Rule Painter - Create Rule

Rule

Rule name: SAMPLE_DISPLAY_RULE

Execution environment: PC Workstation

Hierarchy

( ) Basic
( ) SQL Files...
(•) Display Window...
[✓] Generate display code

Attachments (select to activate):

Input/Output View
Input View
Output View
CHECK_DATE_WIN

OK Cancel

Query WINDOW

Name:

CG_SAME_NAME
CG_SAME_NAME_1
CG_TEST_WINDOW
CHAR_MCLB
CHARGE_RATE_SM_SD
CHARGE_RATE_SM_SL
CHARGE_SM_AL
CHARGE_SM_PD
CHARGE_TRANSL_SM_SD
CHARGE_TRANSL_SM_SL
CHECK_DATE_WIN

Query OK Cancel

If the *Generate Display Code* check box is checked, the Rule Painter window opens with the template of Rules source code for a typical C application display rule. For Java, you can uncheck this option and a blank Rule Painter window opens; to add Java code, use one of the Named Macros to insert skeleton code.

The Rules source code template is for C applications only.

Inserting Object Names

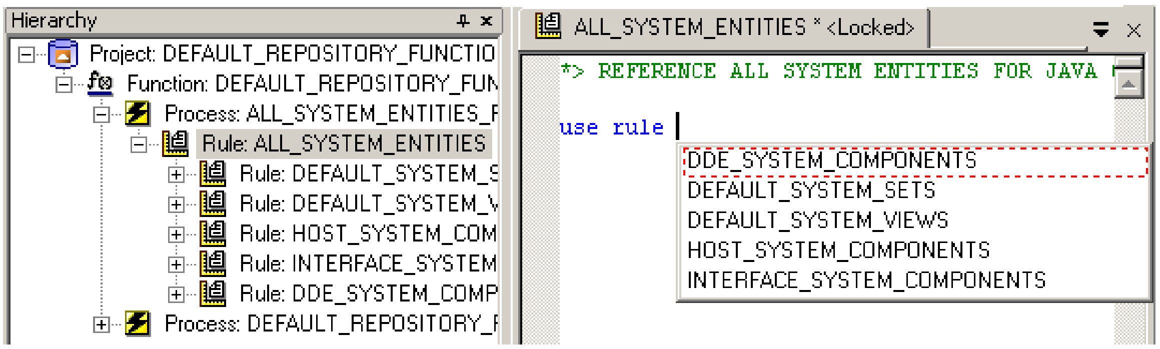
You can insert object names into the Rules source code in the Rule Painter by:

- Using Popup Values
- Auto Completion for Dot Notation
- Using the Insert Commands
- Dragging and Dropping from the Hierarchy

Using Popup Values

For certain key statements, such as USE RULE, CONVERSE WINDOW, and USE COMPONENT, a popup list box of possible objects of a certain type displays.
For example, if you type USE RULE and press the space bar, AppBuilder finds the names of all child rules and lists them in a popup list box. An example is shown in Popup values in Rule Painter. Use the up and down arrow keys to highlight the one you want, if there are more than one. Press *Enter* to select the name of the object to insert.

Popup values in Rule Painter


Hierarchy

Project: DEFAULT_REPOSITORY_FUNCTIO
Function: DEFAULT_REPOSITORY_FUN
Process: ALL_SYSTEM_ENTITIES_F
Rule: ALL_SYSTEM_ENTITIES
Rule: DEFAULT_SYSTEM_S
Rule: DEFAULT_SYSTEM_V
Rule: HOST_SYSTEM_COM
Rule: INTERFACE_SYSTEM
Rule: DDE_SYSTEM_COMP
Process: DEFAULT_REPOSITORY_F

ALL_SYSTEM_ENTITIES * <Locked>

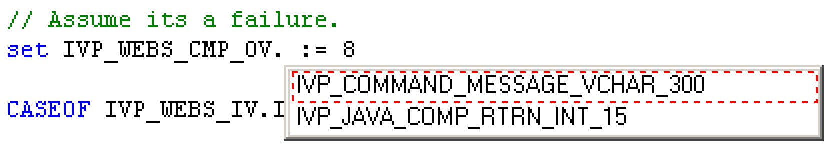
```
*> REFERENCE ALL SYSTEM ENTITIES FOR JAVA

use rule
```

DDE_SYSTEM_COMPONENTS
DEFAULT_SYSTEM_SETS
DEFAULT_SYSTEM_VIEWS
HOST_SYSTEM_COMPONENTS
INTERFACE_SYSTEM_COMPONENTS

Auto Completion for Dot Notation

A popup list of possible values can be displayed after entering a dot "." in the SET statement. To use this feature, select the Auto Completion option in the Text Editor section of the Workbench Options dialog. Refer to the Text Editor Options for more information. Auto Completion example shows an example.

Auto Completion example

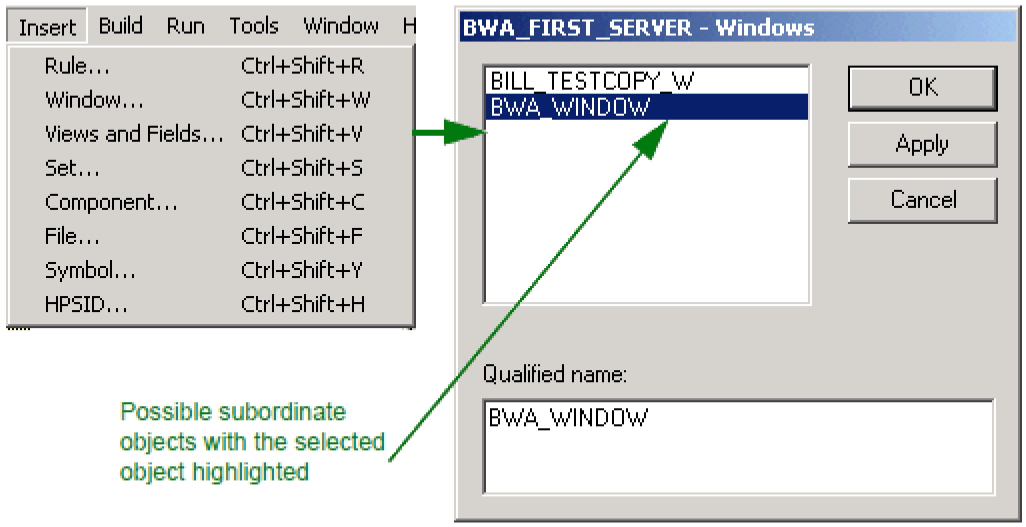


```
// Assume its a failure.
set IVP_WEBS_CMP_OV. := 8

CASEOF IVP_WEBS_IV.I
```

Using the Insert Commands

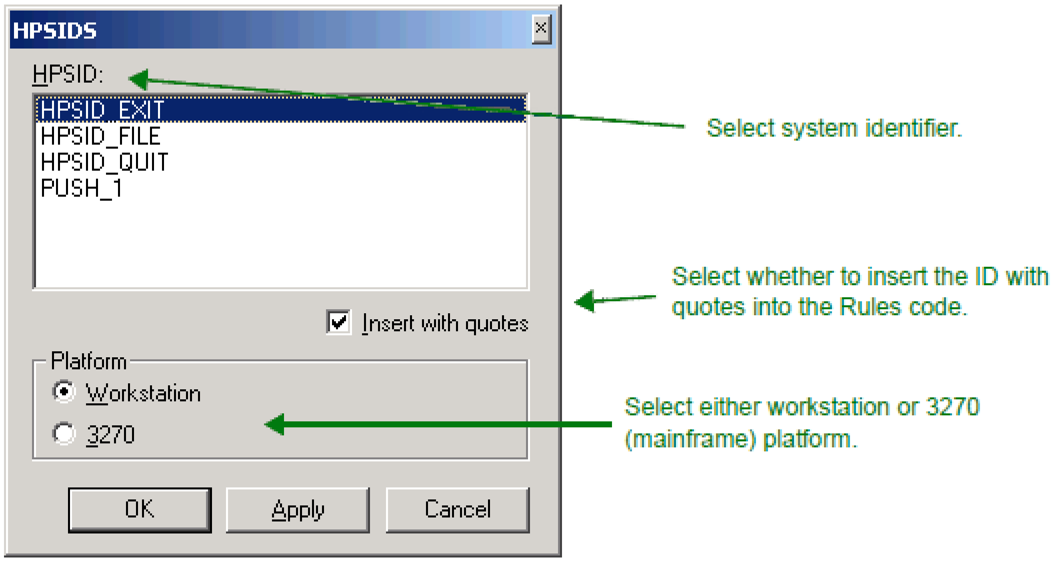
One way to add object names to the Rule source code in a Rule Painter window is to insert them using the Insert commands. Either from the *Insert* menu or the keyboard shortcuts (*Ctrl+Shift+ a letter key*), you can select only objects that are subordinate to the rule that is open in Rule Painter. Insert commands and example dialog shows an example with two possible window names that can be inserted in the rule code. Follow the same procedure for any of the objects in the menu.

Insert commands and example dialog



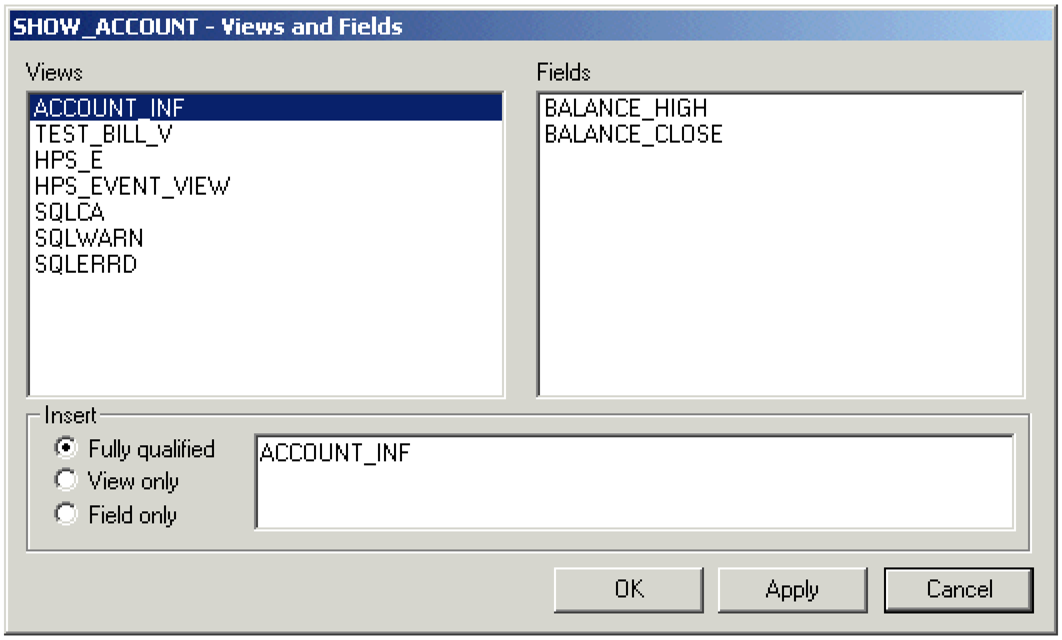
To insert a system identifier (HPS ID), select *HPSID* from the menu or use the keyboard short-cut (*Ctrl+Shift+H*) and the dialog in System Identifier selection dialog displays. Select whether the object is for deployment on a workstation or a mainframe by selecting either the Workstation or 3270 radio button. Then, from the list, select one system identifier. To insert the identifier into the Rules source code with quotes, leave the check box checked. To insert it without the quotes, uncheck the box.

System Identifier selection dialog


HPSIDS
HPSID:
HPSID_EXIT
HPSID_FILE
HPSID_QUIT
PUSH_1
Insert with quotes
Platform
Workstation
3270
OK
Apply
Cancel
Select system identifier.
Select whether to insert the ID with quotes into the Rules code.
Select either workstation or 3270 (mainframe) platform.

To insert a view or a field, select them from the Views and Field dialog as shown in Views and Fields selection dialog. Use the Insert menu or the keyboard short-cut *Ctrl+Shift+V* . You can select either a view or a field of a view. To select a field, select the view that contains it. From the Fields list, select the field. If you would like it fully qualified with the name of the field and the view, select the appropriate radio button. When done, click *Apply* or *OK* .

Views and Fields selection dialog



Views: ACCOUNT_INF, TEST_BILL_V, HPS_E, HPS_EVENT_VIEW, SQLCA, SQLWARN, SQLERRD

Fields: BALANCE_HIGH, BALANCE_CLOSE

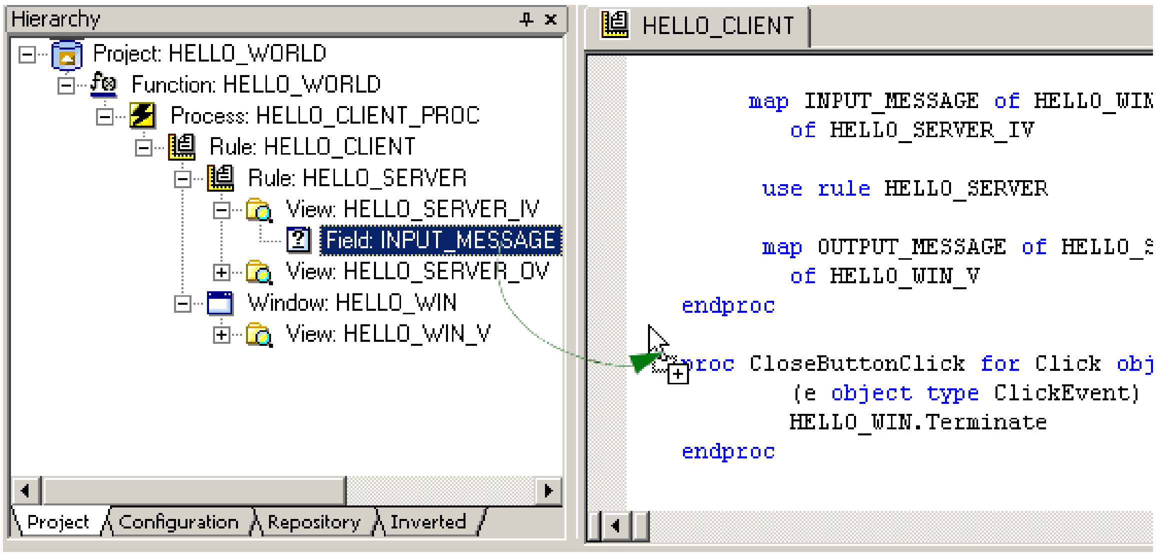
Insert: [✓ Fully qualified] View only Field only

ACCOUNT_INF

Dragging and Dropping from the Hierarchy

You can use the objects in the hierarchy and then drag-and-drop the object from the hierarchy window into the Rule Painter window to add object names to the Rule source code in a Rule Painter window. This is shown in Dragging and dropping a field into a rule. You can drag and drop anything in the hierarchy to the rule in Rule Painter. If it is a field or a view, AppBuilder inserts the fully qualified name, for example, SAMPL_INPUT OF SAMPL_VIEW. For more information about drag-and-drop functionality, refer to Dragging and Dropping.

Dragging and dropping a field into a rule

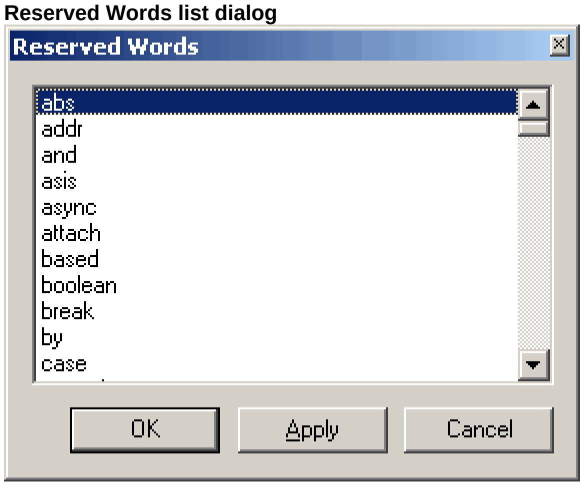


```
        map INPUT_MESSAGE of HELLO_WIN
           of HELLO_SERVER_IV

         use rule HELLO_SERVER

         map OUTPUT_MESSAGE of HELLO_S
           of HELLO_WIN_V
endproc

proc CloseButtonClick for Click obj
        (e object type ClickEvent)
        HELLO_WIN.Terminate
endproc
```

In the Text Editor options (*Tools* > *Workbench Options* > *Text Editor*), you can select *Prefer dot notation* . If this is selected, AppBuilder uses dot notation during drag-and-drop, copy and paste, and in the Insert Views and Fields dialog and the Map Wizard. Refer to Text Editor Options.
By default, any time you drag objects in the Hierarchy tab and drop the objects into another location, the hierarchy tree expands. If you do not want objects to expand automatically when you drag them from the hierarchy tree and drop them somewhere else, uncheck the Auto expand when drag and drop choice under the Double-click action field in Hierarchy options, Workbench Options dialog.
If you hold the *Alt* key while dropping an object in the Rule Painter, another notation is used.

Inserting Reserved Words

Several words are reserved for use in AppBuilder and these have special meaning in the Rules Language. Use them to define data types, make function calls, and in other Rules logic statements. See the *Rules Language Reference Guide* for a list and explanation of reserved words. To see a list of the words and add them one at a time into the Rules source code in Rule Painter, use the Reserved Words list.
To use the Reserved Words list to insert a reserved word, complete the following steps:

1. Open the Rule Painter window for the rule.
2. Click the *Reserved Words* button in the Text Editor - Tools toolbar or, from the *Tools* menu, select *Reserved Words* or press *Ctrl+Shift+E* . The Reserved Words window is displayed (see Reserved Words list dialog).
   Reserved Words list dialog
   

3. You can scroll up and down to find a particular word.
4. If you wish to insert a word into the Rules source code in the Rule Painter, select the word and click *OK* or click *Apply* to add additional reserved words.
5. When done, click *Close* to close the dialog.

Using a Macro

From the Rule Painter tool, you can use a macro to speed development. Refer to Macros for a description of macros and quick macros.

Using Text Editor Features

From the Rule Painter tool, you can use the following additional text editor features:

- Set Text Editor options in *Tools* > *Workbench Options* > *Tools* . See Text Editor Options.
- Use *Text Editor - Tools* toolbar to access the macros and other tools. See Text Editor - Tools Toolbar.
- Use *Text Editor - Bookmarks* toolbar, to find text and navigate with bookmarks. See Text Editor - Bookmarks Toolbar in Dockable Control Bars.
- Use *Text Editor - Event Wizard* toolbar, to apply events and event procedures. See Text Editor - Event Wizard Toolbar.
- Navigate with bookmarks using *View* > *Bookmarks* . See Text Editor - Bookmarks Toolbar in Dockable Control Bars.

Use the icons on the Text Editor - Tools Toolbar or the choices in the Tools menu to add statements to rules using one of the automated tools.

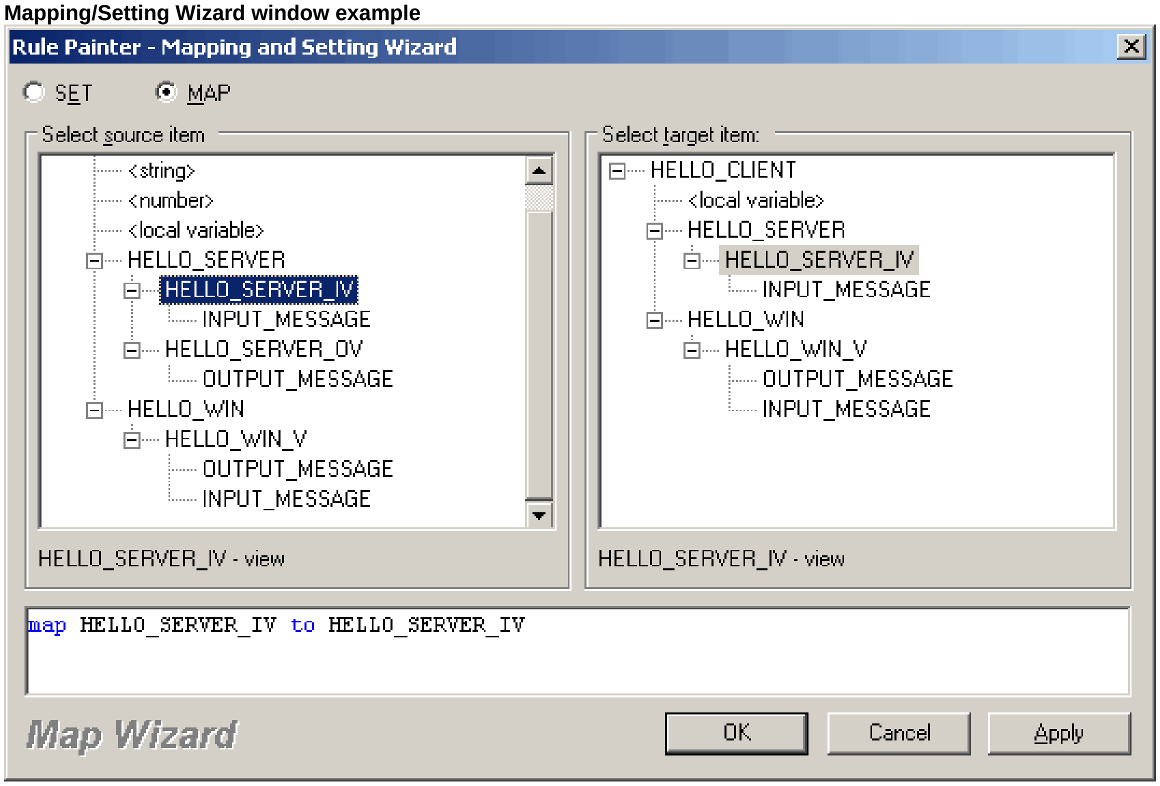
Using the Mapping/Setting Wizard

Use the Mapping/Setting Wizard to quickly build **MAP** (or **SET**) statements. It displays two lists: source data items and target data items. The source data list displays only the data items visible to that rule, which includes its own views, fields, sets and symbols, and output views of accessed rules and components. The target list contains the same views, fields, and child rules; it lists the input views of called rules and components instead of output views.
The Mapping/Setting Wizard validates the data you selected to prevent illegal mappings; however, the Mapping/Setting Wizard does not validate data you type as a literal, constant, or local variable name.
Follow the steps to create a **MAP** (or **SET**) statement:

1. Open the Rule Painter tab for the rule.
2. Right-click in the Rule Painter window and select **Mapping/Setting Wizard** from the pop-up menu. The Mapping/Setting Wizard window is displayed (see Mapping/Setting Wizard window example).

   You can also display the Mapping/Setting Wizard window by clicking on the **Mapping/Setting Wizard** button in the Text Editor -

Tools toolbar or by selecting **Mapping/Setting Wizard** from the **Tools** menu.
Mapping/Setting Wizard window example



```
map HELLO_SERVER_IV to HELLO_SERVER_IV
```

3. Check **SET** or **MAP** at the top of the dialog to create **SET** statements or **MAP** statements, respectively.
4. When creating a **MAP** statement, select a source data item from the left column and a target view from the right column. When creating a **SET** statement, select a target data item from the left column and a source view from the right column. The Mapping/Setting Wizard automatically constructs the **MAP** (or **SET**) statement.
   The system displays a warning message if you attempt to create an invalid **MAP** (or **SET**) statement.
5. Click **OK** to add the **MAP** (or **SET**) statement to the rule or click **Apply** to create another **MAP** (or **SET**) statement.
6. When done, click **Close** to close the window.

Using the SQL Builder

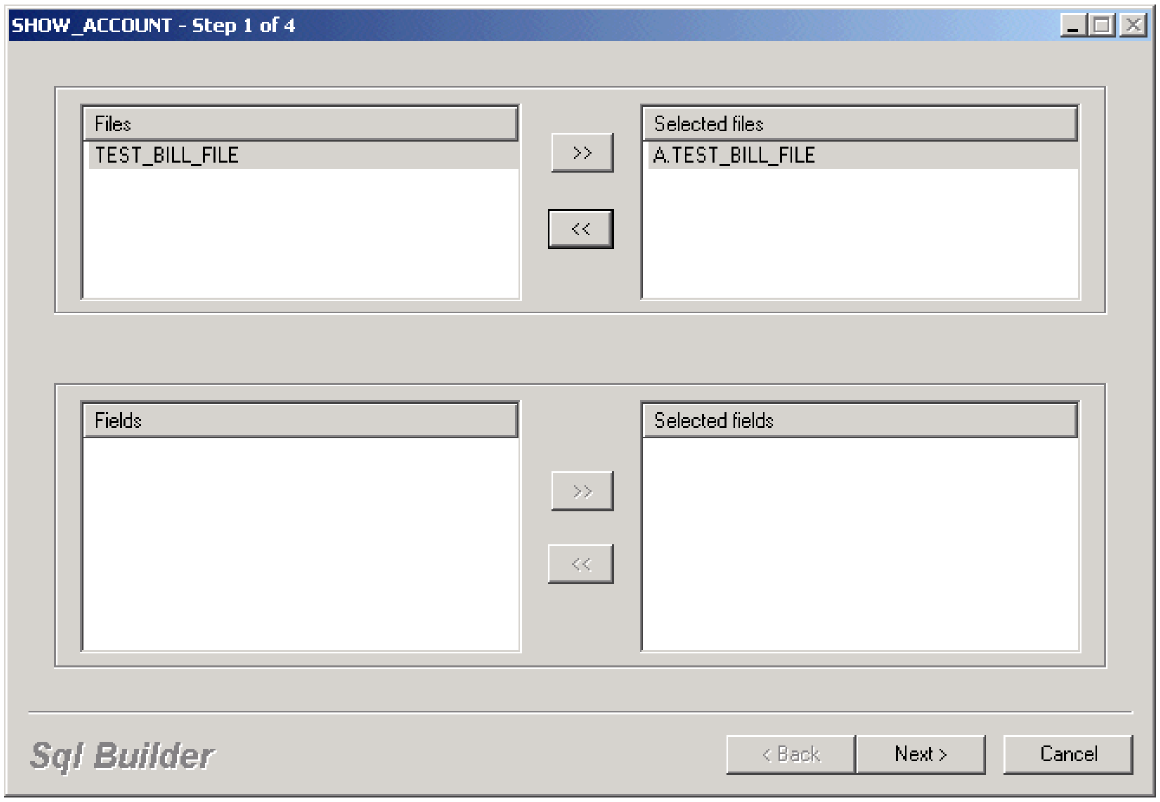
Use the SQL Builder to insert SQL code (query language commands for database access) into a rule. The SQL Builder allows you to customize the generated SQL code before adding it to the rule. The SQL Builder supports WHERE, CURSOR, GROUP BY, and ORDER BY subordinate clauses.
To use SQL Builder and add SQL to a rule, the following conditions apply:

- The *DBMS Usage* property of a rule must be set to *DB2* to open the SQL Builder. This indicates that the rule accesses a database ? not necessarily a DB2 database.
- The rule must access at least one file that owns at least one view, and the view must include at least one field.
- All files must have implementation names.
  You should also give implementation names to all views and fields. Otherwise the generated code has a placeholder where the implementation name should be.

Set the **View Usage** property on the View to File relationship (also known as the *File owns a View* relationship) to **Data View**.

To use the SQL Builder to insert an SQL command, complete the following steps:

1. Open the Rule Painter window for the rule.
2. Click on the *SQL Builder* button in the Text Editor - Tools toolbar or from the **Tools** menu, select **SQL Builder > Query** . The SQL Builder window is displayed.
   SQL Builder Query window



3. The list of files in your hierarchy, in that rule, are shown. Select a file and select the field, if any, in the lower portion of the window. Click **Next**.
4. Step through the windows and when done, click **Finish** and the code is inserted in the Rules source code in Rule Painter.

Verifying Rules

Make sure that you have specified the correct verification language before verifying a rule. To specify the verification language, complete the following steps:

1. Open the Rule Painter window for the rule.
2. Do one of the following to specify the verification language:
   - Right-click in the Rule Painter window and select *Verification Language* > *specific language* from the pop-up menu.
   - Select *Build* > *Verification Language* from the Construction Workbench.
     //
     You can select C, COBOL, CServer, CSharp, Java, JavaHTML, JavaServer, and OpenCOBOL.

To verify a rule, complete the following steps:

1. Open the Rule Painter window for the rule.
2. Right-click in the Rule Painter window and select *Verify* from the pop-up menu.

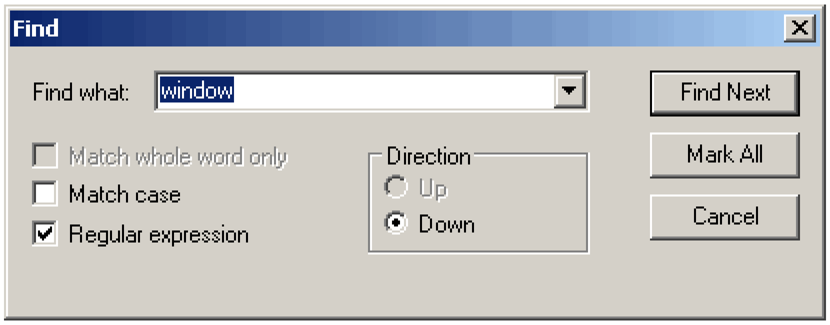
You can also verify a rule by clicking the *Verify* button ✓ in the Text Editor - Tools toolbar or by selecting **Build > Verify** from the Construction Workbench menu.
The system verifies the rule and displays a Verification Report on the *Verify* tab of the Output window. See Verify Tab for more information.
If there are errors, double-click the error message in the *Verify* tab. The cursor displays the associated line in the Rule Painter window.

To stop verification, click the *Stop Verification* button ✓ in the Text Editor - Tools toolbar.

Searching for Regular Expressions

A regular expression is a formula for matching strings that follow a pattern.
To search the code of a rule for regular expressions, complete the following steps:

1. With a rule created by Rule Painter open, select **Edit > Find**. The Find dialog box displays.
   Find dialog



Find what: window

[ ] Match whole word only
[ ] Match case
[✓] Regular expression

Direction: Up [✓ Down]

2. Type the expression you want to find in the rules code in the field and check the Regular expression box.
3. Click **Find Next** to find the next instance of the expression or **Mark All** to highlight every instance of the expression.

The Rule Painter search engine uses a set of regular expression patterns to do a search. The following table lists the special characters that you can use to match patterns with the regular expression object.

Regular Expression search methods

Character	**Example**	**Matches**
\	\d	Used to find literal or special characters. In the example, the pattern matches a digit.
^	^Lemon	The match must occur at the beginning of the line. In the example, the pattern matches lines that start with the string, "Lemon".
$	Lemon$	The match must occur at the end of the line. In the example, the pattern matches lines that end with the string, "Lemon".
	L*	Matches zero or more occurrences. In the example, the pattern matches a string of L's if they are present, otherwise an empty string.
+	L+	Matches one or more occurrences. In the example, the pattern matches a string of L's. If no L's are present, a match is not found.
?	L?	Matches zero or one occurrence. In the example, the pattern matches a single "L" if present, otherwise an empty string.
.	..	Matches any character except new lines. In the example, the pattern matches the first two characters.
\b	\bLe on\b	Defines the boundary for the word: The \bLe example matches any word in the expression that starts with "Le". The on\b example matches any word in the expression that ends with "on".
\B	\BLe on\B	Ensures the match is not on the boundary of a word: The \BLe example matches any word in the expression that contains "Le", but doesn't start with "Le". The on\B example matches any word in the expression that contains "on", but doesn't end with "on".
\d	\d	Used to match any single digit between 0 and 9. The example matches the first single digit in the expression.
\D	\D	Used to match any non-digit. The example matches the first non-digit in the expression.
\s	\s	Used to match any single white-space characters. The example matches the first white-space character.
\S	\S	Used to match any single non white-space character. The example matches the first non white-space character.
\w	\w	Used to match any alphabetic character, digit or underscore. The example matches the first alphabetic character, digit, or underscore.
\W	\W	Used to match anything other than an alphabetic character, digit or underscore. The example matches the first occurrence of anything other than an alphabetic character, digit, or underscore.
\xnn	\x41	Used to match the ASCII character represented by the hexadecimal number nn. The ASCII character for 'A' is 65. The hexadecimal value is 41. The example matches the first occurrence of an A.
[]	[aeiou]	Used to match any of the enclosed characters.
[^]	[^aeiou]	The first match that is not in the enclosed characters.
[c-c]	[a-z]	Used to match a range of characters. The example matches the first lowercase character.
{n}	w{3}	Used to match n occurrences of the previous character. The example matches "www" if three or more W's are found together in the string.

{n,}	w{3,}	Used to match at least n occurrences of the previous character. If there were four w's in the example, it would match "wwww".
{m,n}	w{3,5}	Used to match between m and n occurrences of the previous character. The example tries to match between three and five w's.
()	(em)	Brackets are used for grouping. The value found is stored, and may be used later for referencing. This technique is called back referencing.
|	a|e	Matches either of the character to the side of the operator. In the example, the first match of either an a or an e is found.

Matrix Builder

Use the Matrix Builder to construct a matrix that shows the association between objects. By using Matrix Builder, you can find objects with similarities, such as an entity used by several processes. With this information you can coordinate the different processes to ensure the data associated with the entity is sufficient. The following topics are discussed in this section:

- Creating or Opening a Matrix
- Creating and Editing a Template
- Adding Rows or Columns
- Moving Rows or Columns

Matrix Builder displays entities as matrix elements in rows and columns and displays relationships as the intersection data in cells. If the relationship represented in a cell has properties, flags can be displayed in the cell area.

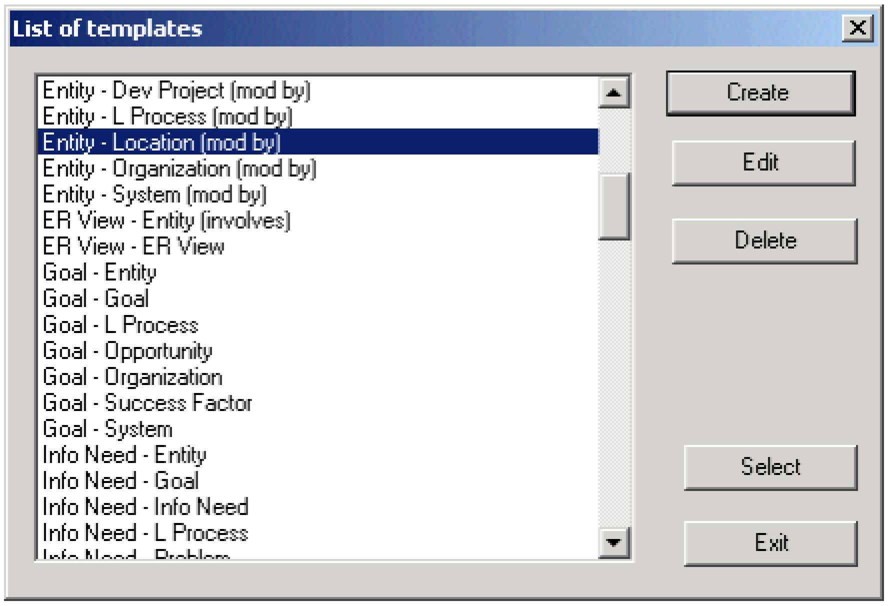
To view or modify the default Matrix Builder options (for example, font selection or header orientation), select **Tools > Workbench Options** from the Construction Workbench menu, then click **Tools > Matrix Builder**. See Matrix Builder Options for more information.

There is also a toolbar available for the Matrix Builder. See Matrix Builder Toolbar.

Creating or Opening a Matrix

To create a new matrix, select **File > New** , and from the Create New dialog, select **Matrix** . A dialog box displays a list of templates. Select a template and click **Select**.

List of templates dialog



A blank matrix tab displays in the Work Area.

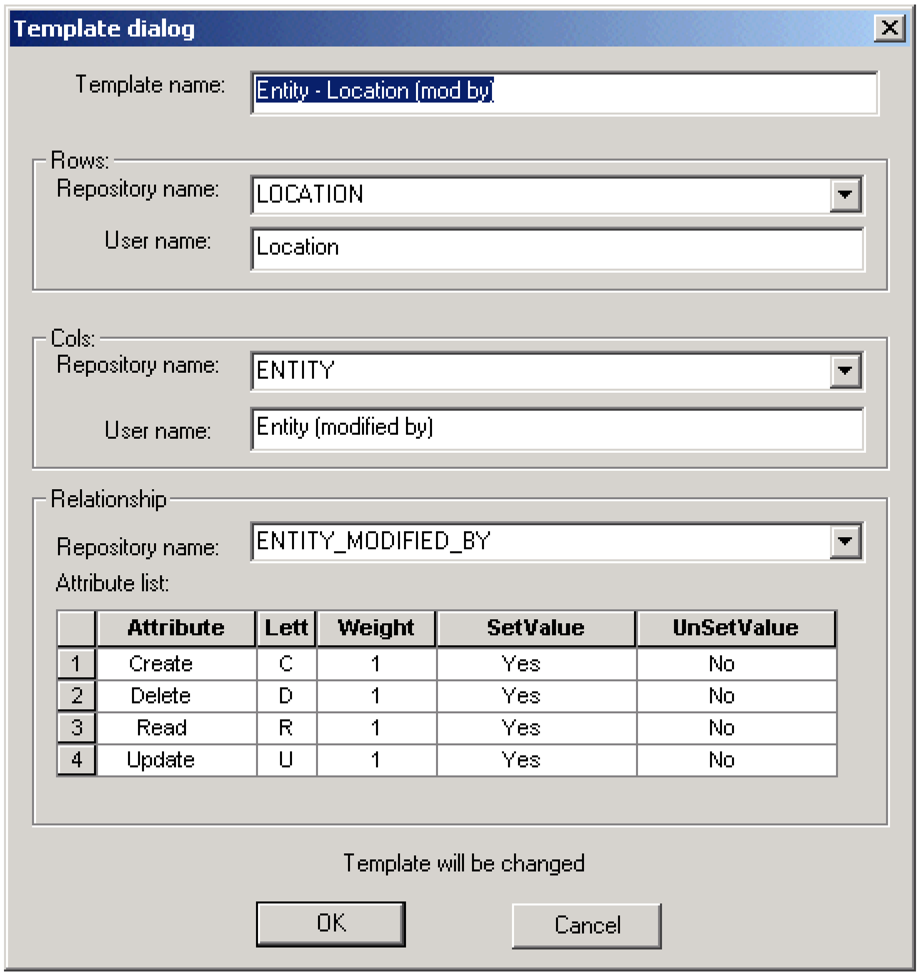
To open an existing matrix, select **File > Open**. From the Open dialog, select **Matrix** , and either type the name of the matrix object or do a query to find the matrix you want. Click **OK**. The matrix tab displays in the Work Area.

If you have a matrix tab open, select **View > Templates** to open the list of templates.

Creating and Editing a Template

From the List of Templates dialog box (List of templates dialog), click **Create** or **Edit**. The Template dialog box displays. If you click **Create**, the dialog box displays all blank fields for you to create a new template. If you click **Edit** , the dialog box displays the existing values, and you can edit them as shown in Edit template dialog example.

Edit template dialog example



Template name: Entity - Location (mod by)

Rows:
Repository name: LOCATION
User name: Location

Cols:
Repository name: ENTITY
User name: Entity (modified by)

Relationship
Repository name: ENTITY_MODIFIED_BY
Attribute list:

	Attribute	Lett	Weight	SetValue	UnSetValue
1	Create	C	1	Yes	No
2	Delete	D	1	Yes	No
3	Read	R	1	Yes	No
4	Update	U	1	Yes	No

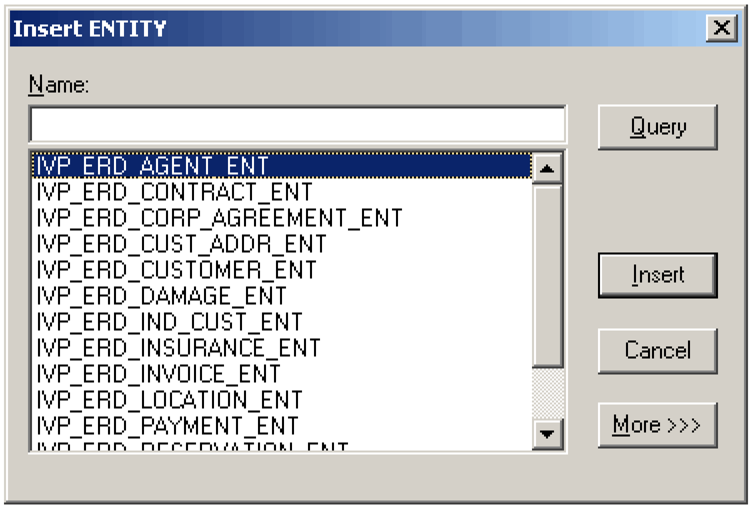
Template will be changed

If you have a Matrix window open, select **View > Current Template** to view the current template, but you can only edit the Attribute List area.

Adding Rows or Columns

To add a row or column to a matrix, double-click a blank header entry. The system displays the Insert Entity dialog box for you to query and select an entity from the repository (see Insert Entity dialog).

Insert Entity dialog



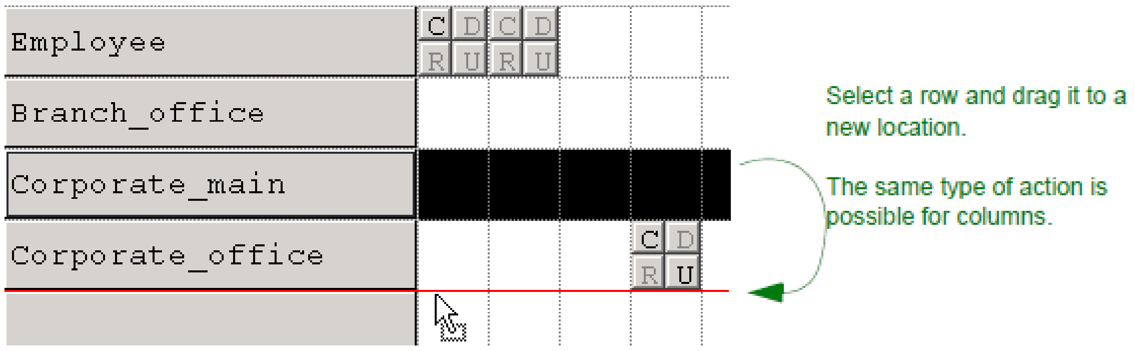
IVP_ERD_AGENT_ENT
IVP_ERD_CONTRACT_ENT
IVP_ERD_CORP_AGREEMENT_ENT
IVP_ERD_CUST_ADDR_ENT
IVP_ERD_CUSTOMER_ENT
IVP_ERD_DAMAGE_ENT
IVP_ERD_IND_CUST_ENT
IVP_ERD_INSURANCE_ENT
IVP_ERD_INVOICE_ENT
IVP_ERD_LOCATION_ENT
IVP_ERD_PAYMENT_ENT

To add a new or existing object to the matrix, click the header. Type the name and press Enter. The system readjusts the header size as necessary. If the object exists, the system adds it. If it does not and the option to create new objects is set, a new object is created.

Moving Rows or Columns

You can move a row or column by selecting it and dragging it to a new location. When you release the mouse button, the row is inserted under the red line for rows and before the red line for columns.

Moving a row


Employee
C D C D
R U R U
Branch_office
Corporate_main
Corporate_office
C D
R U
Select a row and drag it to a new location.
The same type of action is possible for columns.

Sorting Rows or Columns

Rows or columns can also be moved by sorting them based on the amount of filled cells. With a row or column (that has data) selected, select **Edit > Sort** or right-click the row or column and select **Sort**. Rows or columns with more filled cells move to the top or left.

Deleting Rows or Columns

With a row or column selected, you can remove it from the matrix or remove the entity from the repository. To remove the row or column from the matrix, select **Edit > Clear**, or right-click the row or column and select **Clear**. To remove the entity from the repository, select **Edit > Remove**, or right-click the row or column and select **Remove**.

Viewing Properties

Right-click a cell to display the relationship properties (or attributes). The letters indicate the functions or properties for that cell. In Moving a row, the template properties are C (create), U (update), D (delete), and R (read). Clicking the buttons enables and disables that function. See Creating and Editing a Template.

Set Builder

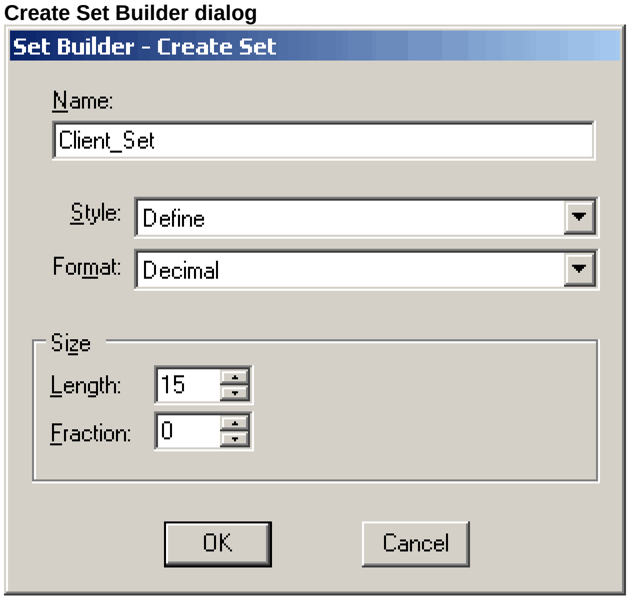
Use the Set Builder to define the members of a set object. The members of a set are symbol objects.

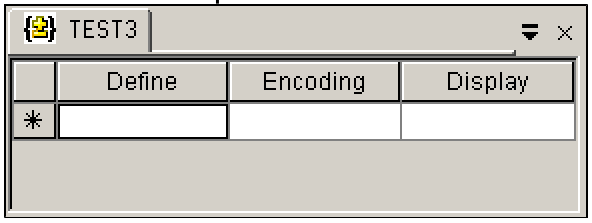
- Creating a Set
- Using Set Builder

For more details about the Set object, see Set.

Creating a Set

To create a new set, complete the following steps:

1. Select **File > New > Set**.
   The Create Set dialog box is displayed as shown in the following figure.
   Create Set Builder dialog
   
   Set Builder - Create Set
   Name: Client_Set
   Style: Define
   Format: Decimal
   Size
   Length: 15
   Fraction: 0

2. Click **OK** .
   The Set Builder window displays as shown in Set Builder tab example.
   Set Builder tab example
   

   Each symbol in the set is represented by a row in the Set Builder.

Use the following table to help you determine the data to type in each field of the Set Builder window:

Set Builder fields

Field	Description
Define	Identifier used in the Rules Language code that references the symbol
Encoding	Identifier used in the application database
Display	Default display value for the identifier unless other languages are specified

Fill in the **Define** , **Encoding** , and (if not a Define Set) **Display** fields for the symbol.

✓ Your system administrator sets which languages (if any) are available in your currently active repository.

Only the Encoding field is restricted to match the data type specified for the set. For example, if the data type is integer, this field cannot contain a decimal or character string value. Also, encoding fields for sets that are of type integer cannot contain the double byte representation of an integer.

To define an error set, use the **Text/Default** choice in the **Selected** menu to enter error message text for each symbol.

If necessary, select **Edit > Delete > Symbol** or **Edit > Delete > Set** to delete a row or the set from the Set Builder window and to delete the symbol it defines from the set or to delete the set.

Repeat these steps until you have defined the members of the set.

To add the Set to the hierarchy, select **View > Navigate to Hierarchy**.

Select **Commit** under the **File** menu to save the symbols.

Using Set Builder

The Set Builder is only available for sets whose Style *is not* **values** . The Set Builder cannot be used to edit values sets. To edit a field in Set Builder, press **F2**.
To view an existing set in the Set Builder (see Set Builder tab example), select one of the following methods:

- Select **File > Open > Set** and browse for the set object.
- In the Hierarchy Window, double-click the set object.
- Right-click the set object in the hierarchy and select **Open Set** from the pop-up menu.

To sort the symbols of a set by Define, Encoding, Display, or Sequence Number, do one of the following:

- While the Set Builder is in focus, select **Edit > Sort by**.
- Right-click inside the Set Builder, and select **Sort by** from the pop-up menu.
- Double-click the header of the column you want to sort by. An icon on the column header shows whether the set is sorted by ascending or descending order. If there is no icon on any header, the set is sorted by sequence number by default.

If you change the Set Style, make sure to fill out the required fields for the new style. For example, a Define Set requires Define and Encoding fields; however, if you change the style to Lookup Set, Display field must also be entered.

To view or modify the default Set Builder options (for example, font selection or whether to confirm a delete), select **Tools > Workbench Options** from the Construction Workbench menu, then click **Tools > Set Builder**. See Set Builder Options for more information.

Window Painter

The Window Painter is a construction tool used to create the graphical user interface (GUI) for each window for your application. You can specify the properties for each object defined in the window and determine the placement and operation of the objects.
When you drag a field from the hierarchy and drop it onto an existing object on the Window Painter, a message box pops up so that you can choose whether to create the link to the existing object or create a new object. You can also choose to do the same action from the dialog. Also, in *Tools* > *Workbench Options* > Tools, the Window Painter set of options, you can set the action to drag and drop using three check boxes. See Window Painter Options for more information.
This chapter discusses the following topics:

- Using Window Painter
- Managing a Window
- Managing Objects in a Window
- Understanding Objects in a Window
- Using 3270 Window Painter

Using Window Painter

The following tasks are involved in using the Window Painter:

- Opening and Closing Window Painter
- Understanding the Window Painter Interface
- Using Window Painter Toolbars
- Setting Window Painter Options
- Setting Font Mappings at Execution Time
- Linking a Window to an Associated Event Procedure in Rule Painter

AppBuilder does not store the screen resolution, so the position and resolution of the window at design time might differ from the position and resolution at execution time. However, if the end user moves a window, AppBuilder stores the position and resolution, so that when the end user reopens the window, the pop-up position is the same.

Opening and Closing Window Painter

Use standard Windows conventions for opening and closing a Window Painter tab.

To open Window Painter, do one of the following:

- Double-click a Window object in the hierarchy.
- Select a Window object in the hierarchy and right-click *Open Window* from the context-sensitive menu.
- Insert a Window object from the *Insert* menu.
- Create a new Window object from the **File > New** menu or open an existing window object from the **File > Open** menu.

To close Window Painter (with the tab in focus), do one of the following:

- Press **Ctrl-F4.**
- Click the *x* in the upper right corner of the tab view.

To maximize Window Painter, double-click the icon in the upper left corner of the tab view.

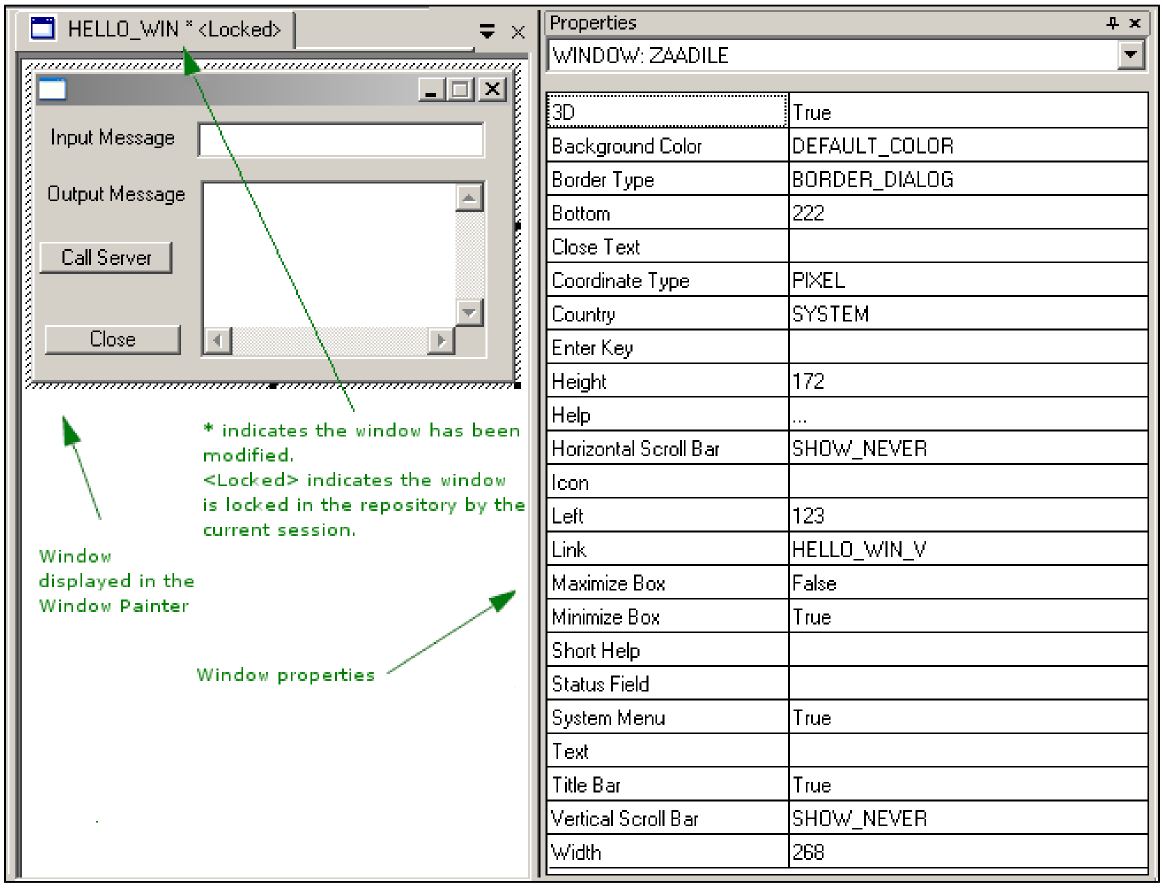
Understanding the Window Painter Interface

Open a Window object from the Hierarchy Window to activate Window Painter. The Window Painter tab displays in the Construction Workbench work area. You can edit the Window directly by moving, modifying, or inserting objects or by opening the Properties window for the Window object or any of the objects in the Window. When the Window object or any of its objects are in focus, the toolbars and menus contain options specific to Window Painter. See Properties Window for more information.

To edit the properties of a window control, you can either use the Properties window or you can move the display in the Panel Layout window (See Specifying Window Layout).

When you insert, move, or otherwise modify objects in the window, the title bar of the window changes to signify that the object has been changed. There are two types of change indications in the Window Painter. If you have made a change that has not been committed to the repository, an asterisk (*) is displayed in the title bar. The word Locked is displayed in the title bar indicating that the object is locked in the repository by the current session and was modified. It cannot be changed by anyone else until you commit your changes. This is important when using a workgroup repository. For more information about locking objects, see the *Repository Administration Guide for Workgroup and Personal Repositories* .

Parts of Window Painter


HELLO_WIN * <Locked>
Input Message
Output Message
Call Server
Close
* indicates the window has been modified.
<Locked> indicates the window is locked in the repository by the current session.
Window displayed in the Window Painter
Window properties
Properties
WINDOW: ZAADILE

3D	True
Background Color	DEFAULT_COLOR
Border Type	BORDER_DIALOG
Bottom	222
Close Text	
Coordinate Type	PIXEL
Country	SYSTEM
Enter Key	
Height	172
Help	...
Horizontal Scroll Bar	SHOW_NEVER
Icon	
Left	123
Link	HELLO_WIN_V
Maximize Box	False
Minimize Box	True
Short Help	
Status Field	
System Menu	True
Text	
Title Bar	True
Vertical Scroll Bar	SHOW_NEVER
Width	268

Using Window Painter Toolbars

The Window Painter has two toolbars so you can quickly add and manipulate interface objects to the window.

- Window – Layout Toolbar

- Window – Objects Toolbar

To display the toolbars, complete the following steps:

1. Select **View > Toolbars** from the Construction Workbench menu.
2. In the Toolbars submenu, select the toolbar to display (**Window Layout** or **Window Objects**).

or

1. Right-click the toolbars area. A contextual menu displays the available toolbars.
2. Select the toolbar to display (**Window Layout or Window Objects**).

Many of the functions available from the toolbars are also available in menus. Refer to Setting Window Painter Options.

Setting Window Painter Options

Use the **Window Painter** set of options of the Workbench Options window to customize the Window Painter. You can specify field linking instructions and several window display options such as color.

To access these options, select **Tools > Workbench Options** from the Construction Workbench menu. Then click **Tools > Window Painter** in the Workbench Options window. Refer to Window Painter Options for more information.

Setting Font Mappings at Execution Time

A configuration file named wpmodel.txt, in the AppBuilder client runtime, controls the font mappings. This file externalizes the font mappings used by the client runtime. For Window Painter, you can select the font definitions or add your own font definitions by editing this file. Refer to Font Mapping at Execution Time.

Linking a Window to an Associated Event Procedure in Rule Painter

You can use the functionality in Window Painter to define the behavior of objects in Windows and link Event Procedures to Rules with a few clicks. When you place an object, such as a push button or a field, you can choose from a list of valid events and parent rules to apply to the object in the Window. You can use this function to create a new event for an object or bring the user to the chosen event in the Rule Painter. The key tool that provides all this functionality is the Event Wizard. You can link a Window or an object in a Window, such as a push button or a field, to an associated Event Procedure in Rule Painter. The Window object must be a child of a Rule.

To link a Window to an associated event procedure in Rule Painter, complete the following steps:

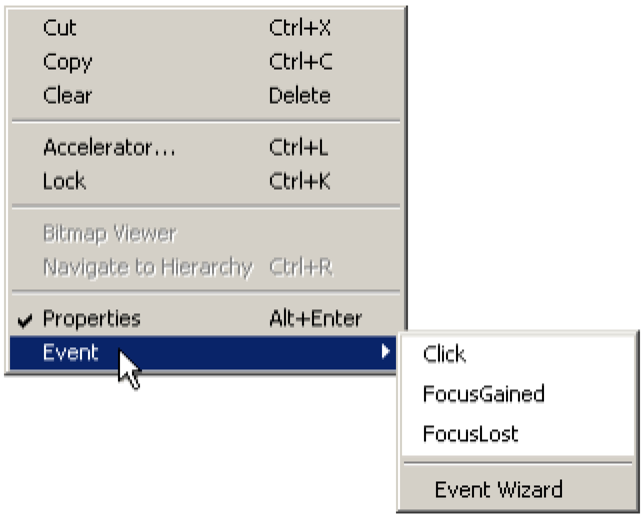
1. In Window Painter, right-click a Window object. A popup menu displays.

When the window is owned by one rule, the Event submenu displays the valid events for that Window object and the Event Wizard entry.

- Each menu item is a valid event for the selected object in the Window Painter.
- If no object is selected, then the menu items are the events for the window itself.
- If one event is already defined in the parent rule, that event displays in bold font.
- Choosing a menu item creates the event or goes to that event depending on whether the event definition exists in the parent rule or not.
- Choosing the event menu items leaves Window Painter and activates the Rule Painter.

When the window is owned by several rules, only the Event Wizard entry displays.

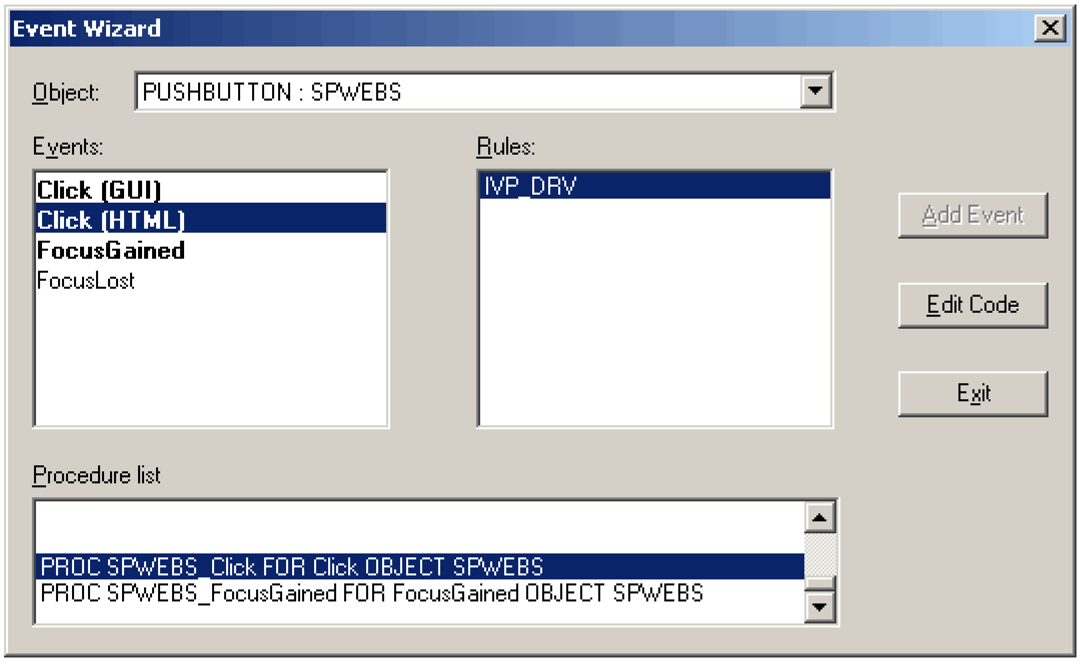
Window Painter popup menu



2. Select **Event**. Select **Event Wizard** to bring up the Event Wizard dialog box.

The Event Wizard dialog box has a combo box with all controls in the window panel, including the window panel itself. Three list boxes display choices to program in your application.

Event Wizard dialog



Object: PUSHBUTTON : SPWEBS

Events:
- Click (GUI)
- Click (HTML)
- FocusGained
- FocusLost

Rules: IVP_DRV

Procedure list:
```
PROC SPWEBS_Click FOR Click OBJECT SPWEBS
PROC SPWEBS_FocusGained FOR FocusGained OBJECT SPWEBS
```

A combo box in the Event Wizard dialog displays all legal events for a Window object. A list box displays all parent Rules of the current window panel. There is a list box to hold all the procedure definitions in one Rule.

- Events: This list box displays all legal events for the selected window objects.
- Rules: This list box displays all parent rules of the current window panel. The selected Rules appear in the controls combo box list.
- Procedure list: This list box displays all the procedure definitions in the selected Rule.

If the procedure for a certain event is found, the **Edit Code** button is enabled. Otherwise, it is grayed out. If the procedure for that event is not found, the **Add Event** button is enabled. Otherwise, it is grayed out.

3. Select the Object to which you want to apply an event.

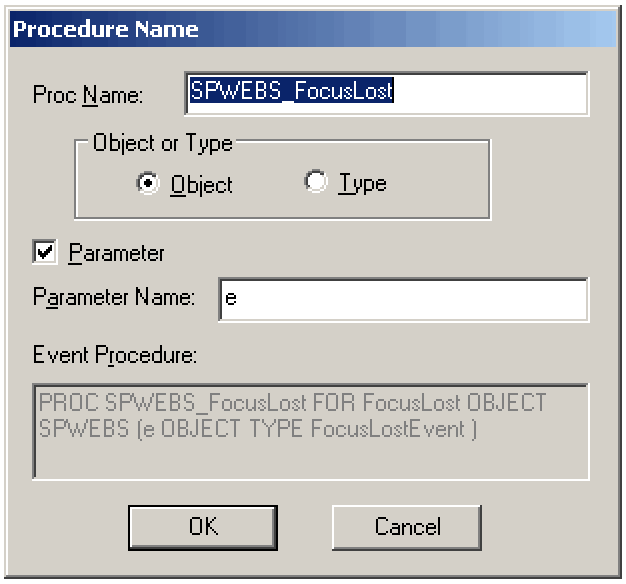
4. Select an event in the Events list box. If a procedure for that event is found, that item is highlighted. If the window has more than one parent, select the parent rule from the Rules list box.

⚠ If you change the selection in the Procedure list, the Object in the Object combo box and the event list are both refreshed.

5. Do one of the following, depending on whether the Edit Code button or the Add Event button is enabled.

- Click the **Edit Code** button to open the Rule Painter and move the cursor to the procedure. Edit code leaves the Event Wizard and activates the specified rule.
- Click the **Add Event** button. The Procedure Name dialog displays so that you can type the name for the procedure. You can choose whether or not to include Object, Type, or Parameter in the procedure definition (see Procedure Name dialog sample).

Procedure Name dialog sample



Proc Name: SPWEBS_FocusLost

(● Object) ○ Type

☑ Parameter

Parameter Name: e

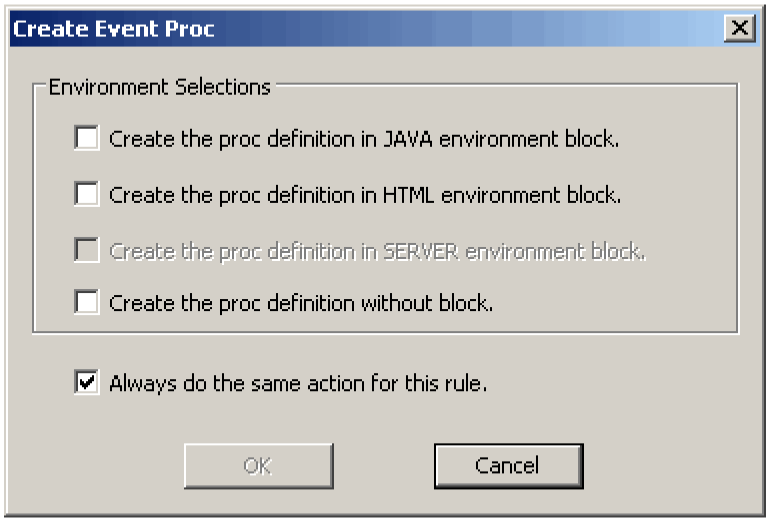
Event Procedure:

PROC SPWEBS_FocusLost FOR FocusLost OBJECT SPWEBS (e OBJECT TYPE FocusLostEvent)

6. Select only one option from Object and Type. Click *Parameter* to include Parameter in the procedure definition. The procedure name can be blank.

7. Click **OK** to create the procedure definition in the Rule source file. The Create Event Procedure dialog displays (see Create Event Procedure dialog sample).

Create Event Procedure dialog sample



☐ Create the proc definition in JAVA environment block.

☐ Create the proc definition in HTML environment block.

☐ Create the proc definition in SERVER environment block.

☐ Create the proc definition without block.

☑ Always do the same action for this rule.

8. Specify the environment block where the procedure will be created. You can also choose to create the procedure outside the environment.

9. Click **OK** to finish creating the procedure.

Managing a Window

The following tasks are involved with managing a window in the Window Painter:

- Creating a Window
- Modifying a Window
- Customizing Colors
- Previewing a Window
- Previewing or Deleting a Window Template
- Generating HTML from a Window
- Checking a Window for Errors
- Specifying Window Layout
- Setting Layout Guides
- Defining Short Help (Tooltip)
- Defining Online Help (F1 Help)

Creating a Window

When designing windows, you can create a window from scratch or create a window using an existing window template. Some processes, such as forward engineering, generate windows so that you do not have to start from scratch.

To create a window based on an existing template, refer to Creating a Window from a Template.

To create a window template, refer to Creating a Window Template.

The name of the window is stored in the panel file associated with the window. If the name of the window is changed without having Window Painter open, the panel file is not updated automatically. If the Window Painter is open and the window name is changed, the change is stored in the panel file when you commit.

Creating a Window from Scratch

To create a new window from scratch, complete the following steps:

1. Click the **New Object** button in the Standard toolbar (see Standard toolbar). The Create New dialog displays.
2. Select **Window** . The Template area of the Create New dialog displays as shown in Create New Window dialog. The default choice under Template is Blank Window.
3. Click **OK**. The Window Painter is displayed with a new blank window.

The wpmodel.txt file defines the attributes for all window painter objects. All default values for all attributes of every kind of control in Window Painter can be changed manually in the wpmodel.txt file. The following attributes are the most commonly used:

- COUNTRY
- FONT
- BACKGRNDCOLOR_NAME
- FOREGRNDCOLOR_NAME
- BACKGRNCOLOR
- TEXT
- JUSTIFICATION
- HELP

If you modify the contents of wpmodel.txt incorrectly, some actions in window painter might fail.

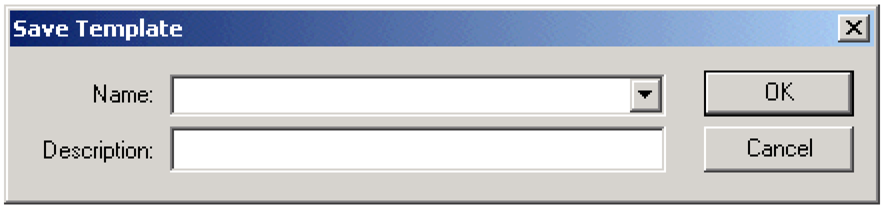
Creating a Window Template

Any window you create can be saved as a template for creating additional windows. To create a window template, complete the following steps:

1. Create or open a window in Window Painter to use as a template.

2. Select **File > Save As Template** from the Construction Workbench menu. The Save Template dialog displays (see Save Template dialog).

Save Template dialog



Save Template

Name:

Description:

OK

Cancel

3. Use the descriptions in Save Template dialog fields to enter data in each field on the window.

Save Template dialog fields

Field	Description
Name	Name of the template. Use the drop-list to select from existing window templates.
Description	Description of the template.

4. Type the necessary information in the Name and Description fields and click **OK**.

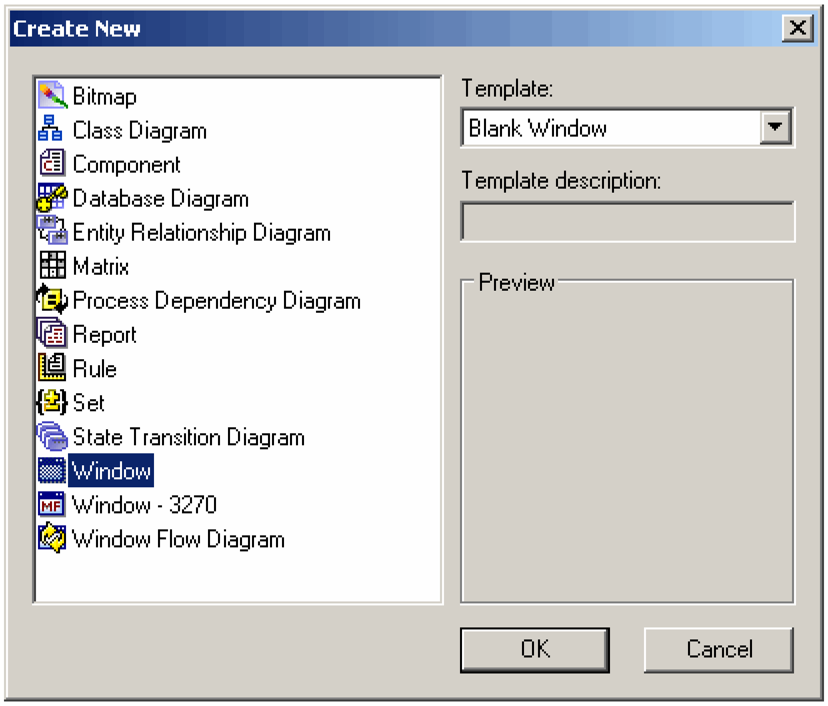
Creating a Window from a Template

To create a new window from a template, complete the following steps:

1. Click the **New Object** button in the Standard toolbar. The Create New dialog displays.
You can also create a new window from a template by selecting **File > New** from the Construction Workbench or by pressing **Ctrl+N**.

2. Select **Window**.
The Template area of the dialog displays as shown in Create New Window dialog.

Create New Window dialog



Create New

- Bitmap
- Class Diagram
- Component
- Database Diagram
- Entity Relationship Diagram
- Matrix
- Process Dependency Diagram
- Report
- Rule
- Set
- State Transition Diagram
- Window
- Window - 3270
- Window Flow Diagram

Template: Blank Window

Template description:

Preview

OK

Cancel

3. Use the **Template** drop-down list to select a template.
The description of the template appears in the Template description field, and a preview of the window appears in the preview box.

4. Click **OK**. The Window Painter displays with a new window object, based on the selected template.

Modifying a Window

To resize a Window, complete the following steps:

1. Click the window in the Window Painter.
2. Click and drag one of the window's handles to resize the window, as needed.

You can also resize the window by changing the window's *Height* and *Width* properties in the Properties window.

To modify the Window's properties, complete the following steps:

1. Display the Properties window for the Window object properties (see Parts of Window Painter).
2. Type the necessary information in the fields and press **Enter**. See Understanding Common Window Object Properties for more information on each property.
3. Click **File > Commit** from the Construction Workbench menu to commit changes to the repository.

Customizing Colors

There are two processes involved in customizing the colors of a window in a project. The first process is creating a customized color and adding it to the scheme. The second process is selecting the customized color from the attribute table on the right.

To add a custom color to the attribute table, complete the following steps:

1. Select **Tools > Workbench Options**.
2. Click **Tools > Window Painter** in the Workbench Options dialog to display the Window Painter set of options.
3. Click the New button in the Colors section. A placeholder appears for the name of the new color in the Color list, with a button with three dots besides it.
4. Click the **...** button. The Color dialog appears.
5. Select or define a custom color and click *Add to Custom Colors* .
6. Enter a name for the new color in the Color list and click *OK* .

To customize the color of a window, complete the following steps:

1. Display the Properties window for the Window object, if it is not already displayed.
2. Double-click the Background Color field in the Properties window. A drop-down list of all available background colors appears.
3. Scroll down the list until you find the customized color that you created. Select that color.

⚠ You can also select Custom in the drop-down list from Properties window to create a custom color.

Previewing a Window

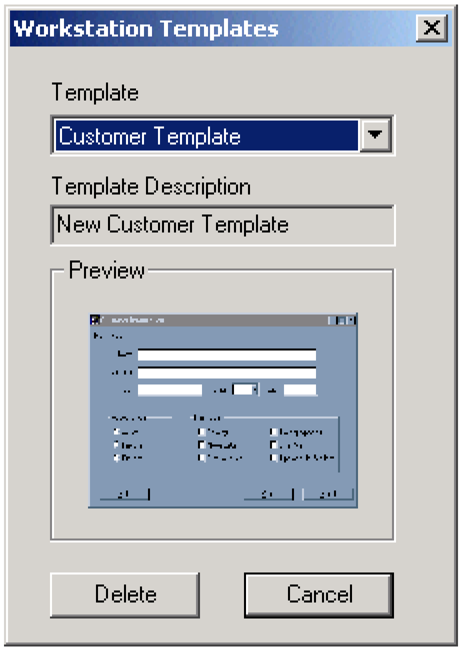
To preview the runtime version of the window, complete the following steps:

1. Open the window in Window Painter.
2. Click the **Preview** button in the Window Layout toolbar (see Window – Layout toolbar). The system displays the window end-user will see it during runtime.
   You can also preview a window by selecting **Layout > Preview in Runtime** (or use the keyboard shortcut **Shift+V**).

Previewing or Deleting a Window Template

To view the collection of window templates, from the Construction Workbench, with a window selected in Window Painter, select **View > Templates**. The Workstation Templates dialog is displayed as shown in Template viewer.

Template viewer



To preview a window template, select a template from the drop-down list. You can view the description and the preview image of the window template.

To delete a window template, select it and click **Delete**.

Generating HTML from a Window

When you create a window in the Construction Workbench using the Window Painter, you can generate HTML forms from the window and then customize or edit them using an HTML editor. Within AppBuilder, you can choose a third-party HTML editor and edit the HTML interface for your application. Refer to HTML Generation for a complete discussion about generating HTML from a window and modifying the HTML.

⚠ AppBuilder does not provide Print or Print Preview for HTML windows developed in the Construction Workbench. Construction Workbench can use a third-party HTML editor for HTML editing. Use the same editing tool for print previewing and printing.

Checking a Window for Errors

To verify a window, select the window in Window Painter and select *Build* > *Verify* . The focus changes to the Verify tab in the Output window. Window Painter issues error messages for conditions that might cause the window to execute incorrectly and warning messages for conditions that might cause unexpected behavior. The following conditions might generate a warning:

- Objects have duplicate system identifiers (HPSIDs)
- Object has no link, if the object type is Edit Field, Combo Box, Radio Button, Check Box, or Bitmap
- Object is placed partially or completely outside of the window
- Objects in the window overlap

A password field linked to any field other than Character, DBCS, Mixed, VarChar is an example of what can cause an error message.

The line "Window Painter Verify" is displayed and followed by any error messages. The number of errors and warnings is totaled. At the end of the list is the line "Verify Complete". For a description of the Output window, refer to Output Window.

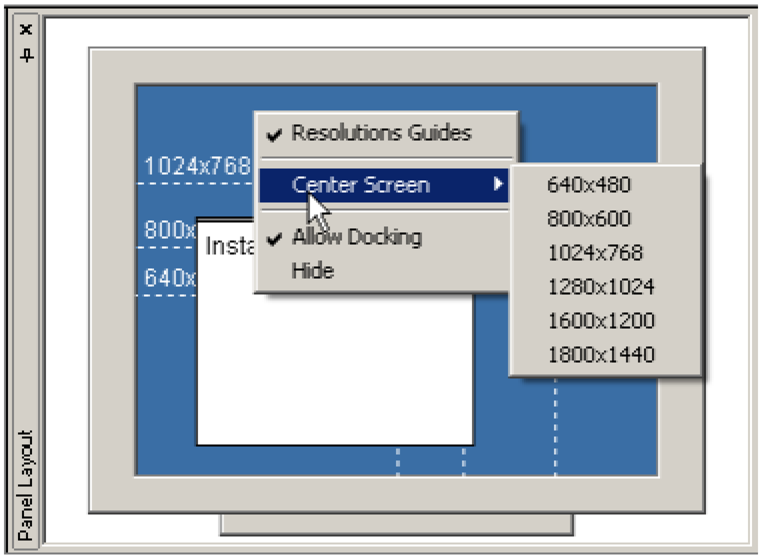
Specifying Window Layout

You can determine how the window is displayed on the target machine. To specify the window's layout and location on the end-users's screen:

1. Open the window in the Window Painter.

2. Right-click the window and select **Panel Layout** from the menu. The Panel Layout window is displayed as shown in Panel Layout window. You can also specify the window layout by selecting **View > Panel Layout** from the Construction Workbench menu or by pressing **Shift+L**.

Panel Layout window



3. Click and drag the window icon to the desired location on the screen.

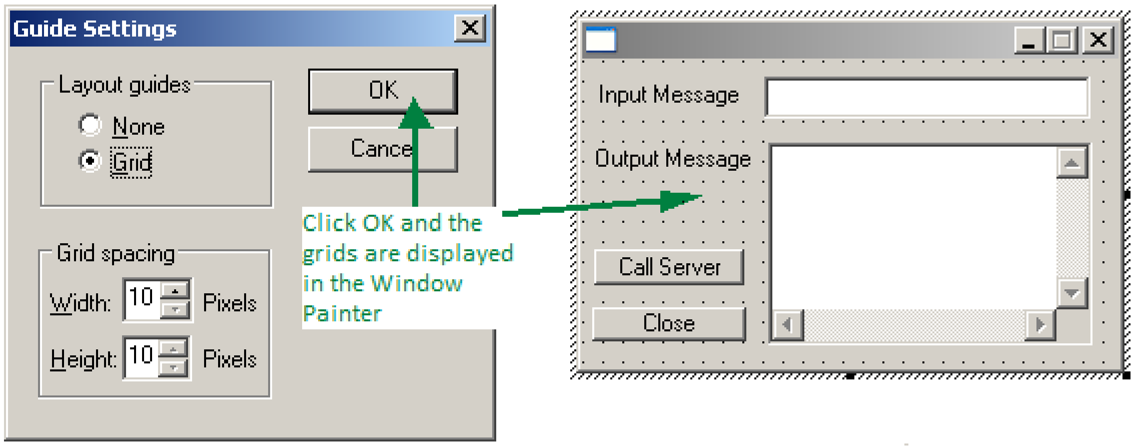
- To toggle the resolution guidelines on/off, right-click the window and select **Resolution Guides**.
- To automatically center the window for a specific screen resolution, right-click the window, select **Center Screen**, and select the desired resolution (in pixels).

The window inside the Panel Layout is positioned relative to the bottom left of the screen. The coordinates of Bottom and Left of the window in the screen are shown in the status bar when the window is selected in Window Painter. To display left and top coordinates, select the **Show objects in (Left, Top) coordinates in status bar** check box in the Window Painter set of options on Workbench Options dialog. For information about the status bar, see Understanding the Window Painter Interface. The bottom and left can also be modified in the Properties window for that Window object.

Setting Layout Guides

Select **Layout > Guide Settings** or press **Ctrl+G**. Set the guide format and the grid spacing in the Guide Settings dialog (see Guide Settings dialog).

Guide Settings dialog



Defining Short Help (Tooltip)

For fields and buttons in a window or for the window itself, you can define a tooltip (for Java applications) or status line help (for C applications). The text that is displayed for these is defined in a property of the object called Short Help.

Defining Online Help (F1 Help)

For fields and buttons in a window or for the window itself, you can define online help that is displayed when the user presses *F1* . The text that is displayed for these is defined in a property of the object called Help.

See the *Developing Applications Guide* for instructions on how to create online help for a window or window controls.

Managing Objects in a Window

The following is a list of the tasks involved in managing objects in a Window:

⚠ In Java code (for thick client and servlet), you must define unique system identifiers (HPSIDs) for all window controls. If you have existing C applications that can use duplicates, be aware that changing to Java requires creating unique system identifiers.

Inserting an Object in a Window

To insert an object in a window, complete the following steps:

1. Open the window in the Window Painter.
2. Click the appropriate object button in the Window Objects toolbar (Window – Objects toolbar) and move the cursor to the Window Painter tab in Work Area. The cursor changes to a cross (+).
   You can also add an object using the *Insert* menu.
3. In the Window Painter tab, click inside the window to add the object to the window. The system places an image representing the object at that location in the window.
   - To move the object, see Moving Objects in a Window.
   - To resize the object, see Resizing a Window Object.

See Understanding Objects in a Window for a list of objects that can be inserted into a window.

You can also insert an object into a window by dragging and dropping it from the application hierarchy in the Hierarchy window to the window in the Window Painter. For example, you can drag a Field object that has a View for a parent in the hierarchy into the window, and a message box that provides you with the choice to either create a link to an existing object or to create a new object appears. If you choose to create a link, it becomes an Edit Field and is linked to that view. For more information about dragging and dropping, see Dragging and Dropping.

Viewing and Modifying Window Object Properties

Window Painter object properties determine the appearance and behavior of the object. You can view and modify a property of an object in the Properties window, an example of which is shown in Properties of a window object. From the Properties window, you can type a value or select a value from the drop-down list. After making necessary changes, click **File > Commit** from the Construction Workbench menu to commit changes to the repository. See Properties Window for more information.

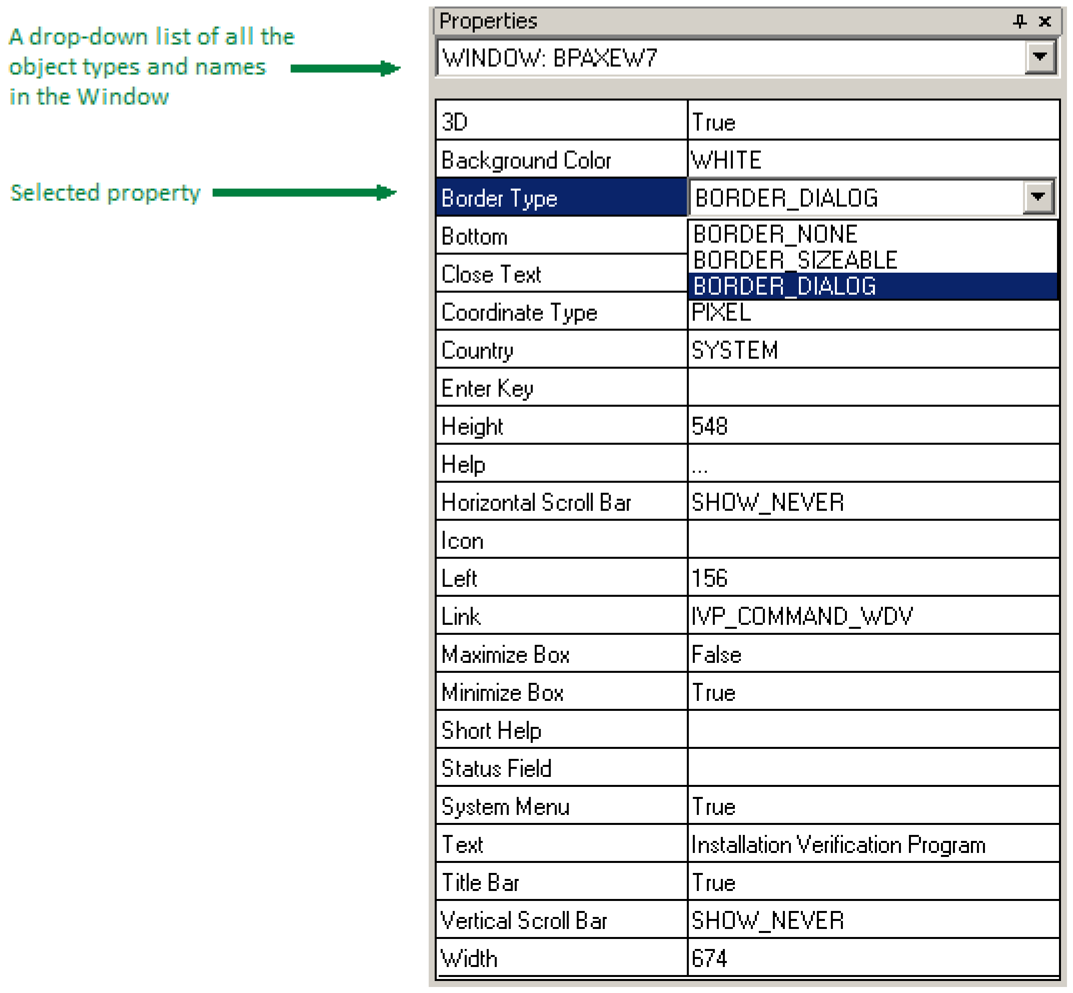
Whether you are designing windows or modifying templates, you can customize your object through its properties. For example, you can create an accelerator, add a bitmap, lock the object, or change the color. You can adjust the location of an object by changing the value in the Left and Bottom properties. See Moving Objects in a Window for other ways to move objects in a window.

The name of the object is at the top of the Properties window for that object. To view the object's properties, select the object in the Window Painter tab or select the name in the drop-down list. Some property values cannot be changed. If there are several values, a drop-down list is shown with the values.

You can scroll to view data that does not fit in the display area.

See Understanding Common Window Object Properties for more information on each property.

Properties of a window object


A drop-down list of all the object types and names in the Window
Selected property
Properties

WINDOW: BPAXEW7

3D	True
Background Color	WHITE
Border Type	BORDER_DIALOG
Bottom	BORDER_NONE
Close Text	BORDER_SIZEABLE
	BORDER_DIALOG
Coordinate Type	PIXEL
Country	SYSTEM
Enter Key	
Height	548
Help	...
Horizontal Scroll Bar	SHOW_NEVER
Icon	
Left	156
Link	IVP_COMMAND_WDV
Maximize Box	False
Minimize Box	True
Short Help	
Status Field	
System Menu	True
Text	Installation Verification Program
Title Bar	True
Vertical Scroll Bar	SHOW_NEVER
Width	674

Understanding Common Window Object Properties

Each Window Painter object has specific properties that control how the object is displayed in the AppBuilder application. The following table describes common object properties and identifies the Window Painter objects that have these properties. See Window Painter Objects and their Attributes for more details.

Common properties for Window Painter objects

Property	What Is Specified	Applicable to this object
3D	Whether the object is to be displayed as three-dimensional or not.	Window, Edit Field
AUTO_CALL	If AUTO_CALL is TRUE for a LISTBOX or SPREADSHEET, control returns to the invoking rule when you try to scroll beyond the first or last occurrence defined in the view data structure associated with the list box. The rule can then retrieve more data Set AUTO_CALL only if you do not use the component SET_VIRTUAL_LISTBOX_SIZE described in the AppBuilder documentation for your system.	Multicolumn list box (MCLB)
Background Color	The background color of the object. You can select a standard color from a menu of colors, or you can create a custom color. To create a custom color, select *Custom* from the drop-down list and use the Windows Color Picker to create the color.	most objects
Bottom	The coordinate for the bottom of the object. See Left for the other coordinate. You can type a value or move the object.	most objects
Check Mandatory Field	Denotes whether users must complete all mandatory objects before pressing the button.	Push Button
Close Text	The text returned to the invoking rule when the end user exits the window with *System* > *Close* function. In a secondary window, the system disables the Close function if the field is blank.	Window

Coordinate Type	Measurement type (*PIXEL* or *CHAR*) used for the object	Window
Country	Country variables and display properties used for the object	Window
Editable	Denotes whether the field is editable or is display-only.	Edit Field
Enabled	Denotes whether the field is enabled.	Edit Field
Enter Key	System identifier (HPSID) of the default Push Button for the window. When the user presses Enter, the system returns the system identifier to the invoking rule (as if the user had clicked the button).	Window
Font	The font for the object	most objects
Foreground Color	The foreground color of the object. See Background Color.	most objects
Form Fit	Denotes whether the field is form fit. See Resizing an Edit Field for more information.	Edit Field
Group	Objects that start a group. See Understanding Tab Groups.	All Window Painter objects accessible by pressing Tab.
Height	Height of the object (in the Coordinate Type unit of measure)	most objects
Help	The help displayed when the user presses F1	most objects
Horizontal Scroll Bar	Denotes whether the object has a horizontal scroll bar (**SHOW_ALWAYS** or **SHOW_NEVER**). Scrollbars are not supported in Java.	Window
HpsID	System identifier used by other AppBuilder objects to reference the object	all objects
Icon	The window icon	Window
Ignore Validation	Denotes whether the user can select the object even if some mandatory fields contain invalid data or if the window contains an error condition.	Push Button, Hot Spot
Immediate Return	Denotes whether the control generates an event at execution time when data is changed.	Edit Field, Combo Box, MCLB
Left	The coordinate for the left side of the object. See Bottom for the other coordinate. You can type a value or move the object.	most objects
Link	Denotes whether the object is linked.	most objects
Lock	Denotes whether the user can change any of the properties for the control.	most objects
Mandatory	Denotes whether the object is mandatory.	most objects
Maximize Box	Denotes whether the window displays a **Maximize** button.	Window
Minimize Box	Denotes whether the window displays a **Minimize** button.	Window
Password	The password is for a field.	Edit Field
Resize	Denotes whether resizing of field is allowed. When Resize is set to *False* , you can still adjust the width, but you must set the Resize property to True before you can change the height. See Resizing an Edit Field.	Edit Field
Row Select	The result of clicking a cell in the MCLB. When set to **True**, clicking the cell selects the entire row. When set to **False**, clicking the cell selects only the cell. MCLBs with an Extended Selection Type do not support Row Select = **False** .	MCLB
Selection Type	How rows and cells can be selected in the MCLB. In most cases, Row Select must be **True** . There are three possible values for Selection Type: Single, Multiple, and Extended. If Single is set, only one row or cell can be selected in the MCLB. If Multiple is set, you can select multiple rows *within the same column* of a MCLB when Row Select is **False**. To deselect a row or cell, press the CTRL key + a mouse click. If Extended is set, you can select a range of rows by selecting a row then, while holding the SHIFT key, selecting another row with the mouse. Cell selection is not supported in MCLBs with Extended selection. A mouse click can be used to deselect a row.	MCLB

Status Field	The HPS ID of an Edit Field in the window that shows the status line/Short help for the current control under the mouse cursor.	Window
Status Line/ Short Help	The help text that is displayed in the status line.	most objects
Stretch to Fit	A bitmap can be stretched to fit the size of Push Button if a bitmap is smaller. This is for Java runtime behavior only. When no datalink is set on a Push Button, Stretch to Fit is disabled.	Push Button
System Menu	Whether the window displays a System Menu.	Window
Tab	Whether the object is accessible by pressing the Tab key.	most objects
Text	Text to be displayed on the object.	most objects
Title Bar	Whether the window displays a title bar.	Window
Vertical Scroll Bar	If the object displays a vertical scroll bar (SHOW_ALWAYS or SHOW_NEVER). Scrollbars are not supported in Java.	Window
Visible	If the object is visible.	most objects
Width	The width of the object (in the Coordinate Type unit of measure).	most objects

Formatting a Window Object

Edit fields, List Boxes, and Multicolumn List Boxes in general inherit format specifications from their corresponding field entities in the repository. This topic describes how to use the format dialog to change format specifications for the following:

- Specifying Character Format
- Specifying Date and Time Format
- Specifying Numeric Format

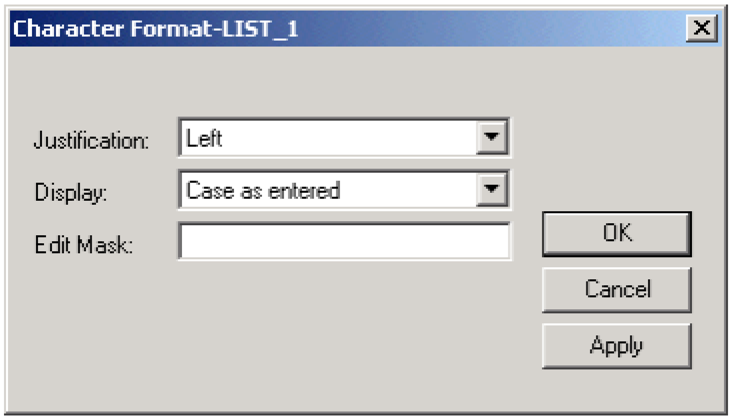
Unless you link a field for the Edit Fields, List Box, or MCLB, the Format option is not activated. To display the format dialog for an Edit Field, List Box, or MCLB:

1. Right-click the Edit Field that is linked to a field.
2. Select **Format** from the pop-up menu to display the format dialog box. The following sections describe how to use the format dialog box.

Specifying Character Format

If the field format property of the linked field is Character, Graphic Character (DBCS), Mixed Character, or VarChar (variable character), the Character Format dialog is displayed as shown in Character Format dialog example. The values are listed in Character Format dialog fields.

Character Format dialog example



Character Format dialog fields

Format property	Description

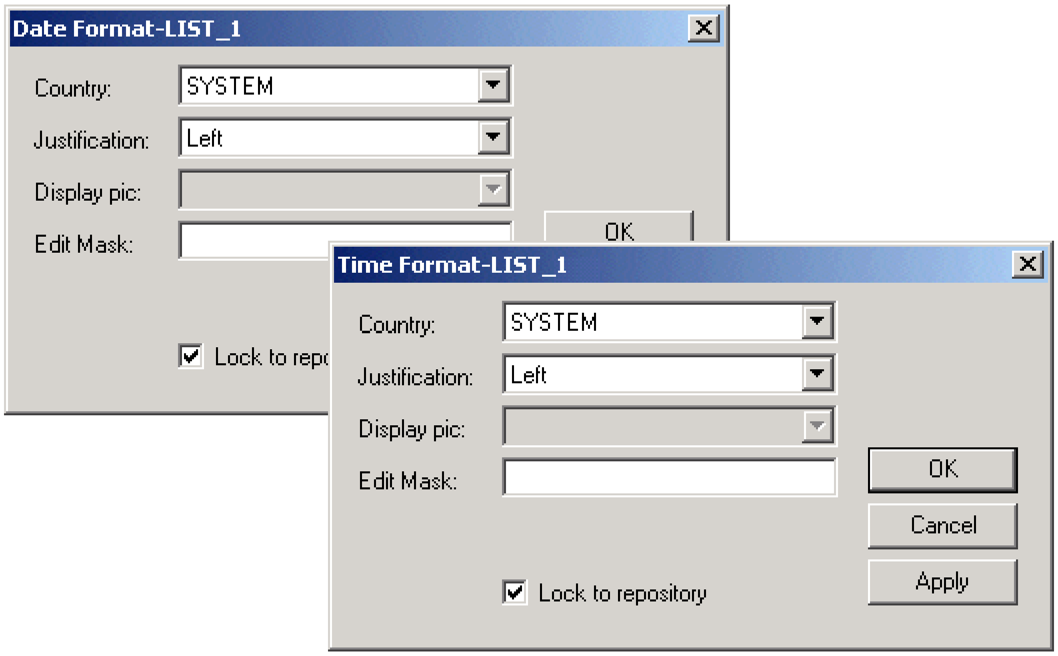
Justification	The justification of the data displayed in the linked field. The options are: • Center • Left • Right The options are mutually exclusive; the default is **Left**. Justification for List Boxes is not supported in C but is supported for Java.
Display	The capitalization of the data displayed in the linked field. The options are: • All lower case • All upper case • Cap first letter, every word • Cap first letter, first word • Case as entered The options are mutually exclusive; the default is **Case as entered**.
Edit Mask	The edit mask to use on the data displayed in the linked field (Java only). Use edit masks to format the data in a field or to restrict the input to certain characters in certain positions.

Click **Apply** to save the changes and keep the dialog open. Click **OK** to save the changes and close the window. Click **Cancel** to close the window without saving the changes.

Specifying Date and Time Format

If the field format property of the linked field is Date or Time, the Date Format or Time Format dialog box is displayed as shown in Date and Time Format dialogs. The dialog boxes are identical except for the title bars. The values are listed in Date and Time Format window fields.

Date and Time Format dialogs



Date and Time Format window fields

Format property	Description
Country	The country setting to govern the default date and time formats of Edit Fields. Choose from a list of countries whose formats are supported by AppBuilder. The default, SYSTEM, enables multiple language support (MLS) for these formats by specifying that the current workstation has control over them.

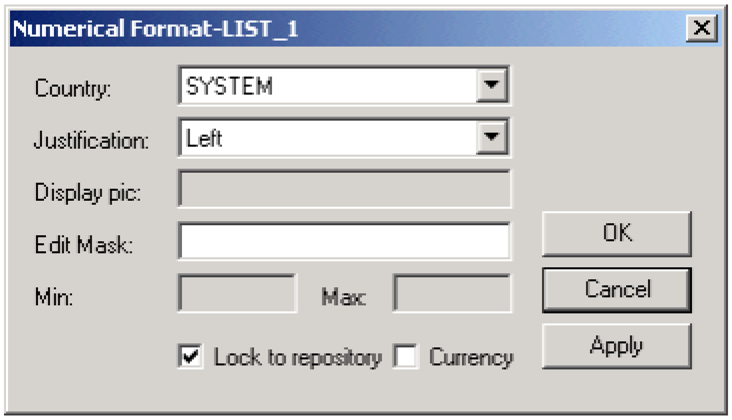
Justification	Specify the justification of the data displayed in the linked field. The options are: • Center • Left • Right The options are mutually exclusive; the default is **Left**. Justification for List Boxes is not supported in C but is supported for Java.
Display Pic	Specifies the format in which data in the field are displayed; it corresponds to the input field **Field Picture – Display** in the Properties window for the field in the Hierarchy diagram. If you have specified a picture for the field, it is displayed in the edit area of the Combo Box when you open the Date or Time Format dialog. The repository picture is not altered; select the **Refresh** Push Button to restore the picture to the display. Customized date or time formats can use only the symbols shown in the standard formats; Julian date formats are supported. Select one of the standard date or time formats in the drop-down list or edit the Combo Box by typing in the edit area. See the *Rules Language Reference Guide* for standard date and time functions.
Edit Mask	The edit mask to use on the data displayed in the linked field (Java only). Use edit masks to format the data in a field when the field does have focus or to restrict the input to certain characters in certain positions.
Lock to repository	Keeps data in sync with the repository values at the time Window Painter is opened. If selected, date and time formats cannot be changed, and the *Refresh* Push Button is grayed out. This property works only while Window Painter is running.

Click **Apply** to save the changes and keep the dialog open, **OK** to save the changes and close the window, or **Cancel** to close the window without saving the changes.

Specifying Numeric Format

If the field format property of the linked field is numeric (Decimal, Small Integer or Integer), the Numerical Format dialog box is displayed as shown in Numerical Format dialog box. The values are listed in Numerical Format dialog values

Numerical Format dialog box


Numerical Format-LIST_1

Country: SYSTEM
Justification: Left
Display pic:
Edit Mask:
Min: Max:
☑ Lock to repository ☐ Currency

Numerical Format dialog values

Format property	Description
Country	The country setting to govern the default date and time formats of Edit Fields. Choose from a list of countries. The default is **SYSTEM**. The drop-down lists countries whose formats are supported by AppBuilder.
Justification	The justification of the data displayed in the linked field. The options are: • Center • Left • Right The options are mutually exclusive; the default is **Left**. Justification for List Boxes is not supported in C but is supported for Java.

Display Pic	The format for displaying data in the field; it corresponds to the input field **Field Picture – Display** in the Properties window for the field in the Hierarchy diagram. If you have specified a picture for the field, it is displayed in the edit area of the Combo Box when you open the Numeric Format dialog. You can edit the picture by typing over it; the repository picture is not altered. You can restore the repository picture to the display by selecting the **Refresh** button. Refer to **Appendix B** for descriptions of display picture characters.
Edit Mask	The edit mask to use on the data displayed in the linked field. (Java only.) Use edit masks to format the data in a field when the field does have focus or to restrict the input to certain characters in certain positions.
Min and Max range values	Sets the minimum and maximum values for the field.
Lock to repository	Keeps data in sync with the repository values at the time Window Painter is opened. If selected, date and time formats cannot be changed, and the **Refresh** button is grayed out. This property works only while Window Painter is running.
Currency	If selected, the field is formatted with the currency symbol of the specified country. If you do not select Currency, but enter the $currency symbol character in the display picture specified for the field, the field is treated as if you selected Currency. Whether or not you select Currency, the position of a $character in a display picture overrides the symbol's position in the default currency format of the specified country. That is, if the specified country is Sweden, the picture $99,999.99 displays the input value of 12345.60 as kr12.345,60. By contrast, if Currency is selected and the picture does not have a $sign, the same input is displayed as 12.345,60 kr. By default, Currency is not selected.

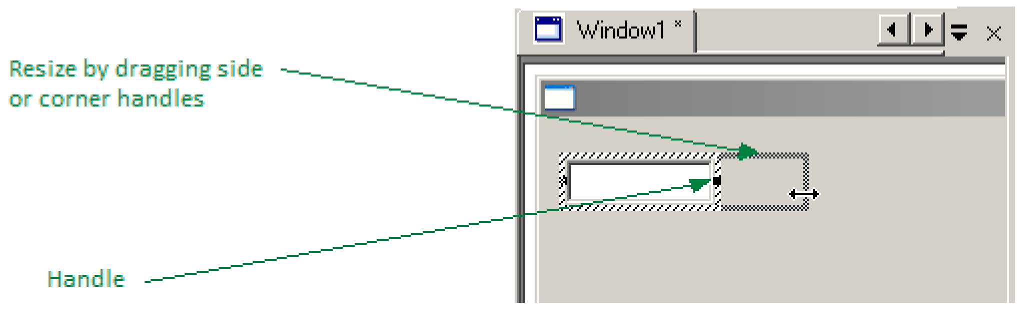
Click **Apply** to save the changes and keep the dialog open, **OK** to save the changes and close the window, or **Cancel** to close the window without saving the changes.

For Country property, the default, SYSTEM, enables multiple language support (MLS) for these formats, by specifying that control over them resides with the current workstation. For example, input to a date field is displayed in German format if the country setting of the end user's workstation is Germany. Any formats the end user customizes are recognized. Other settings for the Country combo box disable the MLS feature: date and time field input are displayed in a predefined format for the specified country, no matter what format the end user specifies. To override the window setting for a given Edit Field, specify a different value for this property. This property is overridden if you specify a format other than the default in the Display Pic combo box in the Date or Time Format dialog. You can specify multiple date or time formats in the same window.

Resizing a Window Object

You can resize most objects that can be placed in the window by selecting the object and dragging a handle to a new location. When the cursor is over one of the handles of the selected object, the pointer changes to arrows that indicate the direction the handle can be dragged. To change both the height and width of the field, select the corner handles.

Resizing an object

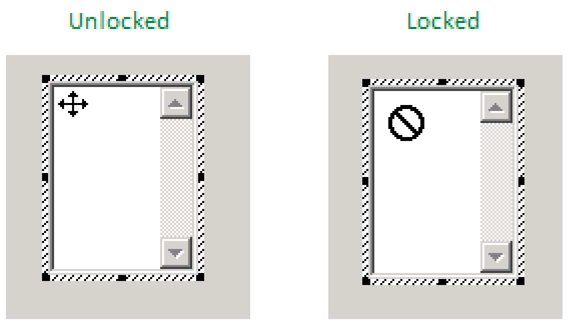


For an Edit Field object, you can adjust the width, but you must set the Resize property to True before you can change the height. See [Resizing an Edit Field]() for more information.

Locking a Window Object

You can "lock" a window object in place to avoid accidentally moving, resizing, or deleting it. When a window object is locked, the cursor changes showing that you cannot do anything with that object. The figure below shows an example of an unlocked and locked List Box.

Cursor change for locked object



To lock an object in a window, complete the following steps:

1. Select the object to lock.
2. Right-click the object and select **Lock** from the pop-up menu. The object is locked in place and cannot be moved or resized. You can also select **Layout > Lock** from the Construction Workbench menu or press **Ctrl+K** to lock an object.

To unlock an object, repeat the above procedure and uncheck **Lock**.

⚠ You can still change the tabbing order of locked objects. See Using the Tabbing Editor for instructions to change tab orders.

Moving Objects in a Window

To move an object to another location, click and hold the object in the window with the left mouse button. While holding down the left mouse button, move the object to a new location and release the left mouse button. You can also use the up, down, right, or left keys to move the selected object gradually.

To move and organize more than one object at a time, see the following topics:

- Aligning Objects in a Window
- Distributing Objects in a Window

Aligning Objects in a Window

To align objects in a window, complete the following steps:

1. Select two or more objects to align.

✓ To select multiple objects, press **Ctrl** while clicking one object at a time. You can also select multiple objects by clicking the mouse on the window and dragging it to draw a rectangle. All objects within the rectangle are selected. To select all the objects in the window, press **Ctrl + A**.

2. Select one of the following options to align objects:

Aligning Objects

To align to...	Select...
Top edge	Click the **Align Top** button in the Window Layout toolbar (see Window – Layout toolbar). You can also align objects by selecting **Layout > Align > Top** from the Construction Workbench or by pressing **Ctrl+Up Arrow**.
Bottom edge	Click the **Align Bottom** button in the Window Layout toolbar (see Window – Layout toolbar). You can also align objects by selecting **Layout > Align > Bottom** from the Construction Workbench or by pressing **Ctrl+Down Arrow**.
Left edge	Click the **Align Left** button in the Window Layout toolbar (see Window – Layout toolbar). You can also align objects by selecting **Layout > Align > Left** from the Construction Workbench or by pressing **Ctrl+Left Arrow** .

Right edge	Click the **Align Right** button in the Window Layout toolbar (see Window – Layout toolbar). You can also align objects by selecting **Layout > Align > Right** from the Construction Workbench or by pressing **Ctrl+Right Arrow**.

Distributing Objects in a Window

To evenly distribute objects in a window:

1. Select two or more objects to distribute.

> To select multiple objects, press **Ctrl** while clicking one object at a time. You can also select multiple objects by clicking the mouse on the window and dragging it to draw a rectangle. All objects within the rectangle are selected. To select all the objects in the window, press **Ctrl + A**.

2. Select one of the following options to distribute objects:

Distributing Objects

To distribute...	Select...
Vertically	Click the **Distribute Vertically** button in the Window Layout toolbar (Window – Layout toolbar). You can also distribute objects by selecting **Layout > Distribute > Vertically** from the Construction Workbench or by pressing **Alt+Up Arrow**.
Horizontally	Click the **Distribute Horizontally** button in the Window Layout toolbar (Window – Layout toolbar). You can also distribute objects by selecting **Layout > Distribute > Horizontally** from the Construction Workbench or by pressing **Alt+Right Arrow**.

Copying a Window Object

To copy a window object, do one of the following:

- Right-click and select **Copy** from the pop-up menu
- Select the object and select **Edit > Copy**
- Press **Ctrl+C** on the keyboard
- Choose Edit > Copy Menu
- Choose Edit > Copy Case

To paste the object into the same window or another window, do one of the following:

- Right-click in the window and select **Paste** from the pop-up menu
- With the destination window in focus, select **Edit > Paste**
- With the destination window in focus, press **Ctrl+V** on the keyboard

Removing a Window Object

To remove (or delete) a window object from a window, select the object and do one of the following:

- Right-click and select **Clear** from the pop-up menu.
- Select the object and select **Edit > Clear**
- Select the object and press **Delete** on the keyboard

Managing Tab Order of Window Objects

When creating a window, you can define the sequence in which the end-user can tab from one window control to another. You can also organize your controls into specific groups. The Tab Order and Group specifications are related in that they are both part of screen navigation, but they operate independently of each other.

The following topics illustrate how to manipulate the tab order of your window objects:

- Using the Tabbing Editor
- Understanding Tab Groups
- Using the Tab Order Option

Using the Tabbing Editor

The Tabbing Editor window displays the list of window controls for a specific window. For each control, the tabbing editor displays whether the control is designated as a tab stop in the tabbing sequence and whether that control starts a group. You can set both tab stop order and group specification from this window.

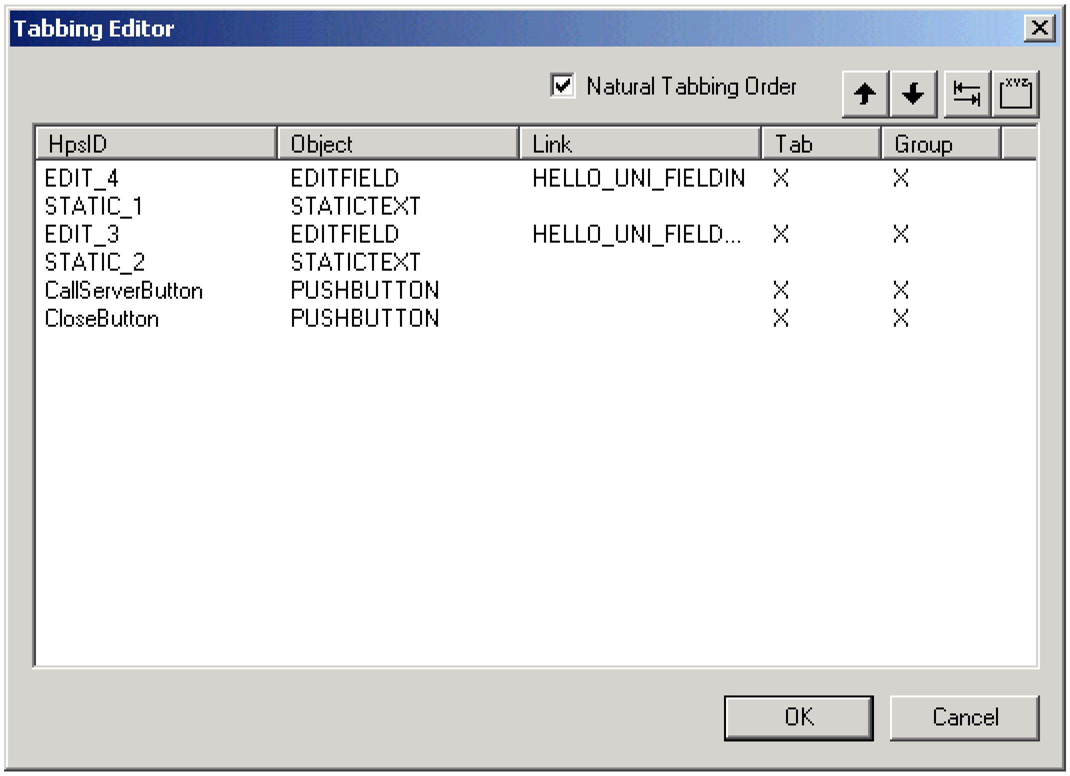
Objects are drawn in the Window Painter according to their tab orders. In case two objects overlap, the object with the lower tab order number is drawn earlier, causing the first object to be hidden underneath the second object. A good example of this is a Hotspot drawn on top of a Bitmap object. Changing the tab orders so that the Bitmap is drawn after the Hotspot makes the Bitmap object visible again.

To use the Tabbing Editor:

1. Open the Window in the Window Painter.

2. Click the Window Painter tab to give it focus and select **Tools > Tabbing Editor** from the Construction Workbench menu or press **Ctrl+T**. The Tabbing Editor window is displayed, showing all the controls for that window.

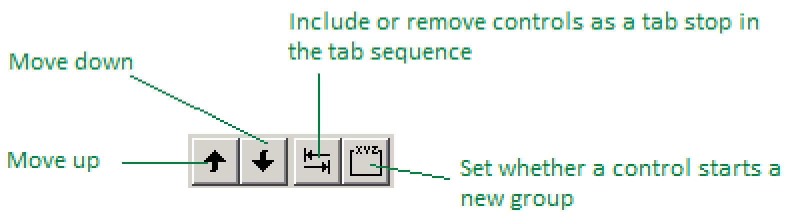
Tabbing Editor window



HpsID	Object	Link	Tab	Group
EDIT_4	EDITFIELD	HELLO_UNI_FIELDIN	X	X
STATIC_1	STATICTEXT			
EDIT_3	EDITFIELD	HELLO_UNI_FIELD...	X	X
STATIC_2	STATICTEXT			
CallServerButton	PUSHBUTTON		X	X
CloseButton	PUSHBUTTON		X	X

3. From the Tabbing Editor toolbar, shown below, you can include or not include a control as a tab stop in the tab sequence, move the item up or down in the tab list, or specify that a control starts a group.

Tabbing Editor toolbar


Include or remove controls as a tab stop in the tab sequence
Move down
Move up
Set whether a control starts a new group

4. After you finish setting the tab order, click **OK**.

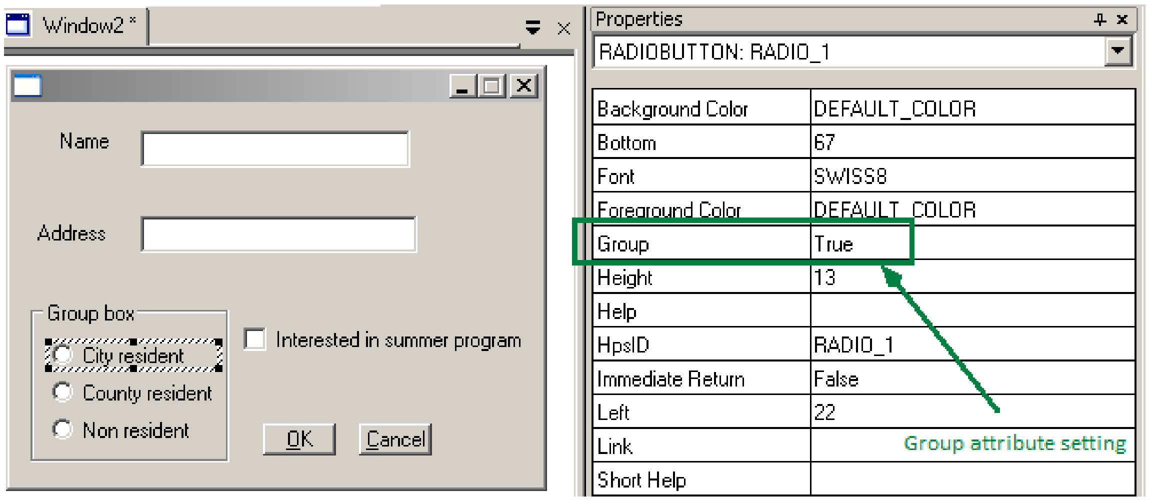
Natural Tab Order

When you click the **Natural Tab Order** check box, the tab order is set based on each control's position on the window (not the order in which the controls were added to the window), ordering the controls by their top to bottom, left to right positions. When setting up tab groups, you will typically not use natural tab order, unless you position the controls for the group in the natural – left to right, top to bottom – position on the window.

Understanding Tab Groups

A Group is a collection of a logical set of objects. When window controls are grouped, the user can use the arrow keys to move from one item in the group to another without having to use the Tab key. A good time to use Grouping is for a series of Radio Buttons. You can set which controls start a Group through the Properties window for the control or through the Tabbing Editor. To start a Group, set the Group property to **True**. To end a group, specify the Group attribute for the next control in the tab order (outside the group) as **True**. In other words, when you start a new group, the previous group is ended.

Example of tab order and grouping controls



RADIOBUTTON: RADIO_1

Background Color	DEFAULT_COLOR
Bottom	67
Font	SWISS8
Foreground Color	DEFAULT_COLOR
Group	True
Height	13
Help	
HpsID	RADIO_1
Immediate Return	False
Left	22
Link	
Short Help	

To specify the Radio Buttons in the Group Box as a group, you can set the Group attribute within the Property window for each control (see Example of tab order and grouping controls), or you can place an *x* in the Group column on the Tabbing Editor window (see Tabbing Editor window).

In the example shown in Example of tab order and grouping controls, the properties for the first button, *City Resident* , are shown. Group attributes for other buttons are explained below:

Settings for controls

Object	Group Attribute
City Resident	True
County Resident	False
Non-Resident	False
Interested in summer program	True
OK	True

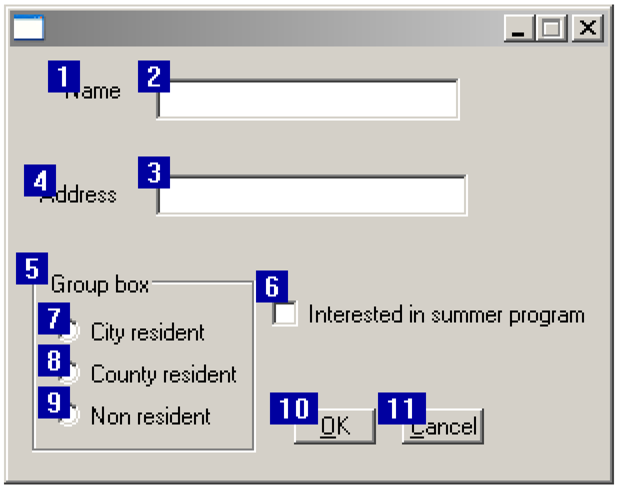
This table indicates that the Radio Button group ended with **Non-Resident** and a new group started with the Check Box **Interested in summer program**. The next item in the tab order after the Check Box is the **OK** Push Button.

Using the Tab Order Option

The Tab Order option, accessed from the Construction Workbench Layout menu, displays a number for each of the controls on your window from 1 to the last control in the window. This number signifies the order in which the controls are created at runtime.

You can specify the tab order for your controls through this option. Regardless of whether a control is designated as a tab stop, the control is numbered on this screen. Thus, to ensure the controls are designated as a tab stop, use the Tabbing Editor (see Using the Tabbing Editor).

Numbered controls displayed using Tab Order



To reorder your tab stops using the Tab Order option:

1. Open the Window in the Window Painter.
2. Right-click on the window and select **Tab Order** from the pop-up window. You can also click **Layout > Tab Order** or press **Ctrl+D**.
3. To rearrange the tab order, click the item that you want to be the first one in the tab order. Then click the second, etc. As you click the object, the number changes to the next in the sequence.

When you are setting up a group, as in Numbered controls displayed using Tab Order, you will **not** use the natural tab order. You will specify the tab stop order based on the items in the group. Notice that the natural tab order is not in place for all items.

Implementing Standard Menu Functions

Reserved system identifiers (HPSIDs) can be used with menu options to perform standard graphical user interface (GUI) actions. You can use the reserved system identifiers to implement the standard menu functions as summarized in Reserved system identifiers. System defined accelerator keys are predefined in Windows.

Reserved system identifiers

Menu function	Reserved ID	Accelerator key	Shortcut key	Description
Copy	HPS_MENU_COPY	Ctrl+Insert	Ctrl+C	Duplicates the text of the selected object(s) and places it on the clipboard. It does not delete the the original. Once on the clipboard, it may be placed elsewhere with the Paste option.
Cut	HPS_MENU_CUT	Shift+Delete	Ctrl+X	Removes the text of the selected object(s) and places it on the clipboard, a temporary storage area. Once on the clipboard, it may be placed elsewhere with the Paste option.
Delete	HPS_MENU_DELETE	Delete		Clears the text that is currently selected.
Help	HPS_MENU_HELP	F1	Delete	Opens the help pop-up window for the selected object.
Paste	HPS_MENU_PASTE	Shift+Insert	Ctrl+V	Places text from the clipboard to the current position of the cursor.
Print	HPS_MENU_PRINT	Print Screen (shift)	F1	Sends the contents of the current window to the printer, as specified in the print manager.
Close	HPS_MENU_CLOSE	Alt+F4	Alt+F4	Closes the active window.

None of the system-defined accelerator keys or shortcut keys should be designated for options other than their standard functions. Usually, the default system meaning supersedes any others.

The examples given for system-defined keys are true of the Windows environment. They are different in other operating environments.

Using the Chart Editor

A Chart object can be linked to a view or set of views by linking the Chart to a field in a multiple-occurring view. In the *Properties* window of the Chart, select a view from the drop-down list. From the right-click menu, select *Chart Editor* . The Chart Editor window is displayed as shown below. The areas are summarized in Chart Editor functions.

Chart Editor window

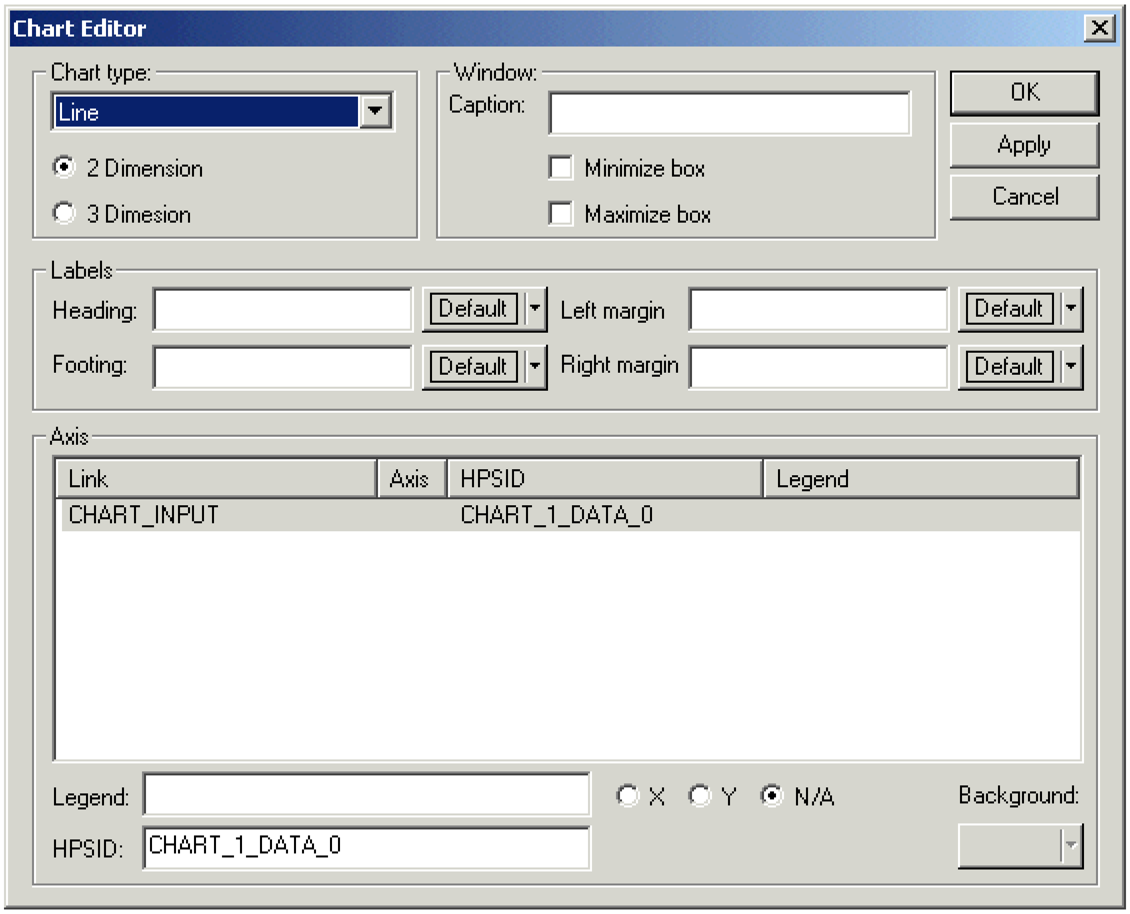


Chart Editor functions

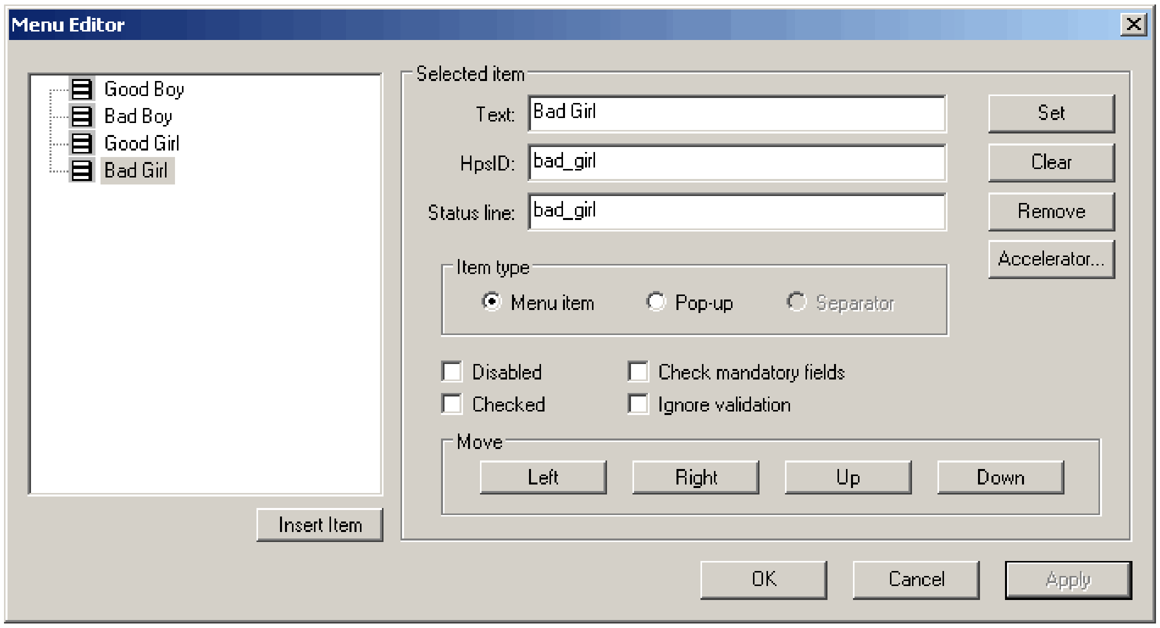
Area	Description
Chart type	The type of chart to display the data.
Window Caption	The caption to display with the Chart
Minimize or Maximize Box	Selection to maximize or minimize the box or do neither.
Labels	Labels for the axes.
Heading and Footing	Text to display in the header or footer of the Chart.
Left and Right Margin	The size of the left or right margin or both.
Link	The view to link to.
Axis	The axis on which to display the data.
HPSID	System identifier.
Legend	A label for the Chart's legend and whether to display the legend.
Background	The background color for the Chart.

Using the Menu Editor

Right-click the window and select **Menu Editor** from the pop-up menu. The Menu Editor window is displayed. You can also display the Menu Editor window by selecting **Tools > Menu Editor** from the Construction Workbench menu or by pressing **Ctrl+U**. This topic includes:

- Reserved Menu Names
- Accelerator Button

Menu Editor dialog box



Text: Bad Girl
HpsID: bad_girl
Status line: bad_girl
Item type: [✓ Menu item] Pop-up Separator

1. Click **Insert Item**.
2. Use this table to enter data in each field on the window:

Menu Editor window options

Option	Description
Selected Item	
Text	The menu option text
HPSID	The system identifier associated with this object. This is a required field for all Java root menu objects.
Status Line	The text to display in the status line at the bottom of the window when the cursor is over this item
Selected Item Push Buttons	
Set	Submits the information that you provided in the above three fields, setting it in the hierarchy at the left of the Menu Editor.
Clear	Clears the text in the Text, HPSID, and Status Line fields.
Remove	Removes the selected item from the hierarchy at the left of the Menu Editor
Accelerator	Displays the Accelerator window so you can assign "hot keys" for this menu item (see Adding an Accelerator). In Java, the Accelerator button is only available for creating a menu item.
Item type	
Menu Item Pop-up Separator	Defines the type of item you are creating.
Disabled	Disables this menu by default.
Checked	Denotes that this menu is checked by default
Check mandatory fields	Denotes that mandatory fields are checked by default

Ignore validation	Denotes that the item's validation is ignored by default
Visible	Controls when a menu is visible or hidden. Use for creating window panels in different languages. To see this check box, you must have MLUI installed and have a Current Language defined in Workbench Options. See *Multi-Language User Interface Guide* .
Move	Moves items in the hierarchy left or right and up or down.

Reserved Menu Names

You can use Reserved Menu Names for standard menu functions. See Implementing Standard Menu Functions.

Accelerator Button

For information on creating an Accelerator, see Adding an Accelerator.

> Java supports Accelerators, but only for menu items. The use of NLS characters in the mnemonic keystroke sequence is only supported in the C client runtime. Java does not support the use of NLS characters for mnemonics.

Using the Multicolumn List Box Editor

The Multicolumn List Box (MCLB) Editor lets you add and format columns and column headings. An MCLB must be linked to an occurring view in the hierarchy.

To insert a multiple-occurring view, display the Properties window for that view. Expand the **Relationship** set of options. In the Occurs Times field, type the number of occurrences (greater than 1) for this view, and click **OK**. The view is now a multiple-occurring view.

This topic includes:

- Opening the MCLB Editor
- Adding and Modifying Columns with the MLCB Editor
- Adjusting Column Widths in MCLB
- Modifying Column Justification in MCLB

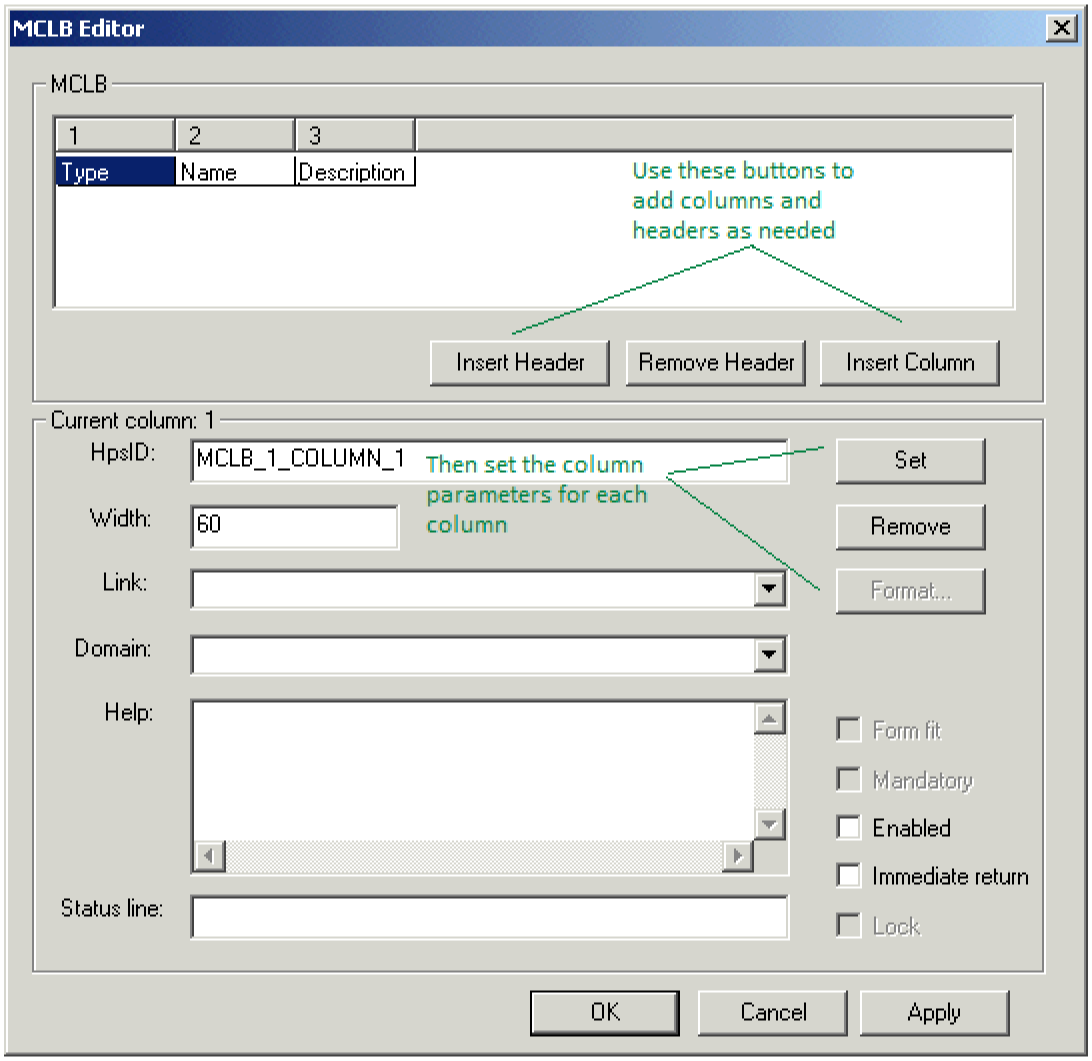
Refer to Multicolumn List Box (MCLB) Object for information specific to 3270 windows.

Opening the MCLB Editor

To open the MCLB Editor:

1. Select the MCLB object in the window.
2. Select *MCLB Editor* from the *Tools* menu or the right-click pop-up menu, or use the shortcut *Ctrl+Shift+M* . The MCLB Editor window displays. A sample is shown in Sample MCLB Editor window.

Sample MCLB Editor window



MCLB Editor fields and options

Interface	Description
MCLB	The part of the window that allows you to see columns and headers as you add them.
Current Column	The properties of the column selected in the MCLB above.
HPSID, Width, Link, Help, Status Line	See Common properties for Window Painter objects for a description of common properties.
Domain	A property that indicates if the object is linked to a Set object.
Form fit, Mandatory, Enabled, Immediate return, Lock.	Properties available for a column. See Common properties for Window Painter objects for a description of common properties.

When you initially open the MCLB Editor for an MCLB that is not linked to repository fields, no columns or headings are displayed in the MCLB Editor.

Adding and Modifying Columns with the MLCB Editor

Each column in the MCLB Editor contains a numbered column selector at the top. (See Sample MCLB Editor window) The column selector refers to the column's placement in the object. It is not displayed in the MCLB object in the window, nor is it visible to the end user at runtime. Select a column by clicking on a column selector, type a value in the "Width" edit field, then click "Set" button, to change the width of one column. You can select only one column at a time.

To build a Multicolumn List Box, add columns, insert headers, and specify features for that column in the bottom half of the window.

The column heading is displayed in the cell below the column selector. The heading is based on the field label settings. By default, any long screen literal you assigned to a field in the repository is automatically displayed as the heading of the column with which it is linked. The heading is truncated if it is longer than the column is wide.

In Window Painter, the headers are in the first row of the MCLB, taking up one of the rows. In 3270 Window Painter, the headers are in one row above the MCLB and do not take up a row.

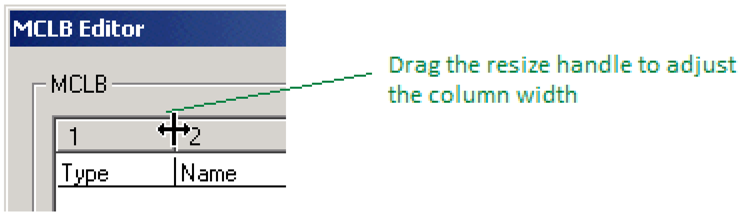
Your MCLB must have at least one column. If the MCLB has zero columns specified, the preview in Window Painter or Window Flow Diagram displays the MCLB as a line.

If you update the parameters section, select **Set** to enable the **Apply** button.

Adjusting Column Widths in MCLB

To adjust the width of a column, click the line at the right of the column. When the pointer changes to a resize handle, drag it to the desired width of the column heading.

Column resize handle



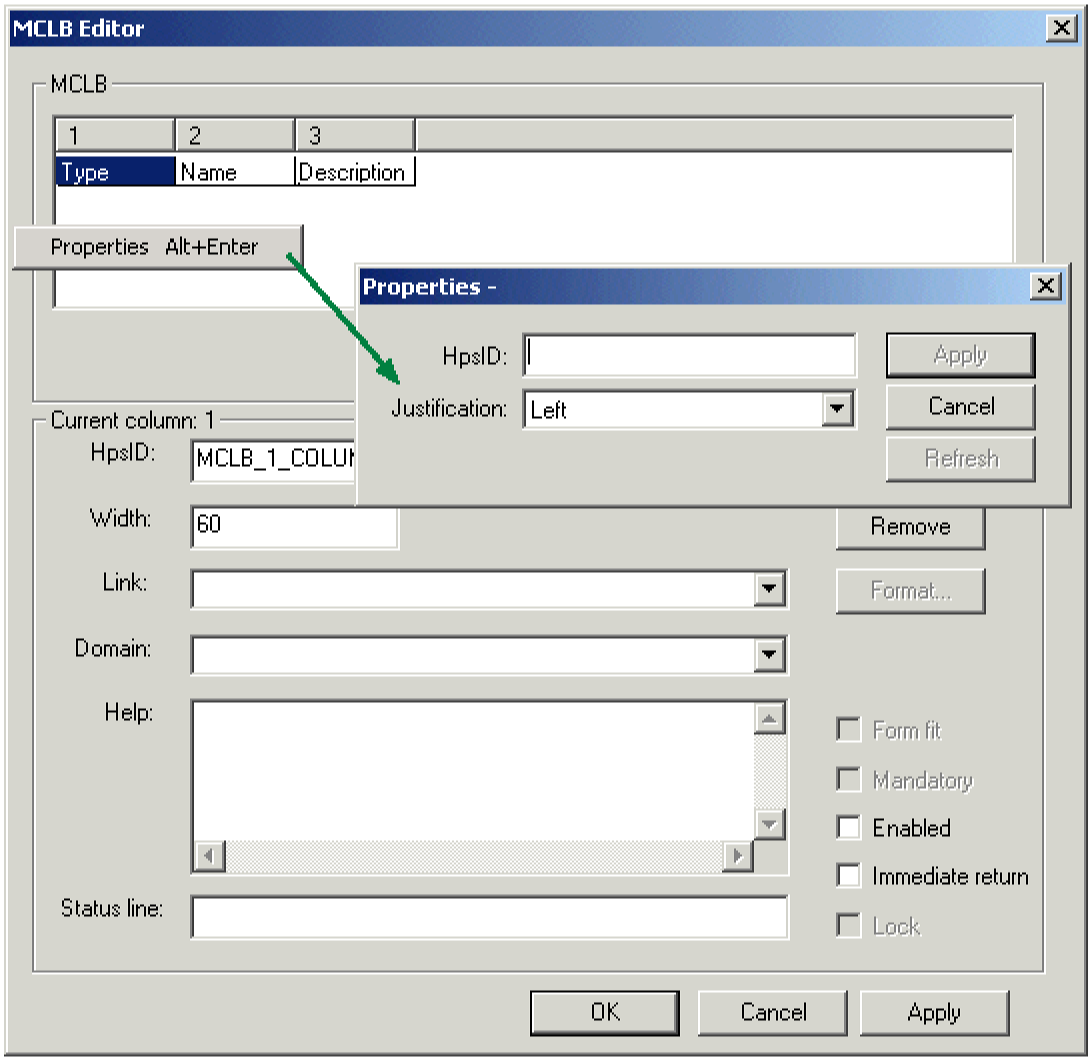
Save changes to the MCLB Editor window by clicking **Apply**.

To close the MCLB Editor, click **OK** or click the **X** in the upper right corner of the MCLB Editor window.

Modifying Column Justification in MCLB

Change the header justification (left, center, right) by selecting the header and selecting **Properties** from the right-click menu, as shown in [Modifying column justification](#).

Modifying column justification



Save changes to the MCLB Editor window by clicking **Apply**.

To close the MCLB Editor, click **OK**, or click the **X** in the upper right corner of the MCLB Editor window.

Understanding Objects in a Window

The following objects can be inserted into a window:

- Bitmap
- Chart (not supported in Java)
- Check Box
- Combo Box
- Edit Field
- Ellipse
- Group Box
- Hot Spot (not supported in Java)
- List Box
- Menu
- Multicolumn List Box
- Multiline Edit Field
- Push Button
- Radio Button
- Rectangle
- Static Text
- Tab Controls

Bitmap

You can place a picture image in the window using the Bitmap Object and the Bitmap Viewer.

- To add an empty Bitmap Object to a window, see Inserting an Object in a Window.
- To modify the object properties, see Viewing and Modifying Window Object Properties.
- To import a graphic image and associate it with a Window, see Bitmap Viewer in Construction Tools.

You can use the Bitmap Viewer tool or the HPS_SET_BITMAP_FILE system component to link runtime Bitmap objects to repository Bitmap objects (See *System Components Reference Guide*). The following entry in the Hps.ini file indicates the location of all bitmap and icon files to be used at runtime or during simulation with the Widow Flow Diagram:

```
BITMAP_DIR=C:\AppBuilder\NT\RT\BMP
```

You can specify more than one directory for bitmap files to be stored by adding the directory names using semicolons to separate each directory as follows:

```
BITMAP_DIR=M:\APPNAME\HPS\NT\RT\BMP;C:\TEMP
```

The image that you use for a Bitmap object can be platform-dependent or platform-independent.

⚠ You must use the correct bitmap image for the targeted execution environment. For example, a Windows formatted bitmap may not display properly on a non-Windows workstation.

Chart

Use Charts to graphically display data. AppBuilder can create many types of charts including bar charts, pie charts, and area charts, with both two-dimensional (2-D) and three-dimensional (3-D) effects. For instructions to add a Chart object to a window, see Inserting an Object in a Window. For instructions to modify the object properties, see Viewing and Modifying Window Object Properties.

When you insert a Chart object in a window, complete the following steps:

1. Link the Chart to a field of a multiple-occurring view. (In the *Properties* window of the chart, select the view from the Link drop-down list. If the list is empty, make sure that you have a multiple-occurring view in the hierarchy under the window object. For information about multiple-occurring views, see Using the Multicolumn List Box Editor.)
2. Select the type of chart. Refer to Chart types for a summary of chart types.
3. Right-click the Chart object and select **Chart Editor** (or press **Ctrl+Shift+M**) to edit the Chart. See Using the Chart Editor.

Chart types

Chart Type	Description
LINECHART2D LINECHART3D	Show trends in data by displaying data as points on a graph connected by straight lines, both 2 and 3 dimensional options.
BARCHART2D BARCHART3D	Show comparisons of data using a horizontal bar for each data item value, both 2 and 3 dimensional options.
PIECHART2D PIECHART3D	Show proportional size of each data item that makes up a whole circular image (pie), both 2 and 3 dimensional options.
STACKEDBARCHART2D STACKEDBARCHART3D	Show the relationship of individual parts of data by displaying sets of data as stacks placed adjacent to each other (as in the bar chart), 2 and 3 dimensional options.
COLUMNCHART2D, COLUMNCHART3D	Show a comparison of data using a vertical column for each data item value, both 2 and 3 dimensional options.
SMOOTHLINECHART2D SMOOTHLINECHART3D	Show a comparison of data in a curved line, both 2 and 3 dimensional options.
SCATTERCHART2D	Shows the scattered data on a graph without lines connecting the data items, 2 dimensional only.
AREACHART2D AREACHART3D	Consist of one or more lines drawn on an X-Y grid, with the area between the line and the X axis filled in, both 2 and 3 dimensional options.
BARLINECHART2D BARLINECHART3D	A combination of bar and line charts, both 2 and 3 dimensional options.
PERBARCHART3D	A perbar chart clusters data on the Z axis, 3D only.

In Java, the Chart object is not supported.

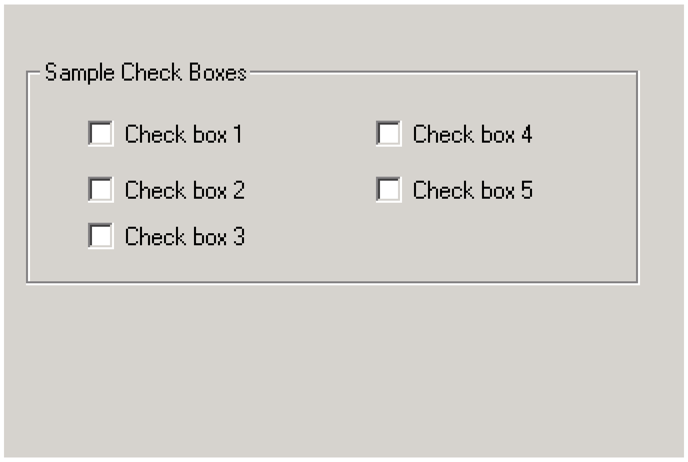
Check Box

A Check Box is a square box displayed next to a text label. Users can select or deselect the box. The system provides visual feedback as to the status of a Check Box by placing (or removing) a check mark in the box. Check boxes are linked to **boolean** , **char** , or **varchar** fields to specify yes/no or on/off conditions. A **char** or **varchar** edit field linked to the checkbox contains an X when the checkbox is checked. When there is no check, the edit field is blank. A **boolean** edit field linked to a checkbox contains TRUE when the box is checked. When the checkbox is unchecked, the **boolean** field contains FALSE.

Only Java and Servlet programs support *boolean* fields being linked to checkboxes.

For instructions to add a Check Box object to a window, see Inserting an Object in a Window. For instructions to modify the object properties, see Viewing and Modifying Window Object Properties.

Sample Check Boxes



When you insert a Check Box object in a window, complete the following steps:

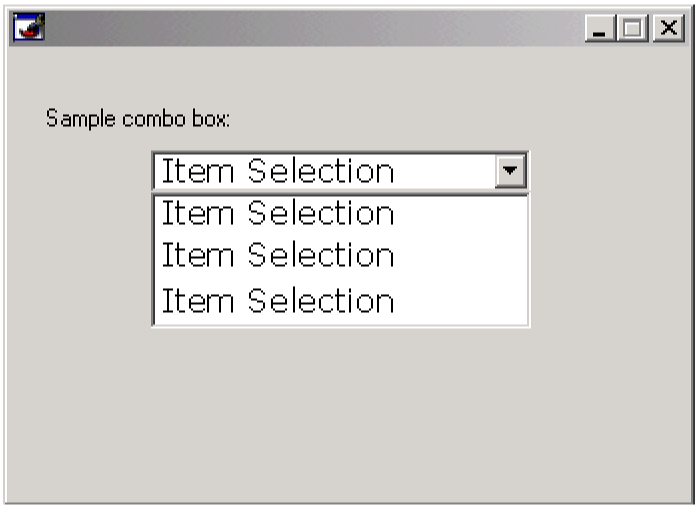
1. Edit the Text property, changing the text that is displayed next to the Check Box.
2. You can link the Check Box to a field using the Link property in the Properties window of the Check Box. You must have at least one Field object under a View object under the Window object in the Hierarchy window. Double-click a field in the Window Painter tab. The Properties window for that field is displayed. Select the *Link* property, and from the drop-down list of possible Field objects in the repository, select a field.

A mnemonic key can be defined by prefacing any character in a Text attribute with an ampersand, "&". Using NLS characters in the mnemonic keystroke sequence is only supported in the C client runtime. Java does not support the use of NLS characters for mnemonics.

Combo Box

A Combo Box is a combination of an Edit Field and a Drop-down List Box. End users can change the text in the edit field by choosing an item from the list box. In the case of editable Combo Boxes, users can also type directly into the Edit Field. A Combo Box is linked to a repository field, which is linked to a set that contains the displayed values. For instructions to add a Combo Box object to a window, see Inserting an Object in a Window. For instructions to modify the object properties, see Viewing and Modifying Window Object Properties.

Sample Combo Box



You can link the Combo Box to a field using the Link property in the Properties window for the Combo Box.
After you insert a Combo Box object in a window, click the Combo Box. The Properties window for the Combo Box displays. Select the following:

- For the **Domain** property, select either a set attaching to the window or the field of a multiple-occurring view.
- For the **Link** property, select a field.
- For the **Style** property, select between DROPDOWN, DROPDOWNLIST and SIMPLE. Simple style is not supported for Java.

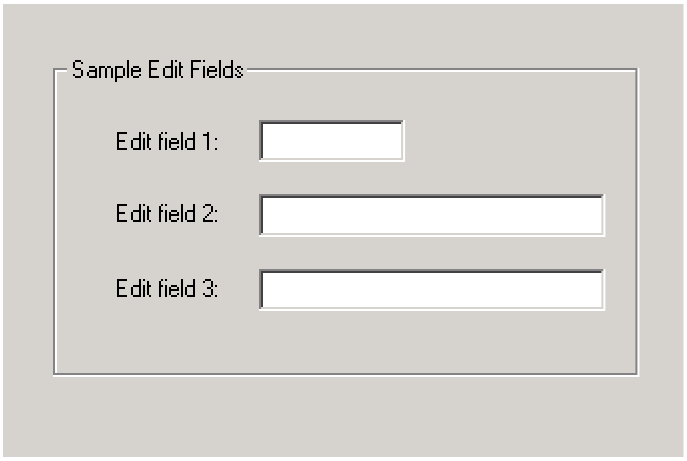
Edit Field

Use an Edit Field to define fields in which the end user can enter and modify a single line of text. The Edit Fields can be **Editable** (not protected) or not Editable. In an Editable field, you can populate fields by typing in data at runtime or by mapping data from the rule. Use a Multiline Edit Field to allow the end user to enter more than one line of text.

For the Windows execution environment, the text the user enters in a field with a default height is clipped. Set the Resize property in the Properties window of the Edit Field to **True** , which allows you to change the height. Refer to Resizing an Edit Field.

Edit Field objects create input and output areas in a window (as in Sample edit fields). In a AppBuilder application, you can populate an Edit Field by typing data at runtime or by mapping data from the rule. For instructions on adding an Edit Field object to a window, see Inserting an Object in a Window. For instructions on modifying the object properties, see Viewing and Modifying Window Object Properties. See also Field.

Sample edit fields



You can link a protected (not editable) Edit Field to a repository field and associate the domain to a Set object containing domained values.

For example, if you have a set with the values **1** - **12** assigned to the months of the year, you can link an Edit Field to a numeric value, and AppBuilder displays the corresponding month's name. Thus, the repository field linked to the object takes only one of the domained values from the set object associated with the field. For protected Edit Fields, the domained values are not displayed (as they are with MCLB columns).

When the domain property is set for an object, the Format menu is grayed out because the format is irrelevant when the object is linked to a set.

C runtime does not support domained Edit Fields.

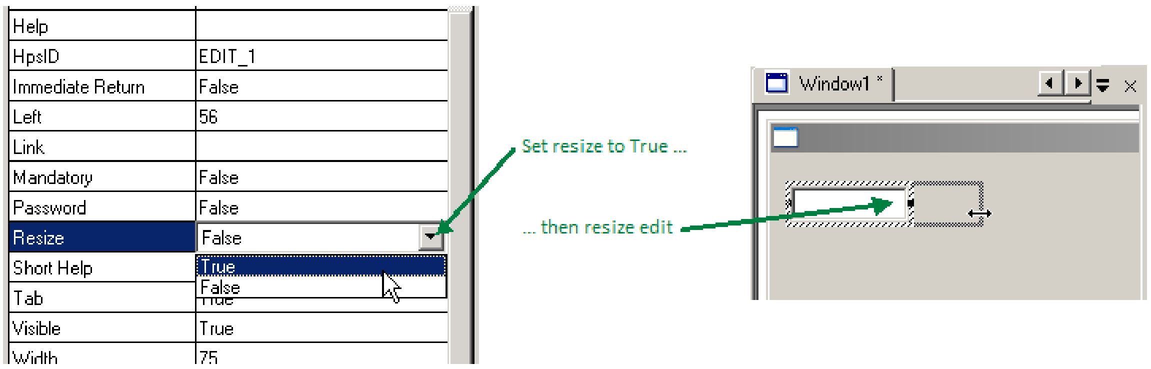
After you insert an Edit Field object in a window:

1. You can link the Edit Field to a field using the Link property in the Properties window for the Edit Field. You must have the following structure in the Hierarchy window: at least one Field object under a View object under the Window object. Display the Properties window for that Edit Field. Select the **Link** property in the Properties window, and from the drop-down list of possible Field objects in the repository, select a field.
2. For the **Domain** property, select either a set or the field of a multiple-occurring view.

Resizing an Edit Field

Unlike other objects that can be placed and resized in the window, the height of the Edit Field object is locked, depending on the font being used. By default, you can change the length, but not the height. To change the height of an Edit Field, open the Properties window for the Edit Field and set the Resize property to **True**. Only then can you resize the Edit Field by clicking and dragging the handles of the Edit Field, as shown in Resizing Edit Field. To lock the height of the field again, you can set the Resize property to **False**.

Resizing Edit Field



Help	
HpsID	EDIT_1
Immediate Return	False
Left	56
Link	
Mandatory	False
Password	False
Resize	False
Short Help	
Tab	True
Visible	True
Width	75

When **Recalculate Field Sizes** is active, **Resize** and **Form Fit** field properties settings affect how an Edit Field is recalculated. See Window Painter Options. The table below shows how the field size is recalculated according to the settings of **Resize** and **Form Fit** when the **Recalculate Field Sizes** option is checked:

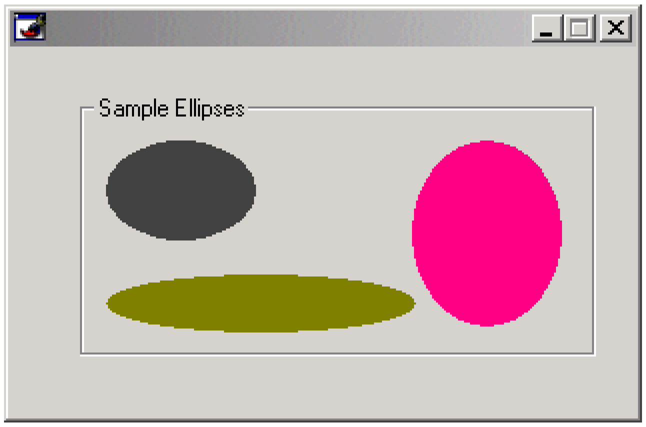
Field Size Recalculation

Resize Property	Form Fit Property	How the Edit Field size recalculation is affected:
False	False	Only Height is recalculated. Width can be manually updated using tools.
False	True	Both Height and Width are recalculated. Width can be manually updated using tools.
True	False	Field size is not recalculated. Both Height and Width can be manually updated using tools.
True	True	Only Width is recalculated. Height can be manually updated using tools.

Ellipse

An Ellipse, like a Rectangle, adds graphical interest or clarification to the window. Like a Rectangle, an Ellipse contains no link to a repository field. For instructions to add an Ellipse object to a window, see Inserting an Object in a Window. For instructions to modify the object properties, see Viewing and Modifying Window Object Properties.

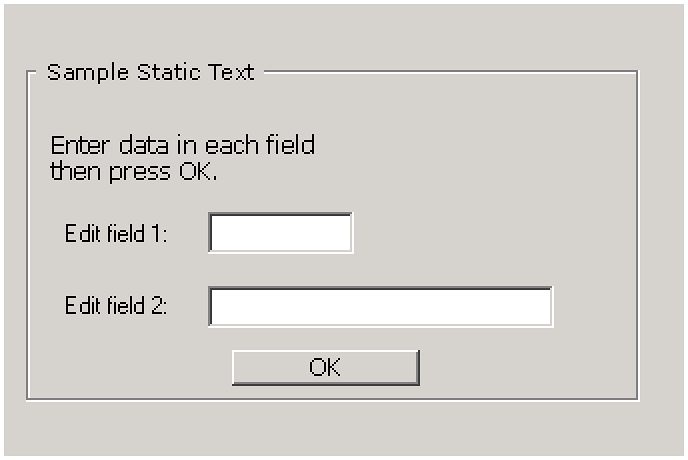
Sample ellipses



Group Box

A Group Box is a rectangular border that contains a text label. Other objects may be placed within a Group Box. This object is typically used to group similar items, such as Radio Buttons or Check Boxes, or different-type items that share a functional or organizational purpose. Group boxes are for display only; they do not affect the order of tabs on a panel and do not link to any repository object. For instructions to add a Group Box object to a window, see Inserting an Object in a Window. For instructions to modify the object properties, see Viewing and Modifying Window Object Properties.

Sample Group Box


Sample Static Text

Enter data in each field
then press OK.

Edit field 1:

Edit field 2:

Hot Spot

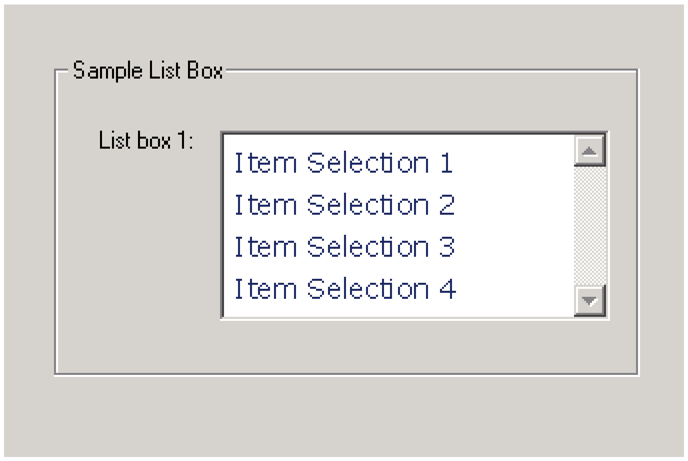
A Hot Spot is an area of the window that, when the user clicks it, sends an event to a rule. The Hot Spot is not linked to a repository object. It uses the predefined system view HPS_EVENT_VIEW to send the event. Unlike Push Buttons, Hot Spots are hidden from the end user at runtime. Use a Hot Spot in conjunction with a Bitmap to make clickable graphic buttons. For instructions to add a Hot Spot object to a window, see Inserting an Object in a Window. For instructions to modify the object properties, see Viewing and Modifying Window Object Properties.

The Hot Spot is not supported in Java.

List Box

A List Box is an object that contains a display-only list of one or more items. Users can scroll the list and can select one or more items. The data items in the list come from a field associated with a multiple occurring view in the hierarchy. Each item in the List Box is linked to a field within the application hierarchy. For instructions to add a List Box object to a window, see Inserting an Object in a Window. For instructions to modify the object properties, see Viewing and Modifying Window Object Properties.

Sample List Box



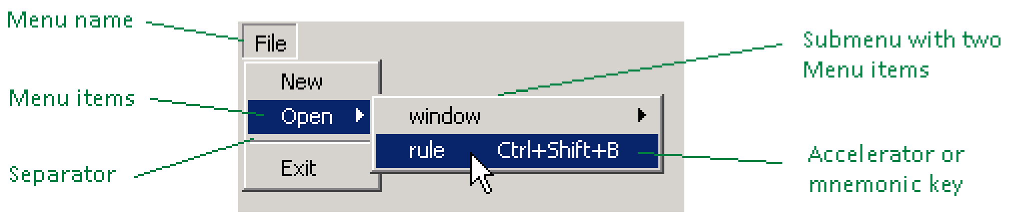
When you insert a List Box object in a window, complete the following steps to link the List Box to a View using the Link Property in the Properties window for the List Box.

1. Click the List Box object in the window. The Properties window is displayed.
2. Click the **Link** field property and select a View from the list in the drop-down List Box.

Menu

As opposed to the click-and-place creation of most window controls, menus are designed in the Menu Editor inside Window Painter. Menu items are displayed in a pull-down menu bar at the top of the window. Each menu item consists of both the text that is displayed in the pulled-down menu at runtime and an underlying unique system identifier (HPSID). The HPSID is used by the rule that converses the panel to call the desired menu item functionality. In the Menu Editor, you can also define accelerator and mnemonic key sequences for menu items. (See Using the Menu Editor.)

Sample Menu Bar



Menu choices do not change the focus state. When a menu item is selected, the focus remains on the object that previously had the focus.

Adding a Menu

To add a Menu Bar to a window, complete the following steps:

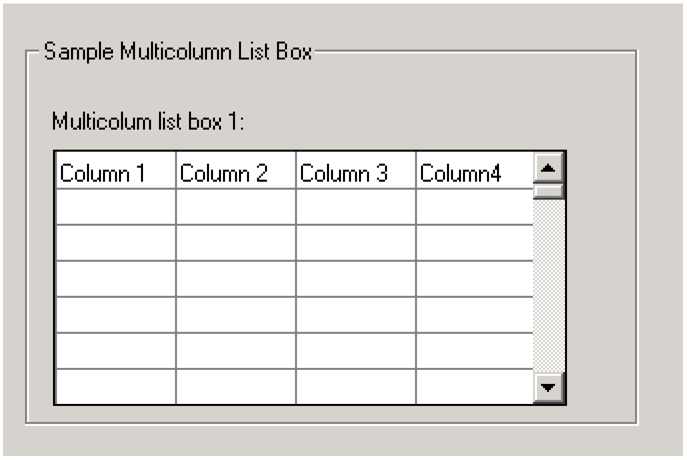
1. Right-click the window and select **Menu Editor** from the pop-up menu. The Menu Editor window is displayed.
   You can also display the Menu Editor window by selecting **Tools > Menu Editor** from the Construction Workbench menu or by pressing **Ctrl+U**.
2. Click **Insert Item**.
3. Enter the various properties for the new menu item and click **Apply**.

For instructions to modify the object properties, see Viewing and Modifying Window Object Properties.

Multicolumn List Box

A Multicolumn List Box (MCLB) is an object that contains a list of one or more items, displayed in a column-and-row format. The user can scroll the list and select one or more items. The data items in the list come from a field associated with a multiple occurring view in the hierarchy. See Sample Multicolumn List Box. For instructions to add a Multicolumn List Box object to a window, see Inserting an Object in a Window. For instructions to modify the object properties, see Viewing and Modifying Window Object Properties.

Sample Multicolumn List Box



You can link a protected (not editable) MCLB column to a repository field and associate the domain to a Set object that contains a set of domained values.

For example, if you have a set with the values *1 - 12* assigned to the months of the year, you can link an MCLB column to the list of numeric values, and AppBuilder displays the corresponding month's name. Thus, the repository field linked to the column takes only one of the domained values from the Set object associated with the column. For MCLB columns linked to a domain, the domained values are displayed at runtime as a drop-down list when the user clicks the column.

- C runtime does not support domained MCLB columns.
- Border color is not supported in C or Java for a Multicolumn List Box.
- When the domain property is set for an object, the Format menu is grayed out because the format is irrelevant when the object is linked to a set.

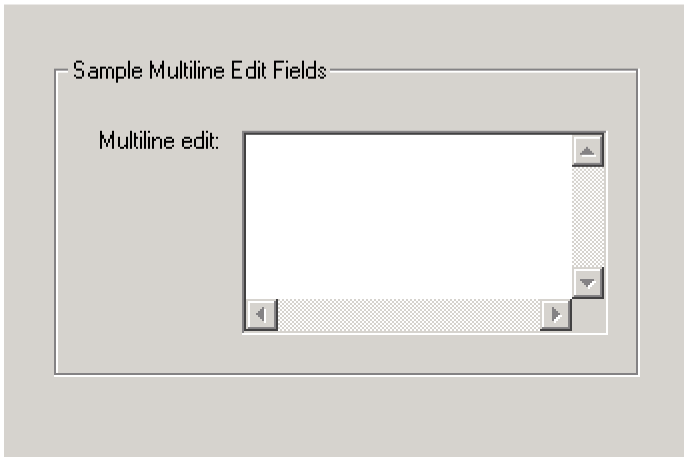
When you insert an MCLB object in a window, complete the followings steps:

1. You can link the MCLB object to a multiple occurring view using the Link property in the Properties window for the MCLB. Click the MCLB. The Properties window displays. Select the **Link** property, and from the drop-down list of possible multiple occurring Views, select a view.*
2. To add columns and link the columns to multiple occurring fields, use the MCLB Editor. Right-click a Multicolumn List Box in the Window Painter tab and select **MCLB Editor** from the pop-up menu. You can also access the MCLB Editor window by selecting **Tools > MCLB Editor** from the Construction Workbench menu. See Using the Multicolumn List Box Editor for more information.
3. In the Hierarchy window, create an occurring view object under the view child of the window object. Create fields for the new occurring view. Expand the occurring view, select the fields by holding down the Control button, and then drag and drop them onto the window painter. This will create an MCLB with columns corresponding to the fields you have selected.

Multiline Edit Field

Use a **Multiline Edit Field** to enable the end user to type more than one line of text. In a AppBuilder application, you can populate a Multiline Edit Field by typing data at runtime or by mapping data from the rule. You can set a Multiline Edit Field to be non-editable (read only). For instructions on how to add a Multiline Edit Field object to a window, see Inserting an Object in a Window. For instructions to modify the object properties, see Viewing and Modifying Window Object Properties.

Sample Multiline Edit Field



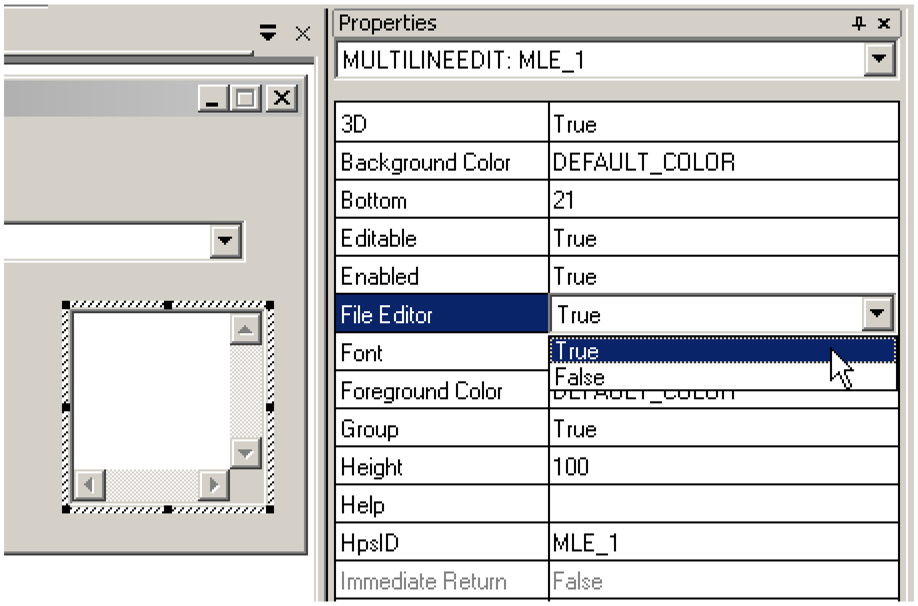
After you insert a Multiline Edit Field object in a window, you can link a Multiline Edit Field to a Field using the Link property in the Properties window for the Multiline Edit Field. Click the Multiline Edit Field in the Window Painter tab. The Properties window is displayed. Select the **Link** property and from the drop-down list of multiple occurring Field objects, select a Field.

Do not link fields with type decimal, integer, date, or time to a multiline edit field.

Opening a File Editor

To allow the end user to view, but not modify, the contents of a text file, you can insert a File Editor object. Click the Multiline Edit Field in the Window Painter tab. The Properties window is displayed. Change the File Editor property to **True**. This automatically disables Mandatory and Immediate Return.

Changing the File Editor property



3D	True
Background Color	DEFAULT_COLOR
Bottom	21
Editable	True
Enabled	True
File Editor	True
Font	
Foreground Color	DEFAULT_COLOR
Group	True
Height	100
Help	
HpsID	MLE_1
Immediate Return	False

Push Button

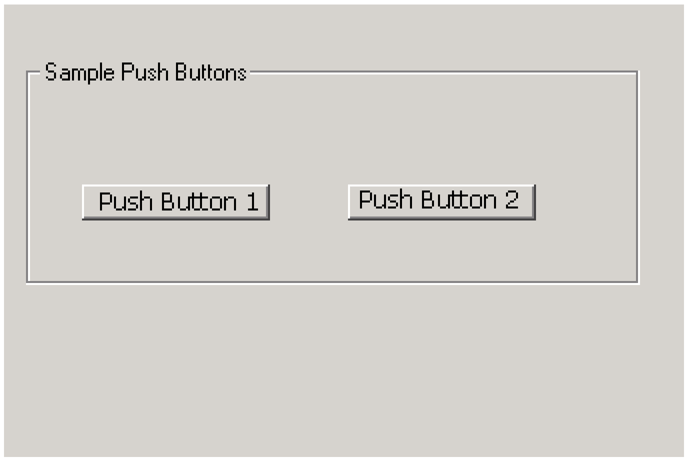
A Push Button text label is displayed on the push button itself. You can select the button using the mouse, space bar, **Enter** key, or an accelerator or mnemonic key sequence. When a user who is registered for an event clicks the push button, the event is sent to the rule. This action returns control to the application. A Push Button has no repository connection; it merely sends an event to the rule.

A Bitmap Push Button is a Push Button displayed as a 3-D bitmap image with a border. When the button is pressed, its border looks sunken. When the button is "up", its border looks raised. When a Bitmap Push Button is disabled, any text within its border is dithered with gray.

⚠ You must use the correct bitmap image type for the specific targeted execution environment.

For instructions to add a Push Button object to a window, see Inserting an Object in a Window. For instructions to modify the object properties, see Viewing and Modifying Window Object Properties. For instructions to create an accelerator for a Push Button to perform an action, see Adding an Accelerator.

Sample Push Buttons



When you insert a Push Button object in a window, do the following:

- You can edit the text in the Push Button using the **Text** property in the Properties window for the Push Button. Click the Push Button. The Properties window is displayed. Enter the appropriate text in the **Text** property.
- You can link the Push Button to a Field using the **Link** property in the Properties window for the Push Button. You must have the following structure in the Hierarchy: at least one Field object under a View under the Window object. Click the Push Button. The Properties window is displayed. Select the **Link** property; and from the drop-down list of possible Field objects in the repository, select a Field.

Adding an Accelerator

You can assign an accelerator or shortcut (which can be a combination of keys) to perform an action. Accelerators can be:

- Virtual keys (function keys **F1** – **F12**)
- Combination of a virtual key and one or more modifier keys (**Alt**, **Ctrl**, or **Shift**)
- Combination of an alphanumeric key and one or more modifier keys.

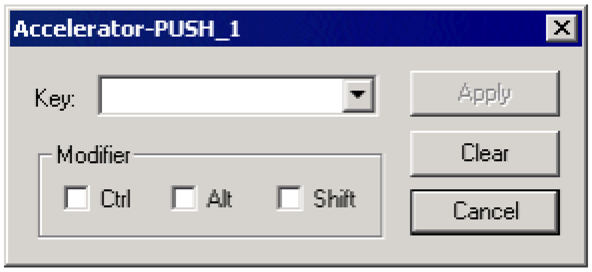
⚠ Shift+F1 combination is not supported for C-runtime

To create an accelerator for a Push Button, complete the following steps:

1. Right-click the Push Button in the Window Painter tab and select **Accelerator** from the pop-up menu. The Accelerator window is displayed.

You can also add an accelerator by selecting **View > Accelerator** from the Construction Workbench menu.

Accelerator window



2. Use the **Key** drop-down list to select the virtual accelerator key for the button.

3. Under Modifier, select one or more check boxes to denote if the accelerator uses a modifier key (**Alt**, **Ctrl**, and/or **Shift**).

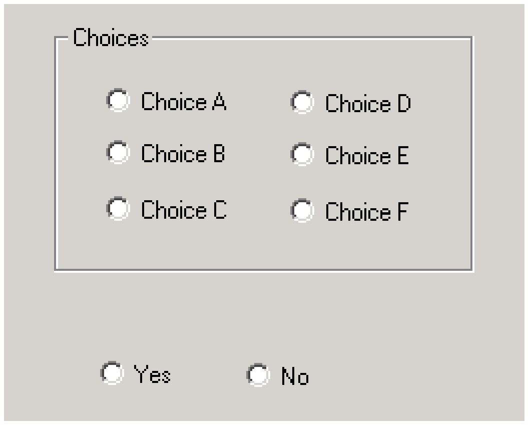
4. Click **Apply**.

Using NLS characters in the mnemonic keystroke sequence is only supported in the C client runtime. Java does not support the use of NLS characters for mnemonics. Accelerators are supported in Java, but only for menu items.

Radio Button

Use a radio button to specify simple selections for field settings. Select the button using the mouse or the space bar. Several radio buttons are typically used to present exclusive alternatives for a single setting. These are often displayed together in a group box and are linked to the same field in the repository. The data contained in the Link field is the system identifier (HPSID) of the selected radio button. For instructions to add a radio button object to a window, see Inserting an Object in a Window. For instructions to modify the object properties, see Viewing and Modifying Window Object Properties.

Sample Radio Buttons



When you insert a Radio Button object in a window, you can edit the text for the radio button using the **Text** property in the Properties window for the Radio Button. To edit text for the radio button, complete the following steps:

1. Click the radio button. The Properties window displays.
2. Type the appropriate text in the **Text** property.
3. Click **File > Commit** to commit changes to the repository.

You can link the radio button to a field using the **Link** property in the Properties window for the radio button. You must have at least one Field object under a View under the Window object in the Hierarchy. To link the radio button to a field, complete the following steps:

1. Click the Push Button. The Properties window displays.

2. Select the **Link** property; and from the drop-down list of possible Field objects in the repository, select a Field.

The **Link** field specifies the system identifier (HPSID) for all the buttons in a given selection. In the example in Sample Radio Buttons, the choices in the Group Box would all have the same system identifier for a link. The **Yes** and **No** buttons would have a different system identifier than the one for the Group Box.

It is not possible to select a Radio Button on an HTML servlet if the Radio Button does not have a link.

3. Click **File > Commit** to commit changes to the repository.

A mnemonic key can be defined by prefacing any character in a Text attribute with an ampersand, "&".
Using NLS characters in the mnemonic keystroke sequence is only supported in the C client runtime. Java does not support the use of NLS characters for mnemonics.

Radio Button Behavior

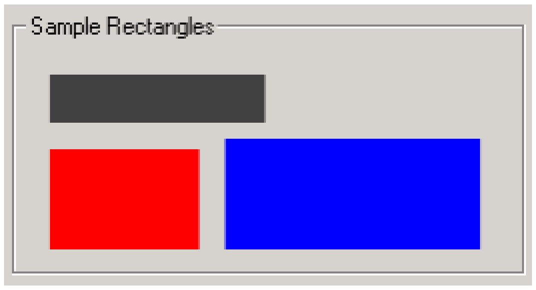
As explained above, radio buttons can be linked to a Field within a hierarchy. If that field is blank or cleared, then the first button in the group is selected. If the field is not blank and does not match any of the buttons in the group, no button is selected.

However, if the field value is not blank and does not match any of the radiobuttons, then when tabbing into the group the focus will go to the first button in the group. If the AUTOSELECT setting is ON or TRUE, then the field will be changed to match the radiobutton with focus.

Rectangle

A Rectangle, like an Ellipse, adds graphical interest or clarification to the window and contains no link to a repository field. For instructions to add a Rectangle object to a window, see Inserting an Object in a Window. For instructions to modify the object properties, see Viewing and Modifying Window Object Properties.

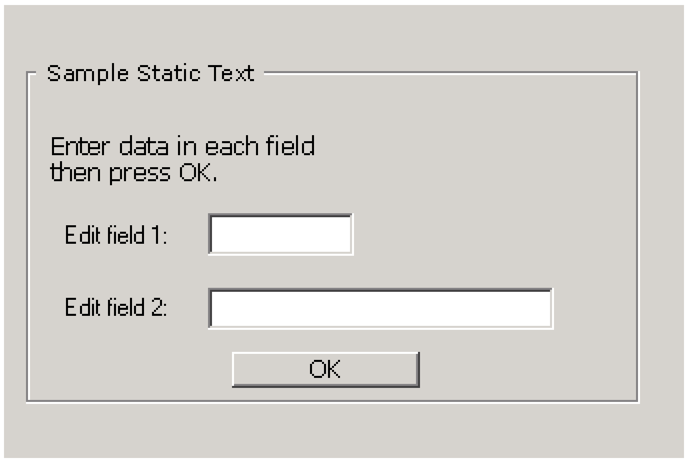
Sample Rectangles



Static Text

Static Text is a text label placed as needed anywhere in a window. It is for informational use within the window and does not link to a repository field. End Users cannot tab to a Static Text object. Static Text cannot be included in the window's tab sequence. (See Previewing a Window). For instructions to add a Static Text object to a window, see Inserting an Object in a Window. For instructions to modify the object properties, see Viewing and Modifying Window Object Properties.

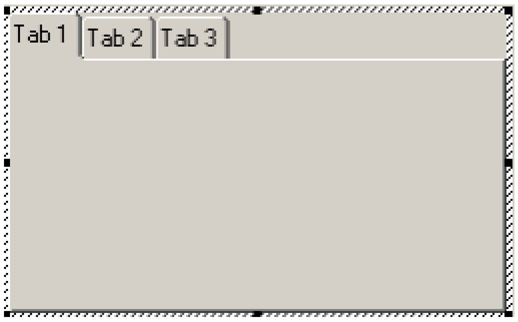
Sample Static Text



Tab Controls

A tab control is an interface control that you can use to group related child controls on tab pages on a single window panel rather than create multiple window panels. You can create tab controls in Window Painter. You can put any control on a tab page except a tab control. You can also drag and drop an object from the window into the tab control or from the tab control to the window.

Sample Tab Control



Tab controls can be oriented along the top of a window panel or along the bottom of a window panel. There is no limit to the number of tab controls you can put on a window panel. You can set a number of properties on the tab control and its respective tab pages, including the width of the tabs.

To create a tab control in AppBuilder, complete the following steps:

1. With the Window Painter open in the Work Area, click the Tab Control button and place the control where you wish.
2. Specify the settings (background color, tab width, etc.) desired.

You can add additional tabs or delete tabs from a right-mouse menu click on the tab control.

The C Runtime does not support multiple rows of tabs. If you have multiple rows in a C application, those tabs will be displayed in one row at runtime.

Using 3270 Window Painter

The 3270 Window Painter tool is similar to the workstation Window Painter but is used for creating windows for 3270 terminals connected to a mainframe. Most of the differences between them are because 3270 terminals support a more limited set of objects than workstations. Topics include:

- Opening a 3270 Window
- Working with the Background Panel
- Understanding Color and Font Support
- Understanding Support for Window Objects
- Using the Function Key Editor

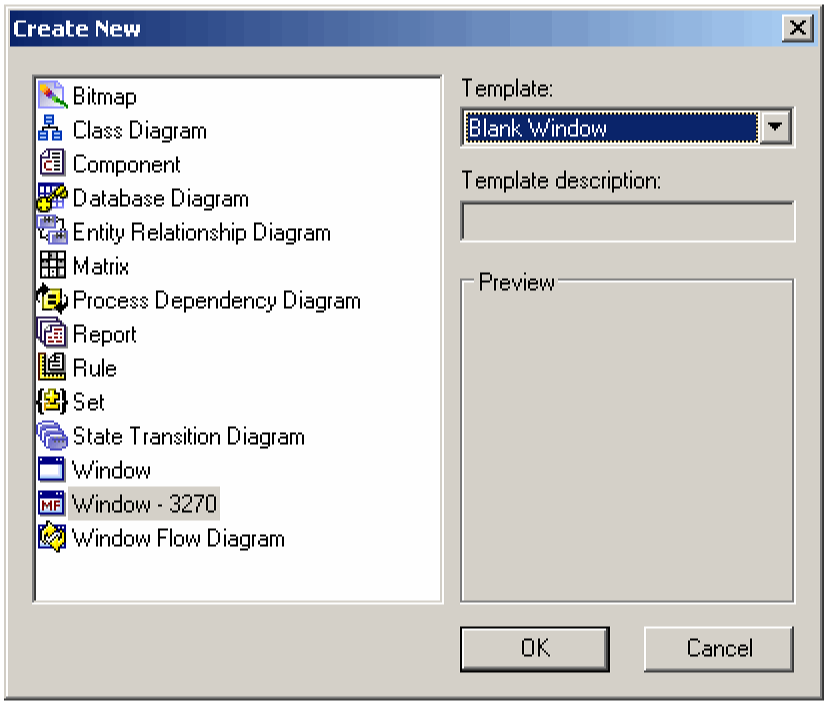
There is a limitation for 3270 window views of 4026 entities (views and fields). This includes multiple occurring subviews, where each occurrence of a view or field in a multiple occurring view is calculated as another entity. If you exceed the limit, the window fails to prepare.

Opening a 3270 Window

When you select the **File > New** or **File > Open** choice in the Construction Workbench, the Create New dialog and the Open Repository Object dialog both include the 3270 window as an option. Available 3270 window templates, if any, can be selected from the Template list.

The 3270 Window Painter layout consists of the 3270 Window Painter tool tab and the Attributes List Box. Both of these windows open in the Work area and can be sized independently.

Create New dialog with Window - 3270 option selected



There are a number of differences between the workstation and 3270. If a menu choice or other element is not explicitly mentioned, assume that it works in the same way as in the workstation Window Painter. In most cases where a feature does not work in the 3270 version, such as the Tabbing Editor and Tab Order, the menu choice is grayed out.

The window object (in effect, background of the window) is black, as opposed to gray in the workstation version, and it cannot be changed to a different color. The only supported object types are Combo Boxes, Edit Fields, Multicolumn List Boxes, and Static Text. The tool palette has icons only for Combo Boxes, Edit Fields, Multicolumn List Boxes, and Static Text enabled. The attributes pop-up windows for 3270 Window Painter objects usually contain fewer fields, due to the simpler nature of 3270 windows. Colors for text can be changed in a 3270 window; however, colors for objects such as fields and Combo Boxes cannot. The Color pop-up window only contains a few colors, and scroll bars for custom colors are not available.

There are also several differences between the 3270 Menu Editor and the Windows version of the Menu Editor. The 3270 version of the Menu Editor does not include separator objects. (Separators copied from workstation windows with *Copy Menu* and *Paste* appear as blank lines in the 3270 menu. The 3270 version of the Menu Editor does not provide attributes pop-up windows for menu items. Tabbing Editor and Tab Order are not supported in 3270 Window Painter.

Working with the Background Panel

A 3270 panel initially measures 24 rows high by 80 columns wide. You can make the panel smaller but not larger. The size and position of objects in the window are expressed in row and column values. Note that workstation and 3270 displays are oriented differently: (1,1) is lower left on the workstation display, but upper left on the 3270 display.

The caption bar does not display on the 3270 terminal at runtime. A caption displays only if you paint it as a Static Text object in the window itself.

You cannot place an object in the leftmost column of the panel because 3270 Converse uses an attribute byte in the space before each object on the 3270 display.

Understanding Color and Font Support

The 3270 Converse supports IBM 3178-2 and IBM 3179-2 terminals. The IBM 3178-2 terminal is a monochrome display station that supports the single color green. The IBM 3179-2 terminal is a color display station that supports four base colors (red, blue, green, and white) and three extended colors (yellow, pink, and turquoise). The extended highlighting feature reverse video is used to implement background color. The background color of the panel and all window controls is set to gray in 3270 Window Painter for visibility, even though the default background color is black in 3270 terminals.

Edit Fields, Combo Boxes, and Multicolumn List Box cells are delineated by square brackets in the 3270 Window Painter, except that an Edit Field or Multicolumn List Box cell with a length of one is delineated by an underscore. The color of the brackets or underscore corresponds to the color of the input text for the object.

You specify the color of label and input text in the *Color* combo box in the Attributes pop-up window for the object. The available colors are red, blue, green, white, yellow, magenta, and cyan, in order as they correspond with the IBM 3179-2 colors listed above. The default is white. When label text in a workstation window is copied to a 3270 window, the color specified by the user is translated as closely as possible, except that black is translated to white. Label and input text are set to a standard nonproportional font, which cannot be changed.

3270 Converse determines the characteristics of the terminal at runtime and constructs the 3270 data stream for the device. Text colors are translated as closely as possible; the terminal may highlight the current screen item by altering its color. Brackets are not displayed; terminal devices that support extended 3270 attributes denote input fields with an underscored data entry area. Reverse video red is used to highlight fields in error. For monochrome 3270 devices, highlighting the single color denotes fields in error. For further information, refer to the information on SET_FIELD_COLOR in the *System Components Reference Guide* .

Understanding Support for Window Objects

The 3270 Window Painter supports the following window objects: Edit Fields, Static Text objects, function keys, Combo Boxes, and Multicolumn List Boxes. You can create menu-bar choices and first-level pull-down menu items with the 3270 Window Painter menu editor, as described in Menu Bar.

3270 Converse uses an attribute byte in the space before each object on the 3270 screen. Generally speaking, you must accommodate this byte; the *Verify* choice in the 3270 Window Painter *Verify* menu issues an error message for insufficient space. In two cases, the 3270 Window Painter takes care of the matter for you: it prevents you from placing objects in the leftmost column of the window, and it automatically adds a space before each cell in a multi-column List Box.

The tab order of window objects is left to right and top to bottom, and cannot be changed. 3270 end users cannot tab to a non-editable (protected) Edit Field or non-editable Combo Box. The Tabbing Editor and Tab Order tools are not available in 3270 Window Painter.

Some of the functions available, depending on the objects in the window, are similar to the Window Painter tools:

- Using the Menu Editor
- Using the Multicolumn List Box Editor

Available only in 3270 Window Painter is:

- Using the Function Key Editor

The window controls that are supported in 3270 Window Painter include:

- Combo Box Object
- Edit Field Object
- Function-key Object
- Menu Bar
- Multicolumn List Box (MCLB) Object
- Static Text Object

Combo Box Object

You link the Combo Box object to a character field defined in the repository with an online validation set that contains the permissible values for the field. At execution, 3270 Converse verifies that the character string entered in the field is contained in the associated set. If an invalid character string is entered, 3270 Converse displays a scrollable pop-up list of values – a Combo Box, in effect – from which the end user can select a valid value. Combo box objects are case sensitive.

Editable Combo Boxes are supported in functionally different ways on the workstation and the host. In a workstation window, an editable Combo Box is one in which the end user can enter a value that does not occur in the associated set. In a 3270 window, an editable Combo Box is one in which the end user can type in the input area of the Combo Box, but cannot enter a value that is not contained in the set.

A Combo Box is editable, if you set the Combo Box property Style to DropDown; this property is selected by default. To make the Combo Box non-editable, set the Style to be DropDownList. As this implies, the 3270 end user cannot type in the data input area of the Combo Box in the default case?the list of valid values can only be displayed by pressing the PROMPT function key. Note, too, that 3270 end users cannot tab to a non-editable Combo Box. In cases where you want a Combo Box to be non-editable on the workstation and "editable" on the host?that is, in which you want 3270 end users to be able to tab to the Combo Box and type in its data entry area? *Style* should be set to DropDownList for the workstation Combo Box, and should be set to DropDown for the 3270 Combo Box.

You can resize a Combo Box vertically only in multiples of its row height, which cannot be changed.

⚠ Support for lookup sets, error sets, and set types for integers cannot be used for Combo Boxes in 3270 Converse.

Edit Field Object

An Edit Field loaded from the repository is sized initially according to its length attribute or picture display, except that the date and time fields for which no picture display has been specified are sized initially with lengths of 10 and 11, respectively. You cannot size the field yourself except by modifying these values. The square brackets surrounding the field are included in the length specification.

If you want to be able to change the width of an Edit Field on the 3270 window that has a linked int field of 15 or 31, you must specify a value in the Display Pic field for that Edit Field.

You cannot resize Edit Fields vertically in a 3270 window. The RESIZE environment variable has no effect in a 3270 window. Unlike the workstation Window Painter, 3270 Window Painter does not have a Resize property in the Edit Field object to allow you to change the height of an Edit Field.

There is a form-fit property, provided for an Edit Field, with which horizontal resizing can be done. We recommend that you avoid using this property for resizing, as this is provided for customers who are using a non-standard data object that has a different data length.

⚠ When defining the window layout, you must take care specifying the lengths of edit fields. The maximum length of the formatted string should be considered when determining the placement of adjacent fields. This is more important with DBCS strings as in addition to the truncation of the string, terminal errors (PROGxxx) can occur due to mismatched SO-SI pairs.

Function-key Object

Function-key objects are painted in the last row of the window with values F2 - F24 in order. (F1 is reserved for help.) You can change a function-key assignment in the **PF Key** combo box in the Attributes pop-up window for the object. Push buttons copied from a workstation window are painted as function keys.

The text for a function key defaults to Push. Change its text by typing over the **Text** field in the Attributes pop-up window for the object. The text is displayed beside the function-key specification in the 3270 window at runtime. The maximum text string that can be assigned to a function key is 28 bytes.

The system identifier (HPSID) of a function key defaults to the characters' "ID" followed by a system-assigned number. Change its system identifier (HPSID) by typing over the **HPSID** field in the Attributes pop-up window for the object. The system identifier is returned to the invoking rule when the end user presses the function key at runtime. Note that the *Verify* choice in the 3270 Window Painter **Verify** menu issues an error message for the first instance of a duplicate system identifier only.

The function-key display wraps to a new line as necessary, depending on the current width of the panel. You can display 24 function keys on a 3270 terminal in two rows. The **Verify** choice in the 3270 Window Painter *Verify* menu issues an error message if you exceed the prescribed number of keys or rows.

Default values of function keys conform as closely as possible to Common User Access (CUA) guidelines, and are listed in the *Enterprise Application Guide* . Except when 3270 Converse displays the default F7 or F8 keys for a scrollable List Box, only user-defined keys and their associated text strings are displayed on the 3270 terminal.

Menu Bar

The menu bar (including the menu bar for a pop-up window) is displayed at the top of the 3270 screen in as many rows as needed; any data overlaid by the menu bar are retained. The default F10 key toggles the display on or off. The default is on.

Only menu bar items and first-level pull-down menu items are supported in 3270 windows. Specify the text for a menu item in the menu editor **Menu text** field. For pull-down menu items, the text automatically populates the system identifier (HPSID) area in the brackets beside the text area. When the end user selects the item at runtime, 3270 Converse returns the system identifier (HPSID) to the invoking rule.

Change the item's system identifier (HPSID) by typing over the **HPSID** field in its Attributes pop-up window. Subsequently modifying the **Menu text** field does not alter the item's system identifier. The **HPSID** area of menu bar items is empty, following CUA guidelines. You cannot display an Attributes pop-up window for menu bar items. Menus are not displayed in the 3270 Window Painter.

Menu item separators, mnemonic keys, and shortcut keys are not supported. To disable menu items, use the component SET_MENU_MODE, described in the *System Components Reference Guide* . Otherwise usage is the same as that described for Window Painter.

3270 pull-down menus are limited to 99 choices; menu bars to 38 choices or 11 rows. Display of any menu choice on the 3270 device is limited to 28 bytes. Pull-down menus are scrollable.

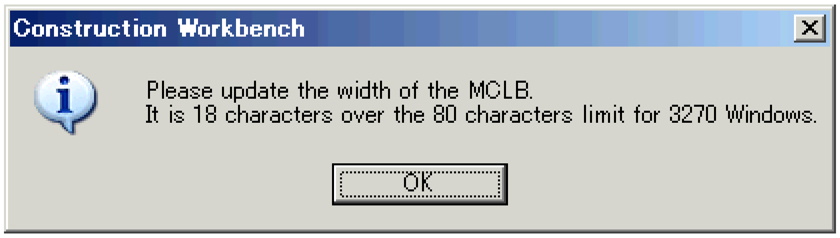
Multicolumn List Box (MCLB) Object

3270 Converse allows only one Multicolumn List Box (MCLB) in a window. For that reason, the **Repository query/Create MCLB** choice in the **Objects** menu is disabled if a Multicolumn List Box is already displayed in the panel. You can use the **Paste** choice in the **Edit** menu if you want to create another Multicolumn List Box object in the window. An error message is issued for the first illegal List Box only. It is your responsibility to delete any extra Multicolumn List Box objects from the panel before you try to prepare it. Panels with multiple List Box objects do not prepare.

Generally, a Multicolumn List Box column is sized initially according to its length attribute or picture display, plus a preceding space to accommodate an attribute byte. Date and time fields for which no picture display has been specified are sized initially with lengths of 10 and 11, respectively, plus a preceding space to accommodate an attribute byte. The square brackets surrounding the column are included in the length specification.

You can increase the width of a Multicolumn List Box object in the 3270 Window Painter only by adding columns or by increasing the width of columns in the Multicolumn List Box Editor, as described in Using the Multicolumn List Box Editor. 3270 Converse does not support horizontal scrolling, so it is important to be sure columns are wide enough to accommodate foreseeable input. That is especially the case when you copy from a workstation window a List Box that uses small fonts to display input text. Note that a column made narrower than its length attribute or picture display will not be displayed correctly at runtime. When a MCLB is wider than 3270 window width, the following warning message is displayed when you click **Apply** button in MCLB Editor:

MCLB warning message for 3270 Window


Construction Workbench

Please update the width of the MCLB.
It is 18 characters over the 80 characters limit for 3270 Windows.

You can resize a Multicolumn List Box vertically only in multiples of its row height, which cannot be changed. Only as many rows are allowed as there are occurrences specified for the view in the repository. Two rows are displayed by default. Any long screen literal you have assigned to a field in the repository is automatically displayed as the heading of the column with which it is linked; the heading is truncated if it is longer than the column is wide. You can change the heading as described in Using the Multicolumn List Box Editor.

When using MCLB Editor for 3270 window, be aware that in 3270 Window Painter, the headers are put in one row above the MCLB and do not take up a row.

Static Text Object

Static text does not wrap to a new line in the 3270 Window Painter. You cannot resize a Static Text object with the mouse.

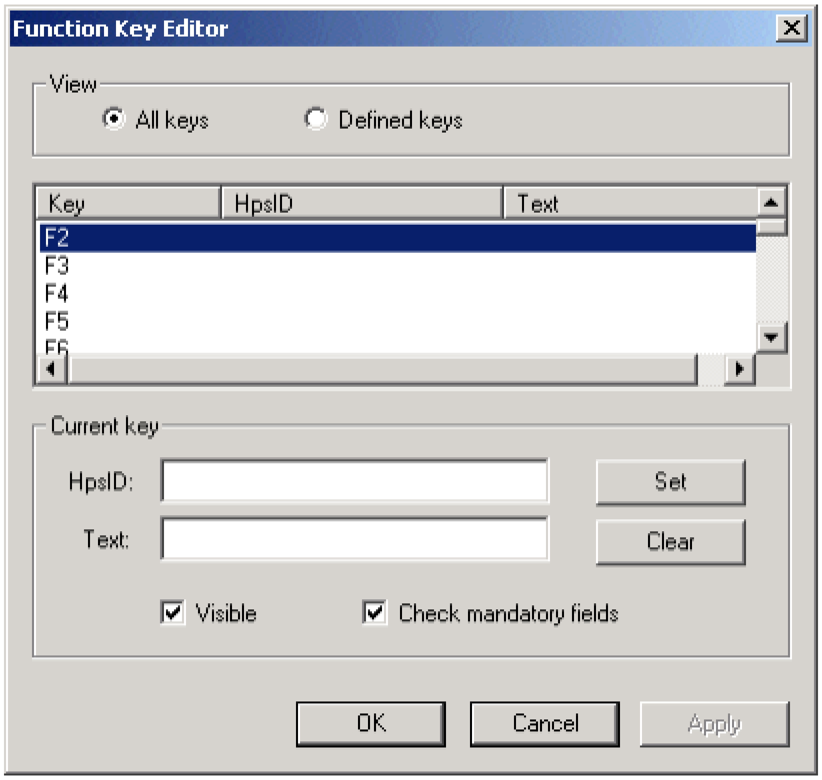
Using the Function Key Editor

The 3270 Window Painter supports function key definition; this is one of the differences between 3270 Window Painter and workstation Window Painter. Assigned function key values are displayed at the bottom of the window in standard "F<#>=<text>" form. When the user presses a function key, its system identifier (HPSID) is passed to the conversing rule, in the same way as a Push Button in a workstation window.

Use the Function Key Editor to assign significance to the function keys on a 3270 terminal for a particular 3270 window. Open the Function Key Editor by selecting **Tools > Function Key Editor**.

Use the Copy function keys command to duplicate the function key assignments from one 3270 window to another.

Function Key Editor dialog



The Function Key Editor window consists of these parts:

- The View group, which you can use to view all the functions keys (All keys) or only certain defined function keys (Defined keys). The choice determines the scope of function keys appearing in the List Box.
- A List Box showing the function keys (depending on whether All Keys or Defined Keys is chosen) and their assignments.

- The Current Key group, which includes information about the currently selected key in the list of keys. This information includes the system identifier (HpsID), the text associated with that function key, whether it is visible, and whether the system should check the field as mandatory.
- The Set and Clear buttons, which you can use to change the system identifier and text for a function key.

When a key is selected in the List Box, its system identifier and Text are entered in the appropriate fields. The system identifier is used by the rule that converses the window; therefore, it is probably a good idea to use a significant mnemonic for each system identifier.

Click **Apply** to save the key assignments; click **Cancel** to close the Function Key Editor without saving changes; and click **OK** to save the key assignments and exit the Function Key Editor.

Window Painter Objects and Their Attributes

This chapter provides information about windows created with Window Painter and the objects (controls) available on them.

Understanding Properties of Window Objects

Every window and object (control) on it has properties which are displayed on a Properties Page (see, for example, Parts of Window Painter). The following sections describe the properties of window objects. Window Painter Objects and Properties lists all objects and properties.

3D

This is a boolean value for viewing an object as three-dimensional.

Available setting options are: True, False.

It applies to: Edit Field, List Box, Multiline Edit Field, Window.

Autocall

This is a boolean value for the behavior of a Multicolumn List box (MCLB) when beginning or ending of data is reached in the view data structure associated with the MCLB. When AutoCall is set to True, control returns to the invoking rule, allowing the rule to retrieve more data. AutoCall should not be used in conjunction with set_virtual_listbox_size, described in the AppBuilder documentation.

Available setting options are: True, False.

It applies to: MCLB.

Auto Select

If a MCLB's Auto Select checkbox is selected, end users select an item by putting the cursor on it; the item from which the end user moves the cursor is deselected. Otherwise, end users can move the cursor without selecting an item. Auto Select is ignored if Selection Type is set to multiple select. By default, Auto Select is not selected.

Available setting options are: True, False.

It applies to: List Box, MCLB.

Auto Tab

This is a boolean value for automatic tabbing. When AutoTab is set to True, focus will automatically be given to the next control in the tab order once the defined maximum value length for the control has been reached.

Available setting options are: True, False.

It applies to: Edit Field.

Background Color

The background color of the object. You can select a standard color from a menu of colors, or you can create a custom color. To create a custom color, select Custom from the drop-down list and use the Windows Color Picker to create the color. Customer colors are stored in hexadecimal format denoted as hBBGGRR.

Available setting options are: Custom, Default_Color, Black, White, Darkgray, Gray, Lightgray, Darkblue, Blue, Darkgreen, Green, Darkcyan, Cyan, Darkred, Red, Darkmagenta, Magenta, Darkyellow, Yellow, Pink, Brown.

It applies to: Checkbox, Combo box, Edit field, Ellipse, List box, MCLB, Multiline edit field, Radio button, Rectangle, Static Text, Tab Control, Tab Page, Window

⚠ Java and Windows have different named colors as set in Workbench options.

Border Type

This defines the border type of the window.

Java does not support resized windows (Border_Sizable). Resizing changes the size of the side(s) dragged. It does not scale the window and the controls on it.

Available setting options are: Border_Dialog, Border_Sizable, Border_None.

It applies to: Window.

Bottom

This is the Integer value for the Y-Coordinate of the bottom edge of an object's location. For a Window, this value is relative to the bottom edge of the screen. For all other controls, this value is relative to the bottom edge of its containing Window. The unit used is based on the (containing) Window's Coordinate Type value.

Values for objects on the window are relative to the originally-sized window. Resizing the window does not change the location of objects on the window.

Available setting options are: (Numeric value).

It applies to: Bitmap, Chart, Checkbox, Combo box, Edit field, Ellipse, Group Box, Hot Spot, List Box, MCLB, Multiline edit, Pushbutton, Radio button, Rectangle, Static Text, Tab Control, Window.

Chart Type

This is the type of chart used for representing plot data graphically. Charts are not displayed at design time.

Available setting options are: Linechart2D, Barchart2D, Piechart2D, Stackedbarchart2D, Columnchart2D, Smoothlinechart2D, Scatterchart2D, Areachart2D, Barlinechart2D, Linechart3D, Barchart3D, Piechart3D, Stackedbarchart3D, Columnchart3D, Smoothlinechart3D, Areachart3D, Perbarchart3D, Barlinechart3D.

It applies to: Chart.

Check Mandatory Field

If Check Mandatory Fields is selected, all Fields with the Mandatory property set to True must be satisfied before the selected event will occur. By default, Check Mandatory Fields is set to False.

Available setting options are: True, False.

It applies to: Pushbutton.

Close Text

The window's Close Text field specifies the text returned to the invoking rule when the end user exits the window with the System menu Close choice. The close function is disabled in a secondary window if the Close Text field is blank. It defaults to blank.

Available setting options are: (Text field).

It applies to: Window.

Coordinate Type

This unit is used for the size and the position of all controls within a Window.

Available setting options are: Pixel, Char.

It applies to: Window.

Country

This specifies the country (locale) to be used for all controls within a window. Display properties will be based on the value for this setting.

Available setting options are: System, Albania, Argentina, Australia, Austria, Belgium, Brazil, Canada_English, Canada_French, China, Czechoslovakia, Denmark, Eurozone, Finland, France, Germany, Greece, Hungary, Iceland, Ireland, Italy, Japan, Luxembourg, Netherlands, New_Zealand, Norway, Poland, Portugal, Romania, South_Africa, South_Korea, Spain, Sweden, Switzerland, Taiwan, Thailand, Thailand_Buddhist, Turkey, United_Kingdom, United_States, Yugoslavia.

It applies to: Window.

Domain

A domain value can be specified as a set contained within a Window. When a domain value is specified, only the values from that set can be selected in a control. In a Combo Box, a field from an Occurring View may also be specified for a domain value.

Available setting options are: SET, Field of occurring field (Combo Box only).

It applies to: Combo box, Edit field.

Editable

This denotes whether the field can be changed. If the Editable property is set to False, it is still possible to select the field, but you cannot type or enter anything.

Available setting options are: True, False.

It applies to: Edit field, Multiline edit.

Enabled

Thus indicates whether the field is enabled. When a field or button is not enabled, you cannot select or click into it.

Available setting options are: True, False.

It applies to: Edit field, Multiline edit, Pushbutton.

Enter Key

This is the HpsID of the default Pushbutton for the window.

Available setting options are: (Enter HpsID).

It applies to: Window.

File Editor

Selecting File Editor for a multiline edit object lets end users view, but not modify, the contents of a text file. When File Editor is selected, the Mandatory and Immediate Return checkboxes are dimmed.

Available setting options are: True, False.

It applies to: Multiline edit.

Font

The Font setting lets you specify the font for label and input text. It defaults to System 8. Use the default font whenever possible so that windows will be device-independent. The Font attribute determines the height of edit fields, radio buttons, and check box objects. You can alter the height of a radio button or check box object only by manipulating this attribute. You can alter the height of an edit field with the mouse only if you include the line RESIZE=YES in the [Window Painter] section of the HPS.INI file. The multiline edit field, file editor, and MCLB support the fonts System 8, Swiss 8, Roman 8, Modern 8, Modern 10, and Modern 12 only. The Roman and Swiss 24 fonts are not supported in windows targeted for UNIX execution, and are translated to Roman and Swiss 18, respectively. The Swiss 12 font is not supported in windows targeted for UNIX execution, and is translated to System 8. Otherwise, the available fonts include: System 8; Modern 8, 10, 12; Roman 8, 10, 12, 14, 18, 24; Swiss 8, 10, 12, 14, 18, and 24.

Strictly for Java runtime, the font name in the font.ini is generally an alias that is defined by the JVM but can be a physical font name. Runtime, both C and Java, resolves these logical font names from the panel file to physical fonts using whatever font.ini is deployed for the runtime environment. Thus it is possible to change the actual fonts used at runtime to be different than at window design time.

The list of fonts displayed here comes from those defined in font.ini. See *INI Settings Reference Guide* .

Available setting options are: System 8; Modern 8, 10, 12; Roman 8, 10, 12, 14, 18, 24; Swiss 8, 10, 12, 14, 18, and 24.

It applies to: Checkbox, Combo box, Edit field, Group box, List Box, MCLB, Multiline edit, Pushbutton, Radio button, Static text, Tab control, Tab page.

Foreground Color

This is the foreground color for the object. In the case of text, it is the color of the text against the background color.

Java and Windows colors are different, each set in Window Painter set of options of Workbench Options dialog.

Available setting options are: Custom, Default_Color, Black, White, Darkgray, Gray, Lightgray, Darkblue, Blue, Darkgreen, Green, Darkcyan, Cyan, Darkred, Red, Darkmagenta, Magenta, Darkyellow, Yellow, Pink, Brown.

It applies to: Checkbox, Combo box, Edit field, Group box, List box, MCLB, Multiline edit, Radio button, Static text, Tab page.

Form Fit

This specifies whether a field is form fit. If Form Fit is selected in an edit field's attributes, Window Painter dynamically sizes the edit field in accordance with its length attribute or picture display. You cannot size the field yourself except by modifying these values. Note that you need not update the field's data link if you change its length or picture in the Hierarchy Diagrammer, and that Window Painter automatically adjusts the field's length to accommodate date, time, currency, or numeric formats specified in the edit field's attributes. Date and time fields for which no picture display has been specified in the Hierarchy Diagrammer are sized initially with lengths of 10 and 11, respectively. Form Fit is selected by default for edit fields loaded from the repository; otherwise, you must select it yourself. Form Fit is only valid when a datalink is attached to the edit field. The attribute is not persistent; you must reselect it whenever you open the window. If Form Fit is not selected, you can alter the length of an edit field as necessary.

Available setting options are: True, False.

It applies to: Edit field.

Group

This sets the objects that start a group. A Group is a collection of a logical set of objects. When window controls are grouped, the user can use arrow keys to move from one item in the group to another without having to use the Tab key. An example for using Group is in a series of Radio Buttons. You set which controls start a Group through the object's Properties or through the Tabbing Editor. To start a Group, set the Group property to True. To end the group, specify the Group attribute for the next control in the tab order as True; in other words, when you start a new group, the previous group is ended.

Available setting options are: True, False.

It applies to: Checkbox, Combo box, Edit field, List box, MCLB, Multiline edit, Pushbutton, Radio button, Tab control.

HeaderBackgroundColor

This is the background color of an MCLB header row. You can select a standard color from a menu of colors, or you can create a custom color. To create a custom color, select Custom from the drop-down list and use the Windows Color Picker to create the color. Customer colors are stored in hexadecimal format denoted as hBBGGRR.

Available setting options are: Custom, Default_Color, Black, White, Darkgray, Gray, Lightgray, Darkblue, Blue, Darkgreen, Green, Darkcyan, Cyan, Darkred, Red, Darkmagenta, Magenta, Darkyellow, Yellow, Pink, Brown.

It applies to: MCLB.

HeaderFont

This is the HeaderFont setting lets you specify the font for an MCLB header row. It defaults to System 8. Use the default font whenever possible so that windows will be device-independent. The Font attribute determines the height of the MCLB header row. The MCLB supports the fonts System 8, Swiss 8, Roman 8, Modern 8, Modern 10, and Modern 12 only. The Roman and Swiss 24 fonts are not supported in windows targeted for UNIX execution, and are translated to Roman and Swiss 18, respectively. The Swiss 12 font is not supported in windows targeted for UNIX execution, and is translated to System 8. Otherwise, the available fonts include: System 8; Modern 8, 10, 12; Roman 8, 10, 12, 14, 18, 24; Swiss 8, 10, 12, 14, 18, and 24. See information on Font above.

Available setting options are: the available fonts include: System 8; Modern 8, 10, 12; Roman 8, 10, 12, 14, 18, 24; Swiss 8, 10, 12, 14, 18, and 24.

It applies to: MCLB.

HeaderForegroundColor

HeaderForegroundColor specifies the foreground (font) color of an MCLB header row.

Available setting options are: Custom, Default_Color, Black, White, Darkgray, Gray, Lightgray, Darkblue, Blue, Darkgreen, Green, Darkcyan, Cyan, Darkred, Red, Darkmagenta, Magenta, Darkyellow, Yellow, Pink, Brown.

It applies to: MCLB.

Height

This is the Integer value for the Height of the object. The unit used is based on the (containing) WINDOW's Coordinate Type value.

Available setting options are: (integer value).

It applies to: Bitmap, Chart, Checkbox, Combo box, Edit field, Ellipse, Group Box, Hot spot, Listbox, MCLB, Multiline edit, Pushbutton, Radio button, Rectangle, Static text, Tab control, Window.

Help

This is the ASCII text help topic displayed when the user presses F1 (also called Simple AppBuilder Help). The text is limited to 30,000 characters. The Help contents display in a separate window for Windows applications.

Available setting options are: (text).

It applies to: Checkbox, Combo box, Edit field, List box, MCLB, Multiline edit, Pushbutton, Radio button, Tab control, Window.

Horizontal Scroll Bar

This denotes whether the object has a horizontal scroll bar (SHOW_ALWAYS or SHOW_NEVER).

> ⚠ Scroll bars are not supported in Java.

Available setting options are: Show_Always, Show_Never.

It applies to: Window.

HpsID

This is the system identifier used by other AppBuilder objects to reference the object. Every HpsID must be unique for the application to work correctly at runtime. The Verify command identifies duplicate or nonexistent HpsIDs in window objects.

Available setting options are: (generated).

It applies to: Bitmap, Chart, Checkbox, Combo box, Edit field, Ellipse, Group box, Hot spot, List box, MCLB, Multiline edit, Pushbutton, Radio button, Rectangle, Static text, Tab control, Tab page.

Icon

This is the window icon. This icon must be added (as a bitmap object) to the window in the hierarchy first; then it can be selected in the drop-down list of the Property window.

Available setting options are: (select the object from the drop-down list).

It applies to: Window.

Ignore Validation

This indicates whether the user can select the object even if some mandatory fields contain invalid data or if the window contains an error condition.

Available setting options are: True, False.

It applies to: Hot spot, Pushbutton.

Immediate Return

This denotes whether the control generates an event at execution time when data is changed. If an object's Immediate Return property is selected, control is returned to the invoking rule when the end user navigates away from the modified object or, in the case of a read-only object, when the end user double-clicks on the object. The HpsID of the object is returned to the rule. By default, Immediate Return is not selected.

Available setting options are: True, False.

It applies to: Checkbox, Combo box, Edit field, List box, Multiline edit, Radio button.

Justification

The Justification property of a control determines the position of the displayed text relative to the left and right edges of the control.

Available setting options are: Left, Center, Right.

It applies to: Static text, Tab page.

Label Height

This is the Integer value for the height of the Tab Control. The unit used is based on the containing window's Coordinate Type value.

Available setting options are: (numerical value).

It applies to: Tab page.

Label Image

Label Image is used to set the display image to be used by a tab page. As for Icons, add the graphic as a bitmap to the window and then select it from a drop-down list in this field.

Available setting options are: (drop-down list).

It applies to: Tab page.

Label Width

Label Width is used to specify the width of a Tab Page's label. This value can only be set by the user if its parent Tab Control's Tab Style property is set to FIXWIDTH or HTML.

Available setting options are: (numerical value).

It applies to: Tab page

Left

This is the Integer value for the X-Coordinate for the left edge of an object's location. For a window, this value is relative to the left edge of the screen. For all other controls, this value is relative to the left edge of its containing window. The unit used is based on the (containing) window's Coordinate Type value.

Available setting options are: (numeric value).

It applies to: Chart, Checkbox, Combo box, Edit field, Ellipse, Group box, Hot spot, List box, MCLB, Multiline edit, Pushbutton, Radio button, Rectangle, Static text, Tab control, Window.

Line draw

Line draw lets you display the object with horizontal and vertical lines, horizontal lines only, vertical lines only, or no lines at all. It defaults to horizontal and vertical lines.

Available setting options are: Nolines, Vlines, Hlines, VHlines.

It applies to: MCLB.

Link

This indicates whether the object is linked. A data link is an association between an object and an entity in the current repository. Data links are automatically created for edit fields and MCLB columns when you copy and paste their corresponding field entities from the Hierarchy Diagrammer or when you paint them with the Object menu Repository Query choices. An object's Link field contains the qualified object name of the linked entity. You can modify the field only by updating the object's data link. You cannot datalink Window Painter objects to field entities of type timestamp.

Available setting options are: (generated or manually entered).

It applies to: Bitmap, Chart, Checkbox, Combo box, Edit field, List box, MCLB, Multiline edit, Pushbutton, Radio button, Window.

Mandatory

This indicates whether the field is mandatory. If an object's Mandatory property is selected, end users must make valid entries in the object before control can be returned to the invoking rule. If end users select a control or menu item before making a valid entry in a mandatory object, a message appears prompting them to make a valid entry. The cursor is automatically sent to the first mandatory object that lacks a valid entry. By default, Mandatory is not selected.

Available setting options are: True, False.

It applies to: Combo box, Edit field, Multiline edit.

Maximize Box

This indicates whether the window includes and displays a Maximize box in the upper right corner during execution.

> ⚠ The Maximize box is displayed if either the Maximize or Minimize property is True. It is enabled only when it is True. It is not included in the window if both Maximize and Minimize are set to False.

Available setting options are: True, False.

It applies to: Chart, Window.

Minimize Box

This indicates whether the window includes and displays a Minimize box in the upper right corner during execution.

> ⚠ The Minimize box is displayed if either the Maximize or Minimize property is True. It is enabled only when it is True. It is not included in the window if both Maximize and Minimize are set to False.

Available setting options are: True, False.

It applies to: Chart, Window.

Multiple Rows

This is a boolean value for displaying largest tabs in a Tab Control on more than one line. This behavior only applies if the sum of contained Tab Page Label Widths exceeds the width of the Tab Control. If True, additional rows of Tabs will be created as necessary. If False, scrolling buttons will be added to the tab control.

The C Runtime does not support multiple rows of tabs. If you have multiple rows in a C application, those tabs will be displayed in one row at runtime.

Available setting options are: True, False.

It applies to: Tab control.

Numbering

When this is selected, the MCLB will include row numbers on each row.

Available setting options are: True, False.

It applies to: MCLB.

Orientation

This is the orientation (location) of Tabs on a Tab Control.

Available setting options are: Top, Bottom.

It applies to: Tab control.

Password

The Password property specifies how content is displayed in an edit field. When the password property is set to true, then an asterisk (*) should be displayed for every typed character.

Available setting options are: True, False.

It applies to: Edit field.

Resize

This controls whether resizing a field is allowed. When Resize is set to False, you can adjust the width, but must set the Resize property to True before you can change the height.

Available setting options are: True, False.

It applies to: Edit field.

Row Select

When Row Select is selected [True] for an MCLB, end users can click on a cell in the MCLB row to select the entire row. Otherwise, only the clicked-on cell is selected. Row Select is selected by default. MCLBs with an Extended selection type do not support Row Select=False.

Available setting options are: True, False.

It applies to: MCLB.

Scroll Lock

If an MCLB's Scroll Lock property is selected, end users can tab to the next row in the MCLB when they reach the end of the current row. At the end of the last row, they can tab to the beginning of the first row. Otherwise, they tab outside the MCLB to the next object, or, if there is no object, the next row. When Scroll Lock is enabled, or when there is no object in the window other than the MCLB (and thus no other object to tab to), end users can exit the list box by pressing Ctrl-Tab. By default, Scroll Lock is not selected.

Available setting options are: True, False.

It applies to: MCLB.

Selection Type

This sets how rows and cells can be selected in the MCLB. In most cases, Row Select must be True.

Available setting options are: Single, Multiple, Extended. If Single is specified, only one row or cell can be selected in the MCLB. If Multiple is specified, the user can select multiple rows within the same column of the MCLB when Row Select is False. To deselect a row or cell, use CTRL + a mouse click. If Extended is set, the user can select a range of rows by selecting a row and then, while holding the SHIFT key, selecting another row with the mouse. Cell selection is not supported in MCLBs with Extended selection. A mouse click can be used to deselect a row.

It applies to: List box, MCLB.

Short Help

This is the help (ASCII) text that is not displayed in a secondary window. It is limited to 30,000 characters. This Help format is displayed as a tooltip, requiring that the user hover the mouse over the control, for Java applications or in the status bar for C applications. Like Help, this help content is ASCII text and preserved within the AppBuilder repository.

Available setting options are: (text field).

It applies to: Checkbox, Combo box, Edit field, List box, MCLB, Multiline edit, Pushbutton, Radio button, Tab control, Tab page, Window.

Status Field

This is the HpsID of an Edit Field in the window that shows the Short Help for the current control under the mouse cursor.

It applies to: Window.

Stretch to Fit

A bitmap can be stretched to fit the size of a Push Button if the bitmap is too small. This is only for Java runtime behavior. When no datalink is set on a Pushbutton, Stretch to Fit is disabled.

Available setting options are: True, False.

It applies to: Pushbutton.

Style

This is the Style of a Combo box.

Available setting options are: DROPDOWN, DROPDOWNLIST, SIMPLE.

It applies to: Combo box.

System Menu

This determines whether the window displays a system menu in the upper left corner.

⚠ The system menu still won't work without the required code in the rule controlling the window.

Available setting options are: True, False.

It applies to: Window.

Tab

This determines whether the object is accessible by pressing the Tab key.

Available setting options are: True, False.

It applies to: Checkbox, Combo box, Edit field, List box, MCLB, Multiline edit, Pushbutton, Radio button, Tab control.

Tab Style

Tab style is used to set the width of tab labels in a Tab Control. AutoWidth will fit each tab according to it's label's contents. FixWidth will adjust width such that each tab page's label account for an equal percentage of the tab control's total width. HTML will adjust each tab page's label width to match the greatest value specified by any member tab page's Label Width property.

Available setting options are: Autowidth, Fixwidth, HTML.

It applies to: Tab control.

Text

This is the text to be displayed on the object. It defaults to the name of the object type generally; a pushbutton's text field defaults to &Push, for example.

Available setting options are: (text).

It applies to: Checkbox, Group Box, Pushbutton, Radio button, Static text, Tab page, Window.

Title Bar

This determines whether the window displays a title bar. If the window displays a title bar, the window's Text can display as the window title.

Available setting options are: True, False.

It applies to: Window.

Vertical Scroll Bar

This specifies whether the object displays a vertical scroll bar. Scroll bars are not supported in Java.

Available setting options are: Show_Always, Show_Never.

It applies to: Window.

Visible

This specifies whether the objects is visible to the end user. Visible is selected by default.

Available setting options are: True, False.

It applies to: Checkbox, Combo box, Edit field, Group box, List box, MCLB, Multiline edit, Pushbutton, Radio button, Static text, Tab control.

Width

This is the Integer value for the width of the object. The unit used is based on the (containing) window's Coordinate Type value.

Available setting options are: (integer value).

It applies to: Bitmap, Chart, Checkbox, Combo box, Edit field, Ellipse, Group box, Hot spot, List box, MCLB, Multiline edit, Pushbutton, Radio button, Rectangle, Static text, Tab control, Window.

Word Wrap

When Word Wrap is selected, the text displayed in a multiline edit object wraps to a new line automatically, determined by the current width of the object. Otherwise the text wraps at hard-coded new lines only. To display hidden text, the end user must scroll the text horizontally. By default, Word Wrap is not selected.

Available setting options are: True, False.

It applies to: Multiline edit.

Window Painter Objects and Properties

Window Painter Objects and Properties provides a spreadsheet illustrating all of the window objects and attributes.

Window Painter Objects and Properties

Property	Bitmap	Chart	Checkbox	Combo Box	Edit Field	Ellipse	Group Box	Hot Spot	List Box	MCLB	Multiline Edit	Pushbutton	Radio Button	Rectangle	Static Text	Tab Control	Tab Page	Window
3D					X				X		X							X
AutoCall										X								
Auto Select									X	X								
Auto Tab					X													
Background Color			X	X	X	X			X	X	X		X	X	X	X	X	X
Border Type																		X
Bottom	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
Chart Type		X																
Check Mandatory Field												X						
Close Text																		X
Coordinate Type																		X
Country																		X
Domain				X	X													
Editable					X						X							
Enabled					X						X	X						
Enter Key																		X
File Editor											X							
Font			X	X	X		X		X	X	X	X	X		X	X	X	
Foreground Color			X	X	X		X		X	X	X		X		X		X	
Form Fit					X													
Group			X	X	X				X	X	X	X	X			X		
Header Background Color										X								

Header Font										X								
Header Foreground Color										X								
Height	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X		X
Help			X	X	X				X	X	X	X	X			X		X
Horizontal Scroll Bar																		X
HpsID	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
Icon																		X
Ignore Validation								X				X						
Immediate Return			X	X	X				X		X		X					
Justification															X		X	
Label Height																	X	
Label Image																	X	
Label Width																	X	
Left		X	X	X	X	X	X	X	X	X	X	X	X	X	X	X		X
Line Draw										X								
Link	X	X	X	X	X				X	X	X	X	X					X
Mandatory				X	X						X							
Maximize Box		X																X
Minimize Box		X																X
Multiple Rows																X		
Numbering										X								
Orientation																X		
Password					X													
Resize					X										X			
Row Select										X								
Scroll Lock										X								
Selection Type									X	X								
Short Help			X	X	X				X	X	X	X	X			X	X	X
Status Field																		X
Stretch to Fit												X						
Style				X														
System Menu																		X
Tab			X	X	X				X	X	X	X	X					
Tab Style																X		
Text			X				X					X	X		X		X	X
Title Bar																		X
Vertical Scroll Bars																		X
Visible			X	X	X		X		X	X	X	X	X		X			
Width	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X		X

Word Wrap											X							

HTML Generation

When you use the Window Painter to create windows in the Construction Workbench, you can generate HTML forms from them and edit them using an HTML editor if you are preparing for thin client. Within AppBuilder, you can choose your favorite third-party HTML editor and edit the HTML interface for your application. This HTML editor installed on your development workstation can be used from AppBuilder to edit the window HTML pages, cascading style sheet (CSS), and JavaScript elements. With AppBuilder, you can modify the window or the HTML directly. When the application is prepared, AppBuilder generates the HTML (if there is none yet generated) or uses the HTML you have created or modified. Topics related to the HTML generation include:

- Working with HTML Files
- Choosing the HTML Editor
- Generating and Viewing HTML from a Window
- Modifying and Regenerating HTML
- Updating HTML Attributes
- Handling Changes to HTML
- Importing and Exporting HTML
- Deleting an HTML File
- Understanding Execution Support
- Using Additional HTML Resources

AppBuilder does not provide Print or Print Preview for HTML windows developed in the Construction Workbench. Since you can use a third-party HTML editor for HTML editing with the Construction Workbench, use that HTML editing tool for print previewing and printing.

Working with HTML Files

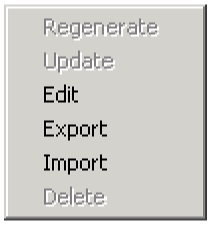
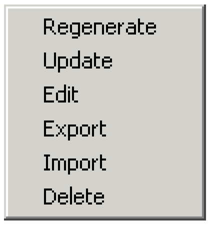
In addition to using the Window Painter to develop the forms or graphical user interface (GUI), AppBuilder can work with an HTML editor (or standard text editor) to create an HTML interface. Both the Window Painter and the HTML editor (or viewer if you have chosen not to select an editor) open separate tabs in the Construction Workbench and offer different tools in the Tools menu.

From the Window Painter or from the window object in the hierarchy, you can generate HTML for a window. AppBuilder generates the window's objects as HTML **<FORM>** elements. The attributes of the object, including font and position, are converted to a cascading style sheet (CSS). The formatting and validation rules are converted to JavaScript elements.

Each time the system loads the HTML editor (that is, through the **Edit**, **Regenerate** or **Update** functions), the system opens an HTML View tab within the Construction Workbench. The HTML View displays the HTML window as it is displayed in Internet Explorer. The system automatically updates the HTML View tab each time you save the HTML file in the HTML editor.

To access the HTML menu, shown in HTML menus, go to the **Tools** menu in the Construction Workbench or right-click the window object in the hierarchy and select **HTML**. This example assumes that an HTML editor has been selected. Otherwise the **Browse** choice appears in place of **Edit**.

HTML menus

When HTML has not yet been generated and HTML editor has been selected	When HTML has been generated and HTML editor has been selected
	

See Using Additional HTML Resources for information on using additional HTML presentation components.

There are many ways to work with HTML generation from a window in AppBuilder. It depends on whether you use the results of HTML generation "as is" or provide your own design for already generated HTML.

You do not have to keep the results of HTML generation in a repository. It is generated every time you need it: when you want to see it in the Construction Workbench or when you prepare a window. If you do not need the HTML version of the window to make changes in it, it is strongly recommended not to commit this HTML window to the repository.

Alternatively, AppBuilder provides certain possibilities for you to work with the HTML tab if you wish to store the HTML in the repository, but you have to understand how AppBuilder works with the HTML.

1. Create the HTML page. You can make changes in the HTML window. You must commit the changes to the repository.
2. You can extract your HTML window from the repository and edit it. You must commit these changes to the repository again.
3. The preparation tool in AppBuilder uses the HTML window that you put into the repository if there is one or creates a new one if there is no HTML window kept in the repository.
4. If you made changes to objects in the repository, such as creating new objects on your AppBuilder window or deleting them, and want to apply these changes to your HTML window, you can update the HTML (**HTML > Update**). This procedure synchronizes AppBuilder objects and corresponding HTML objects without changing their features (colors, font, and so on).
5. In certain cases, a new version of AppBuilder might not be completely backward compatible with old ones. If so, an upgrade procedure must be applied if you want to move your HTML window into the latest version of AppBuilder.

When you prepare a window, if you want to save the HTML in the repository, you must commit your changes.

Choosing the HTML Editor

If you are editing the HTML, AppBuilder by default uses NotePad as the text editor. To select another HTML editor, select **Tools > Workbench Options**, then click **Prepare > HTML Generation**. See HTML Generation Options for more information.

If you do not have an editor selected, the HTML menu shows the choice as **Browse** because you can only view the HTML that is generated and preview results of the HTML generation in the Internet Explorer window. If you have an editor selected, then the menu shows the choice as **Edit** (see HTML menus).

AppBuilder generates the objects of a window as HTML FORM elements. The attributes, including font and position, of an object are converted to cascading style sheets (CSS). The formatting and validation rules are converted to JavaScript elements.

Each time the system loads the HTML editor (that is, through the **Browse**, **Regenerate**, or **Update** functions), the system opens an HTML viewer within the Workbench. The HTML viewer displays the HTML window as it is displayed in Internet Explorer. The system automatically updates the HTML View tab each time you save the HTML file in the HTML editor.

To revert changes made during the session, close the HTML editor and select **File > Rollback**. If the HTML editor is still open, the system displays a warning message and automatically closes the HTML View window.

Using an HTML Editor

Use the **Call Editor** option from the Construction Workbench dialog to have AppBuilder call a particular HTML editor, so that you can either edit the generated HTML or view the generated HTML in an editor of your choice. By default, the **Call Editor** option is unchecked, and AppBuilder does not call an HTML editor; so when you generate HTML files for windows, you can view, but not edit the HTML. If you do not intend to modify the generated HTML, leave this option unchecked. When the HTML menu items are invoked, AppBuilder opens an HTML viewer in the Work Area of the Construction Workbench with the specific HTML file.

When the **Call Editor** option is checked, the HTML viewer still displays and acts as a state window between the Workbench and the external editor. If this state window is closed, the link between the editor and Construction Workbench is broken and changes to the HTML are lost. The **Editor File Name** field becomes active. Browse to the directory on your machine to select the HTML editor executable file. By default, the editor is Windows NotePad (notepad.exe). If the **Call Editor** option is checked and the **Editor File Name** field is empty or has an incorrect executable specified, you receive an error window when trying to access HTML functions from the Construction Workbench.

When using an external editor, the editor is loaded with either the generated HTML or a version of the HTML that has been extracted from the repository. Before regenerating the HTML from the Construction Workbench, close the external HTML editor. If the external editor is still active during a regenerate, a failure error might result because the external editor has locked the file for editing. Or, it may allow the update of the contents of the file, but it might require a refresh from the editor to load the new changes. These behaviors are editor specific; check the editor documentation for specific details.

Template for HTML Page

To give your generated HTML pages a standard look and feel that might include headers, footers, a company logo, etc., use a template page with these elements. AppBuilder provides a standard HTML template that you can customize. Select the template file from this field. Use the browse button to locate the template file. When the HTML template is specified, the HTML generator uses the template as the starting point and appends the generated HTML FORM into this template so pages have a uniform look and feel.

Export/Import Location

Specify the location, directory path, for imports and exports of HTML. Type the path or click the browse button to select the path.

Temporary HTML Location

Specify the location, directory path, for temporary HTML when generating HTML. Type the path or click the browse button to select the path. There are additional HPS.ini settings that you must manually update in your hps.ini file. There is not an option to update the hps.ini file from the Construction Workbench. These settings are as follows:

- THINCLIENT_ERROR_HTML - Use this setting to customize your HTML error page. You must specify a full path statement without any spaces.

- THINCLIENT_START_HTML - Use this setting to customize your first HTML page. You must specify a full path statement without any spaces. After preparation, this HTML file is renamed with the *rule name*.html.

Generating and Viewing HTML from a Window

To create a new HTML page from an AppBuilder window, complete the following steps:

1. Use the Window Painter to create the window.
2. Select **Tools > HTML > Edit** (if you have an editor selected) or **Tools > HTML > Browse** (if you do not have an editor selected) to convert the window to HTML. This command generates the new HTML code for the window and opens the editor or viewer with the HTML.

   ✓ You can also right-click the window object in the project hierarchy and select **HTML > Edit** from the pop-up menu.

   AppBuilder converts the window to HTML. From the **Analysis** tab of the Output window, you can see the status of the conversion. After converting the window, AppBuilder displays an HTML version of the window and the HTML code in the specified HTML editor or viewer.

You can use a third-party HTML editor to modify any of the generated contents (HTML, CSS, JavaScript, etc.) or add and delete the contents of the window apart from the Window Painter.

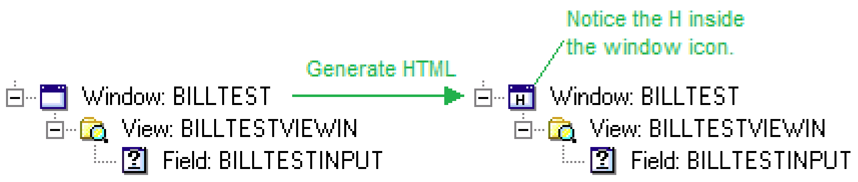
After saving the HTML file from an external editor, you must perform a commit or rollback from the Construction Workbench to put the HTML in the repository:

1. Save the HTML file and close the editor.
2. In Construction Workbench, select **File > Commit** to store both the GUI workstation panel and its corresponding HTML, CSS, and JavaScript elements in the repository.

The HTML files and window are independent objects. If you make changes to the window in the Window Painter, the system does not automatically update the HTML window. See Modifying and Regenerating HTML for information on maintaining both windows.

When the HTML code for a window has been committed to the repository, the icon in the hierarchy changes, as shown in HTML indicator in window icon.

HTML indicator in window icon



There are several ways to create an HTML window. Some of them are implicit, some are explicit, and you must understand what happens every time you deal with your HTML window.

1. You may select **Export** to export your HTML window from the repository. If there is no HTML window at this point, one is generated.
2. You may select **Regenerate**. In this case a new HTML file is generated even if you already had an HTML window for this AppBuilder window stored in the repository.
3. You may select **Edit**. As in the first case, it shows you the HTML that is stored in the repository. If there is no HTML, one is generated.
4. You may run preparation. If there is no HTML associated with your AppBuilder window, it is generated.

Creating an HTML window does not mean that you automatically store it in the repository. To store an HTML window in the repository, you must commit your changes to the repository. Generating or preparing does not automatically store the HTML in the repository. Also, if you set your Workbench Options in the HTML Editor tab so that you do not have an HTML editor, you cannot change the HTML window in AppBuilder nor store it in the repository.

Modifying and Regenerating HTML

Modifying and regenerating HTML involves the following:

- Updating or Regenerating
- Seeing Results of Update or Regenerate
- Committing or Rolling Back the Changes

Updating or Regenerating

You can use a third-party HTML editor to modify any of the generated contents (HTML, CSS, JavaScript, etc.) or add and delete the contents of the window apart from the Window Painter. Because the Window Painter and HTML windows are independent files, changing one might cause the other to become out of sync. AppBuilder provides these functions to ensure that both versions of windows (Window Painter and HTML) are identical:

- **Update** – Incorporates (in the existing HTML file) any additions or deletions of objects to the window made in Window Painter. Use this function to update the HTML version of the window without losing any custom HTML modifications made in an editor outside of AppBuilder. This merges the changes done in the Window Painter with the changes done to the HTML in an editor. Any modifications (for example, property changes) are ignored.
- **Regenerate** – Creates a new HTML file based on the current window in Window Painter. This function completely overwrites the existing HTML file.

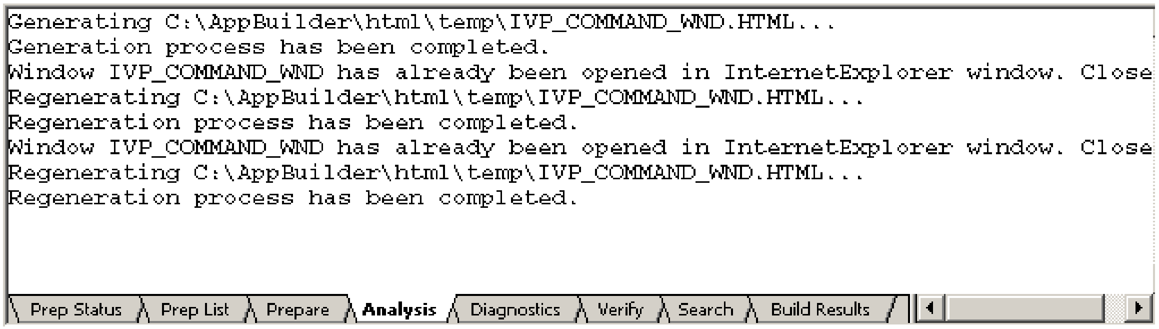
For a full explanation of the procedure of updating the HTML, refer to Updating HTML Attributes.

For information about any of the messages, refer to the *Messages Reference Guide*.

Seeing Results of Update or Regenerate

After using one of the HTML options, **Update** or **Regenerate**, the results are displayed in the Output window in the Analysis tab. For example, when the HTML is updated, the Analysis tab, as shown in Results of HTML Update, lists the HTML attributes that are updated. For a regeneration, the Analysis tab simply shows a single statement that a regeneration was completed. When you update an HTML window, all graphical user interface (GUI) information, including the menu, is regenerated.

Results of HTML Update



```
Generating C:\AppBuilder\html\temp\IVP_COMMAND_WND.HTML...
Generation process has been completed.
Window IVP_COMMAND_WND has already been opened in InternetExplorer window. Close
Regenerating C:\AppBuilder\html\temp\IVP_COMMAND_WND.HTML...
Regeneration process has been completed.
Window IVP_COMMAND_WND has already been opened in InternetExplorer window. Close
Regenerating C:\AppBuilder\html\temp\IVP_COMMAND_WND.HTML...
Regeneration process has been completed.
```

An initially generated HTML file is relatively easy to update: you only have to regenerate it again. Whether there is a change in the generation model or not, it does not matter. A newly regenerated file is is acceptable.

The temporary directory that AppBuilder uses to place the generated HTML file can be modified. Refer to the Matrix Builder Options in Workbench Options.

Committing or Rolling Back the Changes

After using one of the HTML options, **Update** or **Regenerate**, AppBuilder loads the HTML viewer or editor to allow you to verify or further modify the HTML file. After saving the HTML file, use the **Commit** function to ensure that the Window Painter windows and HTML windows are identical. AppBuilder automatically updates the HTML View tab to reflect the updated or regenerated HTML file.

If you perform a **Rollback**, the system "rolls back" (that is, reverts) all the changes made during the session. If the HTML editor is still open, the system displays a warning message and automatically closes the HTML View tab. You must close the HTML editor.

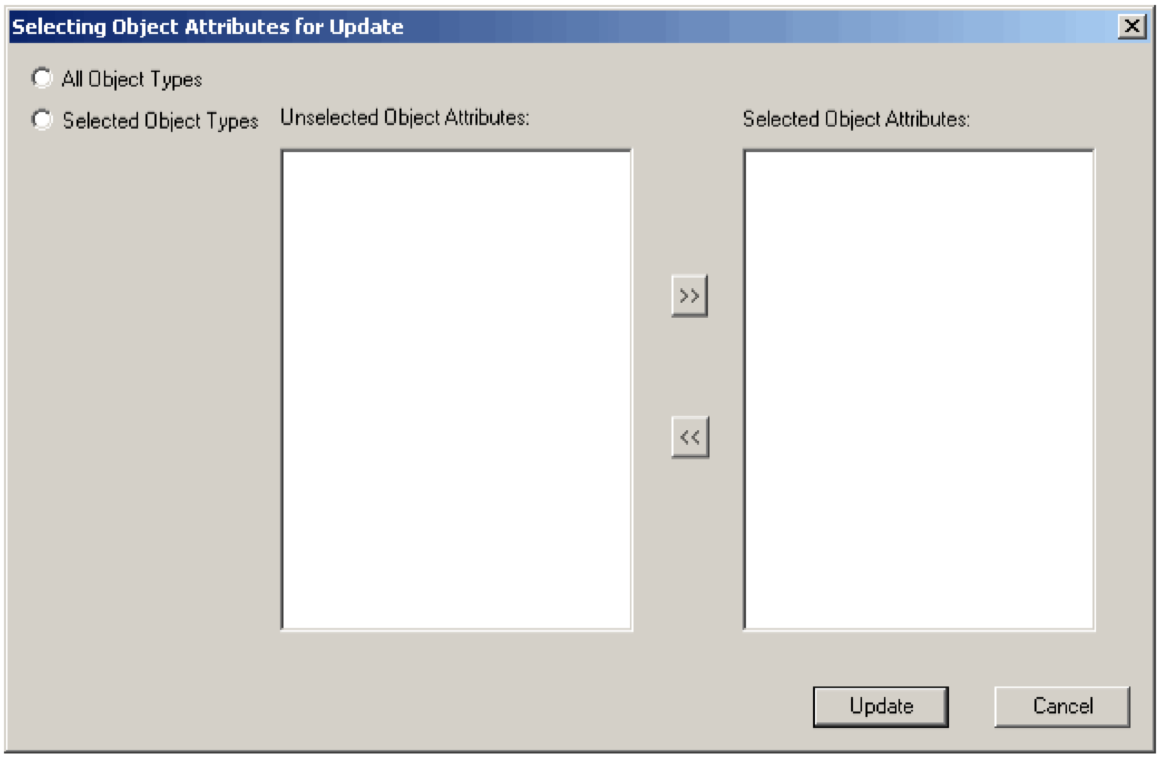
Once you decide to change your HTML window and to keep your changes, then the process is a little more involved. You have to align the changes of the window in the repository with any changes in the HTML. To keep the copies aligned (or in sync), update your HTML window every time you need it using the **Update** function. Also, in some cases you must apply the upgrade procedure if you want to have a new version of generated HTML. Refer to Handling Changes to HTML.

Updating HTML Attributes

To update the HTML file, complete the following steps:

1. Select the window object in the hierarchy and right-click and select **HTML > Update** or from the Window Painter of the Window to update, select **Tools > HTML > Update**. The **Select Object Attributes for Update** dialog displays (empty, see Empty Select Object Attributes for Update dialog).

   Empty Select Object Attributes for Update dialog


Selecting Object Attributes for Update
All Object Types
Selected Object Types
Unselected Object Attributes:
Selected Object Attributes:
>>
<<
Update
Cancel

2. Select the object types in the window whose attributes are to be updated in the HTML file.
   - **All Object Types** selects all the object types in the window (see Updating All Object Types but only selected attributes)
   - **Selected Object Types** enables you to select a subset of the object types (see Updating Selected Object Types and only selected attributes).

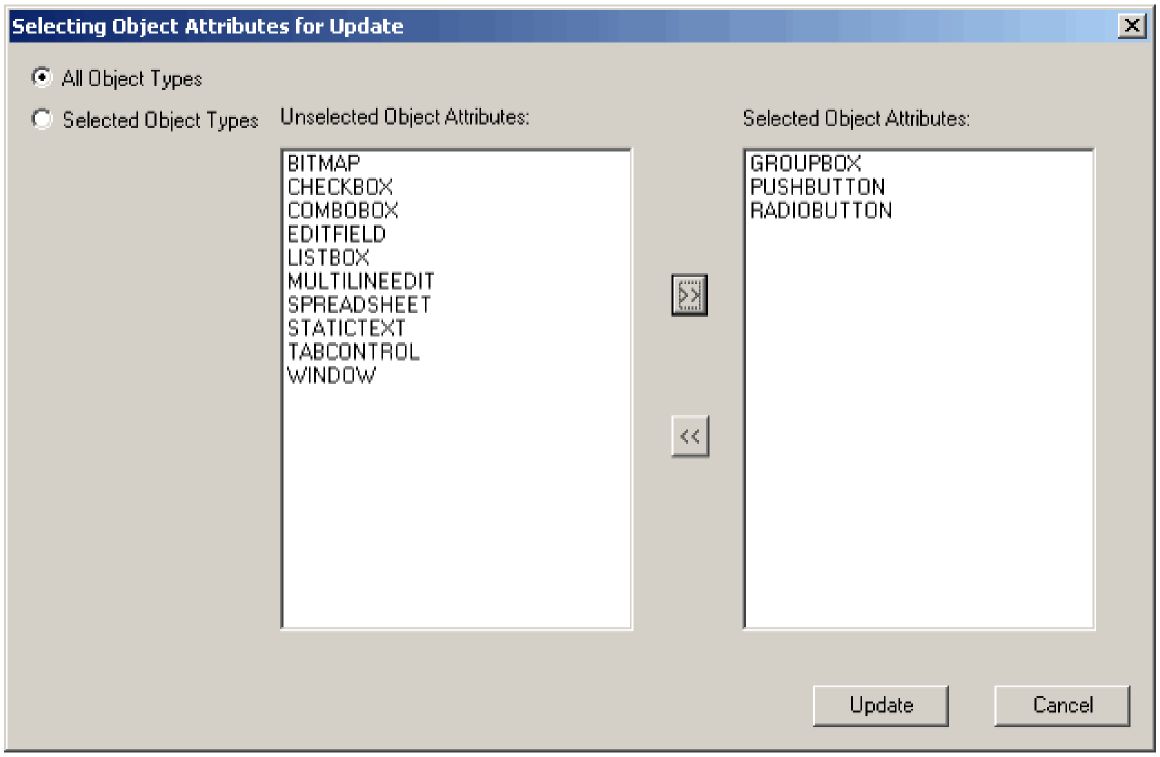
   The examples in Updating All Object Types but only selected attributes and Updating Selected Object Types and only selected attributes show the different methods of selecting object types. Refer to Objects and attributes for HTML update for a list of the objects and attributes that can be updated. Notice that this is a subset of all the attributes of all the objects. Only these are supported for updating.

Objects and attributes for HTML update

Object	Selectable Attributes
Bitmap	Link
Check Box	Visibility, Immediate-Return
Combo Box	Visibility
Edit Field	Visibility, Read-Only, Protected, Password, Format, Immediate-Return, Mandatory
Group Box	Visibility
List Box	Visibility, SelectionType
MCLB (Spreadsheet or Table)	Visibility, SelectionType, DrawLines, AutoSelect
Multiline Edit Field	Visibility, WordWrap, Editable
Push Button	Visibility, CheckMandatoryFields, IgnoreValidation, Link
Radio Button	Visibility, Immediate-Return, Link
Static Text	Visibility
Window	Visibility
Tab Control	MultipleRows, Orientation, Tab_Style, Visibility

The **Link** attribute is only supported as selectable for some of the selectable objects that have the Link attribute. Bitmap, Push Button, and Radio Button are the only objects for which this is a selectable attribute.

Updating All Object Types but only selected attributes


Selecting Object Attributes for Update

(● All Object Types) ○ Selected Object Types

Unselected Object Attributes:

BITMAP
CHECKBOX
COMBOBOX
EDITFIELD
LISTBOX
MULTILINEEDIT
SPREADSHEET
STATICTEXT
TABCONTROL
WINDOW

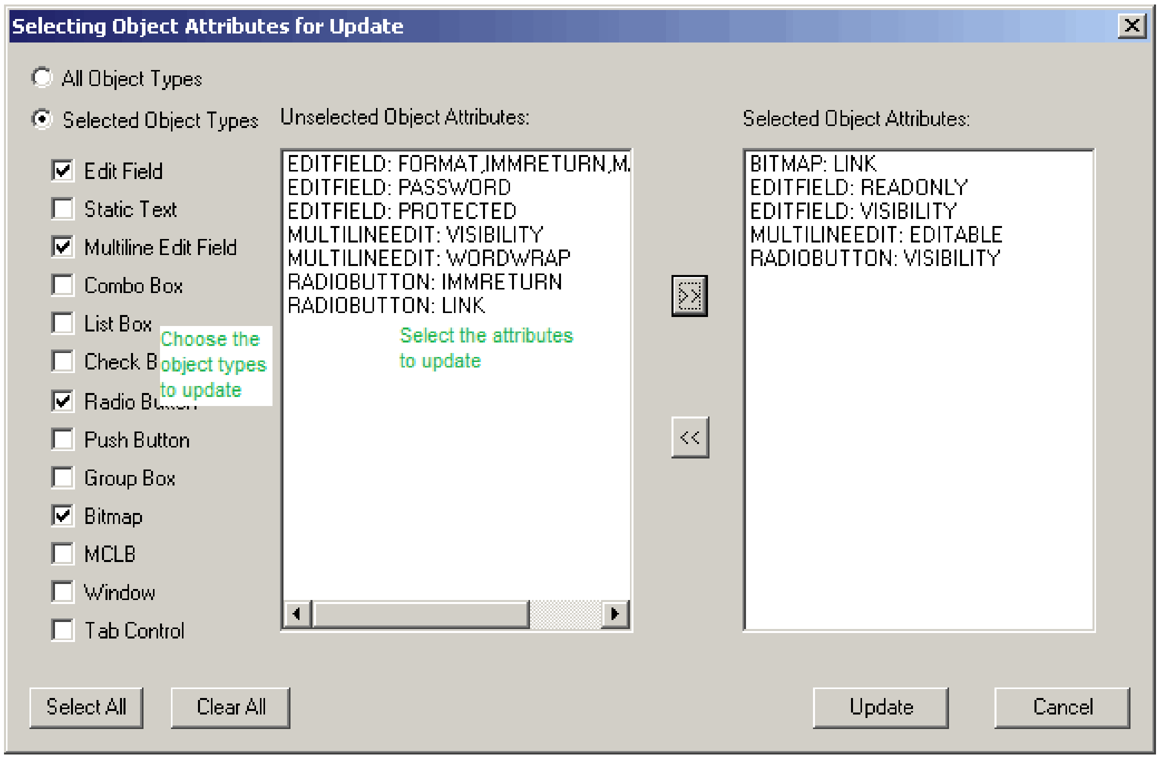
Selected Object Attributes:

GROUPBOX
PUSHBUTTON
RADIOBUTTON

With All Types selected, the attributes for them appear in the list. Select the attributes to update and click the **>>** to move them to the Selected list. Click **Update** when done.

3. If you want to select only certain object types, choose **Selected Object Types** and a list of check boxes is displayed that allows you select (or de-select) object types.

Updating Selected Object Types and only selected attributes


Selecting Object Attributes for Update

○ All Object Types (● Selected Object Types)

- [x] Edit Field
- [ ] Static Text
- [x] Multiline Edit Field
- [ ] Combo Box
- [ ] List Box
- [ ] Check Box
- [x] Radio Button
- [ ] Push Button
- [ ] Group Box
- [x] Bitmap
- [ ] MCLB
- [ ] Window
- [ ] Tab Control

Unselected Object Attributes:

EDITFIELD: FORMAT,IMMRETURN,M.
EDITFIELD: PASSWORD
EDITFIELD: PROTECTED
MULTILINEEDIT: VISIBILITY
MULTILINEEDIT: WORDWRAP
RADIOBUTTON: IMMRETURN
RADIOBUTTON: LINK

Selected Object Attributes:

BITMAP: LINK
EDITFIELD: READONLY
EDITFIELD: VISIBILITY
MULTILINEEDIT: EDITABLE
RADIOBUTTON: VISIBILITY

Choose the object types to update
Select the attributes to update
>>
<<
Select All
Clear All
Update
Cancel

4. When you have selected the object types, the attributes for those types appear in the **Unselected Object Attributes** list (see Updating Selected Object Types and only selected attributes). Select the attributes that you want to update by moving them from the **Unselected Attributes** list to the **Selected Attributes** list.
5. When the selected list of attributes to update is complete, click **Update**. The HTML Editor and HTML View are launched with the results

of the updated HTML file. Also, the Analysis tab in the Output window lists the results. Refer to Results of HTML Update.

Handling Changes to HTML

You can edit the HTML files outside of AppBuilder. Problems may arise if changes are made to the HTML file apart from the changes of the window in the Window Painter, if you want to retain these changes in one place. In order to solve the problem of updating the generated HTML with changes to the window in Window Painter, AppBuilder keeps a special comment section in the HTML file. This comment section contains information about the AppBuilder objects that are included in the most recent generation. For example, the comment section might look like this:

```
<!-- HPS IDs
FIELDSET#group_1:GROUPBOX
FIELDSET#text_group_1:GROUPTEXT
INPUT#check_1:CHECKBOX
INPUT#text_check_1:CHECKTEXT
TEXTAREA#mle_1:MULTILINEEDIT
SELECT#combo_1:COMBOBOX
IMG#bit_1:BITMAP
INPUT#push_1:PUSHBUTTON
SELECT#list_1:LISTBOX
LABEL#static_1:STATICTEXT
-->
```

This section is placed as the first comment section in the FORM element. You must *not* change or move this comment section that contains specific repository information. Doing so would cause an error message to be displayed and the generation process to be stopped. As a last step of every generation, AppBuilder creates this section in the HTML file that contains information about AppBuilder objects.

✓ If you do a backspace in some HTML editors, such as Microsoft FrontPage, you may accidentally delete the comment section that contains the system identifiers (IDs). Backspacing may delete an entire object. Special care should be taken when using such HTML editors, because if this comment section is deleted, you will be unable to update the HTML from the Construction Workbench.

Each time you do an update, AppBuilder parses this section to see what it should delete and create. First, it deletes the objects from the HTML that are not in the initial AppBuilder window anymore. Then, it creates the objects that are not in the original HTML but have been added to the window. For AppBuilder objects that already are included in the old HTML file, AppBuilder regenerates the necessary attributes, such as ID and Name. You should not change these; every updating process restores them from the repository. By performing an update, AppBuilder creates the HTML for all the objects that are in the window object in the repository and puts that HTML in the HTML file.

Importing and Exporting HTML

AppBuilder also contains functions to import and export HTML windows.

- **Export** – Exports one or more Window Painter windows to a specified target directory. If you have not previously created an HTML window version of the Window Painter window, the system automatically generates the necessary HTML files. The name of the HTML file is identical with the long name of the Window Painter window.
- **Import** – Imports one or more HTML windows into the repository. The name of the HTML file must match the long name of the Window Painter window.

The directory that AppBuilder uses for exporting and importing can be modified. Refer to the HTML Generation Options.

These functions assume that AppBuilder has a window in the repository from which it can do HTML generation and export. The exported HTML can be altered in a preferred HTML editor, and then re-imported, but you should have a window in the hierarchy before starting in HTML. The Import function is intended for importing previously exported HTML, which was generated from a window. If you try to import HTML without having an initial AppBuilder window with some contents, this does not work.

Deleting an HTML File

The HTML menu provides the **Delete** menu choice. The **Delete** menu choice Deletes the current HTML file from the repository. You can generate a new HTML file using the instructions in Using an HTML Editor. It is not saved in the repository until you commit the changes. It is only made available to be committed when it is first saved from the external editor. If there is no HTML file in the repository (corresponding to a window) during a prepare for thin-client windows, a new HTML file is generated from the current window pane for deployment.

Understanding Execution Support

This includes understanding:

- Execution Agreements

Execution Agreements

AppBuilder generates HTML code that works with Java and JavaScript execution (runtime) support. There are some guidelines that are set between generation and execution. Failure to follow these guidelines may cause problems at execution. You must understand the generated code and the agreements between generation and execution. Common guidelines include:

1. Elements have the style attribute "position: absolute". AppBuilder uses this attribute to position the object on the page. Changing this attribute changes the positioning, so we recommend not changing this attribute.
2. The other parts of the style attribute of an HTML element contains visual information (colors, borders, etc.) about the object. It is usually safe to change this information.
3. The ID attribute is used for the update procedure (in HTML generation) and in the JavaScript execution for MCLBs. Changing it may interfere with the updating process and cause severe problems for MCLBs.
4. The name attribute is used in Java execution for getting data after a form is submitted. Changing the name attribute makes an object invisible for execution.
5. The class attribute contains common features like borders, colors, etc. Changing this attribute changes the object's appearance. The user should not change this attribute for Menu and MCLBs because JavaScript execution for these objects uses this attribute.
6. Elements are generated inside the FORM element; there is only one in an HTML file. Moving the object outside the FORM tags makes it invisible for execution.
7. The first comment section inside the FORM element (labeled HPS IDS) contains information about the AppBuilder objects that are generated. It is used for the update process. Changing this section may cause problems in the update process.

Generated HTML for a Window

AppBuilder uses a special HTML template to generate an HTML file from a window. The name of this template may be specified in the HTML Generation set of options of the Workbench Options window. If no template name is set, then AppBuilder uses the default template. There is a default HTML template file that is installed with AppBuilder. The file and path that AppBuilder uses for the template can be modified. Refer to the HTML Generation Options.

The HTML template is simply an ordinary HTML file. There is only one thing to remember about this template: the BODY element has to contain a DIV element with ID=LVEL_TODO. The content of this DIV element is ignored, so it is a good idea to keep it empty. It does not have to have a style attribute position: absolute, though it is really hard to imagine a reasonable layout without this attribute.

Almost everything that AppBuilder writes goes into this DIV element. In addition, several SCRIPT elements and one LINK element (that includes several common files) is added to the HEAD of the HTML file. You do not need to include these elements because AppBuilder takes care of it.

The system generates the following HTML for a window object in AppBuilder:

```html
<HTML>
<HEAD>
<TITLE>
…
</TITLE>
<LINK HREF="./includes/styles.css" type=text/css rel=stylesheet>
<SCRIPT language=javascript src="./includes/support.js" type=text/javascript></SCRIPT>
<SCRIPT language=javascript src="./includes/menu.js" type=text/javascript></SCRIPT>
<SCRIPT language=javascript src="./includes/mclb.js" type=text/javascript></SCRIPT>
<SCRIPT language=javascript src="./includes/hpsformat.js" type=text/javascript></SCRIPT>
<SCRIPT language=javascript>
document.LVEL_Version = '…';
document.LVEL_Number = …;
</SCRIPT>
</HEAD>
<BODY>
<DIV ID=LVEL_TODO …>
<FORM id=mclb action=ABServlet method=post>
<!-- HPS IDS
HTML_Type#HPSID: HPS_Type
…
-->
</FORM>
</DIV>
</BODY>
</HTML>
```

Window HTML Details

In the HEAD section there are several includes:

- styles.css that contains all styles defined for AppBuilder objects
- support.js, menu.js, mclb.js, hpsformat.js – for runtime support
- extension.js – for runtime support and user runtime support
- SCRIPT section that contains version information

Do not change any of these includes, except the last one. The last file can contain some user changes. Do not change the SCRIPT section that contains version information. It is used by the Upgrade procedure. You can change the TITLE section.

The FORM element that is generated by HTML generator has to be the only one inside the DIV element with id=LVEL_TODO. Parameters of the FORM element should not be changed. There may be other FORM elements in this HTML file, but all of them outside of <DIV id=LVEL_TODO>.

Inside the FORM element there is a comment section (the first element inside the form). This section contains information about AppBuilder objects for update procedure. You can *not* change it.

Generated HTML for a Menu

The system generates the following HTML for a menu object in AppBuilder:

```
<DIV id=lvel_menubar_div style="border: none; width: 100%; position: absolute; top: 1px; height: 21px;
z-index: 1001;background: transparent">
<SCRIPT language=javascript type=text/javascript>
LVEL_MNUItem(0, 'LVEL_MenuBar', '', '')
LVEL_MNUItem(1, 'hpsid1', 'item1', '0000')
LVEL_MNUItem(2, 'hpsid2', 'item2', '0000')
LVEL_MNUItem(1, 'hpsid3', item3', '0000')
LVEL_MNUInitialize()
</SCRIPT>
</DIV>
```

Menu HTML Details

The DIV element above should not be changed. Menu generation and runtime support is a very complex process. The SCRIPT element contains set of calls that describe Menu structure. You can change it if you want to change a menu structure. LVEL_MNUItem procedure: this procedure is called every time one more menu item is needed.

The first parameter indicates the level of the menu item (closed element above with the level equal current level minus one is an owner of the current element). The second parameter is the system identifier (HPSID); the third one is the menu item text as it is displayed on the screen. The last parameter contains a scale of bits where each bit has its own meaning:

1. checked menu item
2. disabled menu item
3. check mandatory fields when this item is pressed
4. ignore validation when this item is pressed

Menu Forms

Two types of menus are supported by the HTML thin client. The first one is the standard pull-down menu, as close to the standard as is possible for HTML. The second one shows a menu at the left in the form of a tree. The tree form is as close as possible to the standard.

A user can open and close tree nodes without submitting the page to the server. The rest of the form goes into the special <DIV id=LVEL_Right> element. When the menu-tree gets too wide it moves this DIV to the right and when it shrinks it moves the DIV back.

Using Additional HTML Resources

Your application may require additional resource files (such as multimedia files and graphics) as part of the HTML pages. These can be stored in a Component Folder object but must be put in a particular place in the object hierarchy. These additional resources might include global resource files common to many or all of the Web pages (such as a cascading style sheet, META tags, etc.). Store these files as part of a Component Folder defined as a child of the process object in the hierarchy. These additional resources might also include Web-page-specific resource files, unique to specific pages. Store these files as part of a Component Folder defined as a child of the window object in the hierarchy.

The resource files should be defined with the Folder Type as **HPS_WEB_RESOURCE** . Specify the Folder Type on the **General** section of the Object Property window. Component folder entries with the given Folder Type are prepared with the Partition or Window object and are copied to the *images* subdirectory.

Take the following steps to work with resource files in the Component Folder:

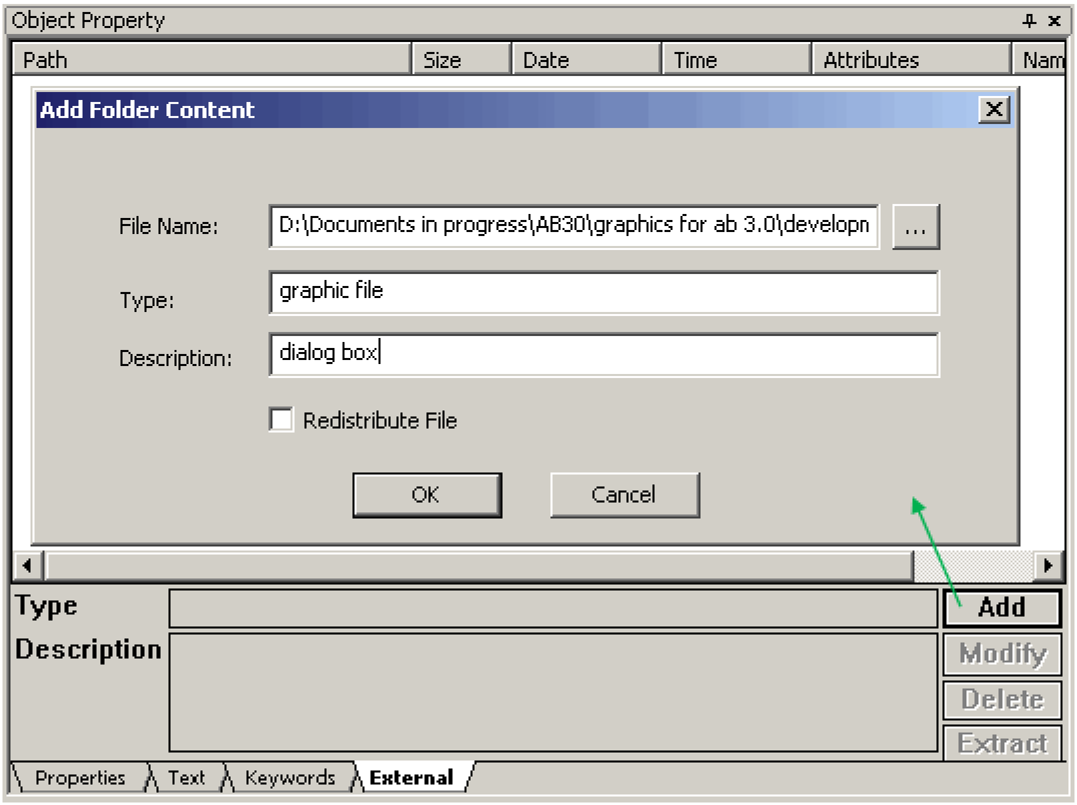
1. Insert the Component Folder object into the correct location in the hierarchy.
2. Display the Object Property window for the Component Folder object, if it is not already displayed.
3. Click the **External** tab. You can perform the following tasks:
   - Adding Files to the Component Folder
   - Removing Files from the Component Folder
   - Modifying Files from the Component Folder
   - Extracting Files from the Repository

Adding Files to the Component Folder

To add files to the Component Folder:

1. Click **Add** from the External tab of the Object Property window.
   The Add Folder Contents window displays (see Add Folder Contents dialog sample).

Add Folder Contents dialog sample



File Name: D:\Documents in progress\AB30\graphics for ab 3.0\developn

Type: graphic file

Description: dialog box

Redistribute File

2. Click the browse button (**...**) next to the File Name field, to locate the file you want to include.
3. Enter a description of the file type in the Type field.
4. Enter a text description of the file in the Description field.
5. To distribute this file at runtime, check the **Redistribute File** check box.
6. Click **OK** to add this file to the Component Folder. This file is now added to the repository.

Removing Files from the Component Folder

Removing a file from the Component Folder removes it from the repository. To remove files from the Component Folder:

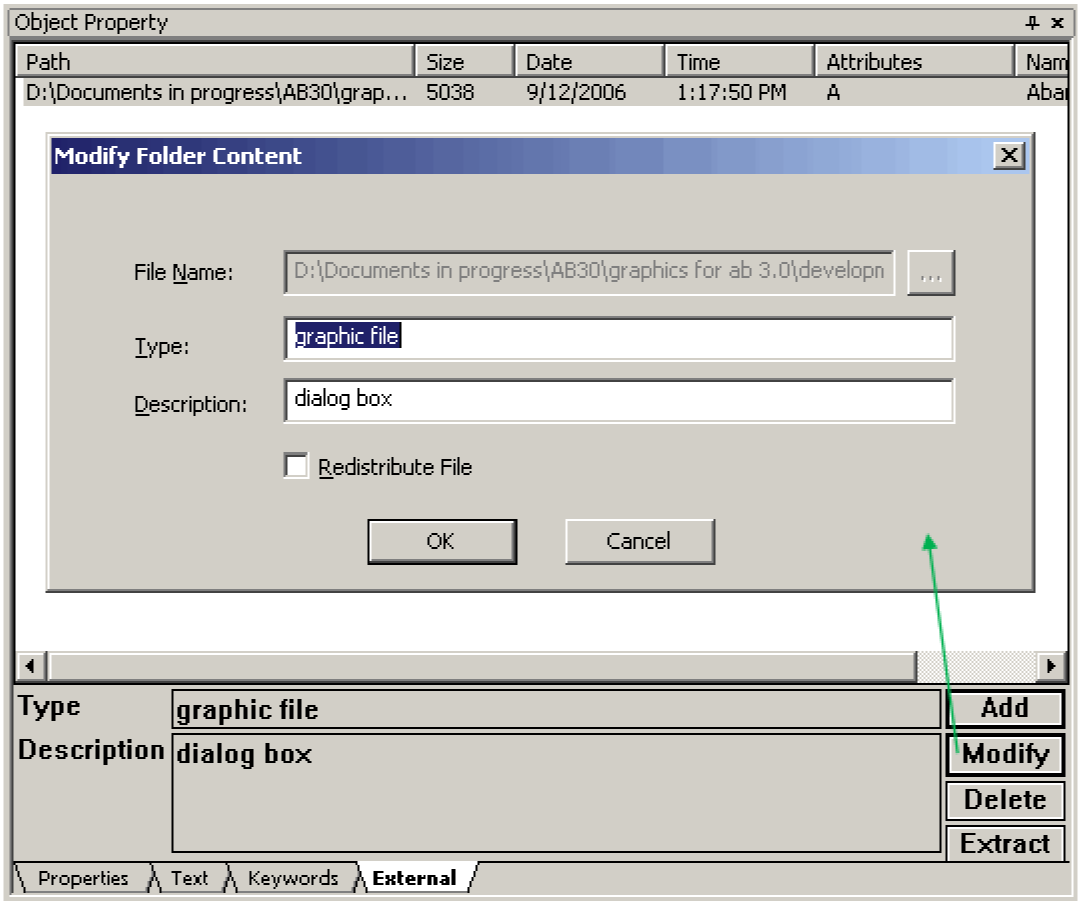
1. Highlight the file to be removed in the Folder Contents list.
2. Click **Remove** from the External tab of the Object Property window.
3. The system verifies that you want to remove this file. Click **OK**.
   The file is removed from the repository.

Modifying Files from the Component Folder

To modify the file type and description of a file in the component folder:

1. Highlight the file to be modified in the Folder Contents list.
2. Click **Modify** from the External tab of the Object Property window.
   The Modify Component Folder Content window displays.

Modify Component Folder contents


Object Property

Path	Size	Date	Time	Attributes	Nam
D:\Documents in progress\AB30\grap...	5038	9/12/2006	1:17:50 PM	A	Abai

Modify Folder Content

File Name: D:\Documents in progress\AB30\graphics for ab 3.0\developn

Type: graphic file

Description: dialog box

☐ Redistribute File

Type: graphic file

Description: dialog box

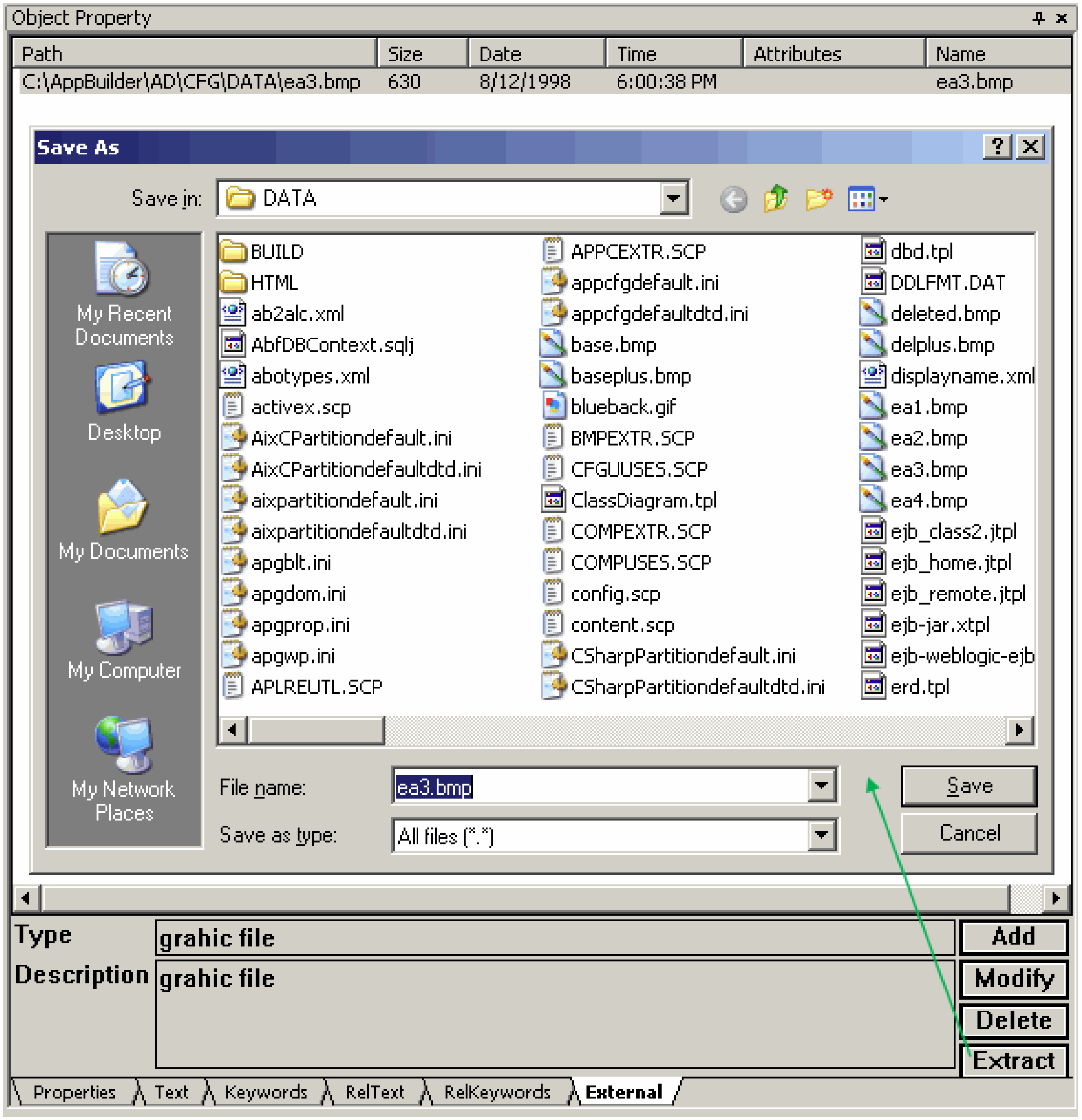
3. You can make changes to the file Type, Description, or Redistribute File setting.
4. Click **OK** to save the changes.

Extracting Files from the Repository

To extract a copy of a file in the Component Folder from the repository and save it to a new location, complete the following steps:

1. Highlight the file to be extracted in the Folder Contents list. To highlight a range of multiple files, press Shift and click the last file in the range. To highlight multiple individual files, press **Ctrl** and click the files you want to extract.
2. Click **Extract** from the External tab of the Object Property window.
   The Save As window displays.
3. In the Save In field, specify the drive and folder where you want to save the extracted file.

Extract file from Component Folder contents


Object Property

Path	Size	Date	Time	Attributes	Name
C:\AppBuilder\AD\CFG\DATA\ea3.bmp	630	8/12/1998	6:00:38 PM		ea3.bmp

Save As

Save in: DATA

BUILD	APPCEXTR.SCP	dbd.tpl
HTML	appcfgdefault.ini	DDLFMT.DAT
ab2alc.xml	appcfgdefaultdtd.ini	deleted.bmp
AbfDBContext.sqlj	base.bmp	delplus.bmp
abotypes.xml	baseplus.bmp	displayname.xml
activex.scp	blueback.gif	ea1.bmp
AixCPartitiondefault.ini	BMPEXTR.SCP	ea2.bmp
AixCPartitiondefaultdtd.ini	CFGUUSES.SCP	ea3.bmp
aixpartitiondefault.ini	ClassDiagram.tpl	ea4.bmp
aixpartitiondefaultdtd.ini	COMPEXTR.SCP	ejb_class2.jtpl
apgblt.ini	COMPUSES.SCP	ejb_home.jtpl
apgdom.ini	config.scp	ejb_remote.jtpl
apgprop.ini	content.scp	ejb-jar.xtpl
apgwp.ini	CSharpPartitiondefault.ini	ejb-weblogic-ejb
APLREUTL.SCP	CSharpPartitiondefaultdtd.ini	erd.tpl

File name: ea3.bmp

Save as type: All files (*.*)

Type: grahic file

Description: grahic file

4. You can change the file name in the File name field.
5. Click **Save** to save a copy of the file to the new location.

Edit and Display Masks

AppBuilder supports the use of edit (or input) masks, for Java clients, and display masks (or display pictures) supported on Java and C. Use edit masks to format the data in a field when the field does have focus or to restrict the input to certain characters in certain positions. Use the display masks to format the data in a field when the field does not have focus. The supported edit and display masks are listed and explained in this topic. These masks are based on the characters used in Microsoft VisualBasic.
The edit masks include:

- Alphanumeric Edit Masks
- Date and Time Edit Masks

The display masks include:

- Display Masks (Numeric)

Edit Masks

Edit masks are input masks intended to restrict or filter the characters that an end user can enter in an edit field. The end user input depends on the mask characters in that position. The valid mask characters are given in Alphanumeric mask characters. For example, with the edit mask

###\\-##
-####, the '#' is a digit place holder; it accepts only digits. The mask character '~~'- (sign mask) will be treated as a literal because it is escaped with ' '. This mask only allows the user to type digits and only at character positions 1,2,3,5,6,8,9,10 and 11. At runtime, the mask positions are filled with under scores and literals are shown. For example, the above mentioned mask is shown as __~~-__. Entering the value 123456789 is accepted as 123-45-6789.

The following types of edit masks are supported:

- Numeric edit masks (including Integer edit masks and Decimal edit masks)
- String edit masks
- Date edit masks
- Time edit masks

⚠ Edit masks are not supported in HTML.

The edit masks, for Java only, include:

- Alphanumeric Edit Masks
- Date and Time Edit Masks

Alphanumeric Edit Masks

This section describes the alphanumeric mask characters.

Alphanumeric mask characters

Mask character	Description	Input mask?	Display mask?
#	Digit placeholder.	Y	–
.	Decimal placeholder. The specified character is used as the decimal placeholder in the international settings. This character is treated as a literal.	Y	–
-	Sign placeholder. The specified character is used as the sign placeholder. This character is treated as a literal.	Y	Y
,	Thousands separator. The specified character is used as the thousands separator in the international settings. This character is treated as a literal.	Y	Y
:	Time separator. The specified character is used as the time separator in the international settings. This character is treated as a literal.	Y	Y
/	Date separator. The specified character is used as the date separator in the international settings. This character is treated as a literal.	Y	Y
	Treat the next character in the mask string as a literal. Use this character to include *#* , *&* , *A* , and *?* characters in the mask.	Y	Y
' .... '	Single quotes hold a group of literals. For example, the mask *'Today is' EEEE 'the' dd'th'* is displayed as *Today is Monday the 13th* . This character is treated as a literal.	Y	Y
&	Character placeholder. Valid values are ANSI characters *32 – 126* and *128 – 255* .	Y	–
>	Convert all the characters that follow to UPPERCASE.	Y	–
<	Convert all the characters that follow to lowercase.	Y	–
A	Alphanumeric character placeholder (entry required). For example: *a – z* , *A – Z* , or *0 – 9* .	Y	–
a	Alphanumeric character placeholder (entry optional).	Y	–
9	Digit placeholder (entry optional). For example: *0 – 9* .	Y	–

C	Character or space placeholder (entry optional). This operates exactly like the *&* placeholder and ensures compatibility with Microsoft Access.	Y	–
?	Letter placeholder. For example: *a – z* or *A – Z* .	Y	–
Literal	All other symbols are displayed as literals; that is, as themselves.	Y	Y

Examples

Here are some examples of alphanumeric display masks.

Alphanumeric Edit Masks

Mask	**Description**	**Sample Input Data**	**Display**
###\\-##-####	String character mask for SSN input digits only (since hyphen is a valid mask character we need to escape it with the double back slashes, -).	123456789	123-45-6789
9,999.99	Number mask	123456	1,234,56
[2-5]9	First digit takes a range of digits from 2 – 5 (inclusive) and second digit takes any digit.	41	41
[A-F]AAA	First character takes a range of letters from A – F (inclusive) and subsequent characters take any alphanumeric character.	babc	babc

Date and Time Edit Masks

The following sections describe the date and time mask characters.

Mask Character Descriptions

Character descriptions for date and time masks

Mask character	**Description**
M	For month digit replacement. (for example, **MM /dd/yy** for month in a date format)
d	Day digit replacement (for example, **dd/MM/yy** for day of the month in a date format)
y	Year digit replacement (for example, **dd/MM/yy** for 2 digit year and **dd/MM/yyyy** for 4 digit year)
E	Week day placeholder
H	For 24 hour time hour replacement (for example, **HH:mm:ss**)
h	For 12 hour time hour replacement (for example, **hh:mm:ss**)
m	For minute replacement digit (for example, **hh:mm:ss**)
s	For seconds replacement digit (for example, **hh:mm:ss**)
T	For **AM/PM** placeholder

Date and Time Masks

Date and time masks

Mask	**Example**	**Input mask?**	**Display mask?**
M	7	–	Y
MM	07	Y	Y
MMM	Mar	Y	Y
MMMM	March	–	Y
d	7	–	Y

dd	07	Y	Y
EEE	Wed	Y	Y
EEEE	Wednesday	–	Y
yy	99	Y	Y
yyyy	1999	Y	Y
H	22	–	Y
HH	02	Y	Y
h	1	–	Y
hh	01	Y	Y
m	1	–	Y
mm	01	Y	Y
s	1	–	Y
ss	01	Y	Y
t	A/P	–	Y
tt	AM/PM	Y	Y

Examples

Here are some examples of date masks.

Example Date Masks

Mask	Description	Sample Data	Display	
MM/dd/yy	Month, day, and year (2 digit year)	010203	01/02/03	
MM/dd/yyyy	Month, day, and year (4 digit year)	01022003	01/02/2003	
MMM/dd/yy	Short month name (Jan, Feb, Mar, etc.), day, and year	Jan0203	Jan/02/03	
MMMM dd yy	Long month name (January, February, etc.) day, and year	January0203	January 02 03	
dd/MM/yy	Day, month and year	020103	02/01/03	
EEE dd/MM/yy	Short day name (Sun, Mon, etc.) day, month, year	Thu020103	Thu 02/01/03	
EEEE dd/mm/yy	Long day name (Sunday, Monday, etc.) day, month, year	Thursday020103	Thursday 02/01/03	

Here are some examples of time masks.

Example Time Masks

Mask	Description	Sample Data	Display
HH:mm:ss	Hours (0-23), minutes, seconds	210101	21:01:01
hh:mm tt	Hours (0-11), minutes and AM/PM	1001AM	10:01 AM
hh:mm	Hours and minutes only	1220	12:20

Display Masks (Numeric)

Numeric display picture characters

Character	Description
9	Specifies a numeric character.

.	Specifies the decimal separator. Governed by the country setting of the field.
,	Specifies the thousands separator. Governed by the country setting of the field.
$	Specifies the currency symbol. Governed by the country setting of the field.
Z	Suppresses leading zeroes.
*	Suppresses leading zeroes and replaces them with asterisks.
S	Specifies sign. Governed by the country setting of the field.
+	Specifies sign. Displayed as a blank character if the field's value is zero or negative. Not governed by the field's country setting.
-	Specifies sign. Displayed as a blank character if the field's value is zero or positive. Not governed by the field's country setting.
cr	Specifies credit. Displayed as blank characters if the field's value is zero or positive. Not governed by the field's country setting.
db	Specifies debit. Displayed as blank characters if the field's value is zero or positive. Not governed by the field's country setting.

Font Mapping at Execution Time

There is one cross-platform table of font definitions used by various AppBuilder tools including Window Painter, Report Writer, C runtime, and Java runtime. A configuration file named **wpmodel.txt** is in the AppBuilder client runtime. This file externalizes the font mappings used by the client runtime. The client runtime references the font.ini to retrieve the mapping for any logical font names specified in panel files. The mappings in the font.ini match those previously defined internally to the product. Thefont.ini file is located in the same directory as the system configuration file (hps.ini).

Refer to Font Mappings for more details about how AppBuilder refers to the logical font and to which execution time font this corresponds.

Refer to Font Definition Syntax to understand how to edit the font definitions in the wpmodel.txt file.

Changes to the font mappings in the font definition section of the configuration file (wpmodel.txt) might affect the appearance of the font on windows in the execution environment.
Be careful when making any changes to the wpmodel.txt file.
When you do make changes, you must follow the Font Definition Syntax rules.

There are several things that you must note if the wpmodel.txt file is changed. If you make a syntax mistake with the font mapping (for example, no valid point size specified or a misplaced comma), AppBuilder defaults to not specifying a font for any objects using that logical name and this results in the system font being used. No warning is given about the substitution of fonts. If the syntax is correct, but no such font exists for the font mapping specified, AppBuilder attempts to use the font as specified and it is left to the operating system to provide any substitution. No warning is given about the substitution of fonts. Any unknown or unsupported styles are ignored, without any warning. Sizes of window controls are not determined by these font definitions, so you must make sure that the chosen font size fits the controls. The existing font mapping that is specified in the wpmodel.txt or in Window Painter are not affected by this. This font mapping applies only to AppBuilder-defined standard fonts.

Font Mappings

Font mappings for runtimes shows how AppBuilder maps the logical font names shown in the workstation tool (such as Window Painter) to the actual fonts used at execution time. Previously, this table was statically defined internal to the runtime. Now, you can edit the values in font.ini to change the mappings. If you change font.ini, the mappings in this table are only of value for historical purposes.

Font mappings for runtimes

Logical Font	**Window Painter**		**Windows C Runtime**	
	Face Name	**Size**	**Face Name** [a,b]	**Size**
MODERN0	Courier	10	Courier	10
MODERN10	Courier	12	Courier	12
MODERN12	Courier	15	Courier	15
ROMAN8	Tms Rmn	8	Tms Rmn	8
ROMAN10	Tms Rmn	10	Tms Rmn	10
ROMAN12	Tms Rmn	12	Tms Rmn	12

ROMAN14	Tms Rmn	14	Tms Rmn	14
ROMAN18	Tms Rmn	18	Tms Rmn	18
ROMAN24	Tms Rmn	24	Tms Rmn	24
SWISS8	Helv	8	Helv	8
SWISS10	Helv	10	Helv	10
SWISS12	Helv	12	Helv	12
SWISS14	Helv	14	Helv	14
SWISS18	Helv	18	Helv	18
SWISS24	Helv	24	Helv	24
SYSTEMFONT	System	10	System	10

a. The Tms Rmn font maps to MS Serif font in the Windows operating system.
b. The Courier font maps to MS Sans Serif font in the Windows operating system.

Font Definition Syntax

```
[<LOGICAL NAME>]
<platform>=<face name>,<point size>,<style1>, ..., <style n >
<platform>=<face name>,<point size>,<style1>, ..., <style n >
GUI_SIZE= width, height
PRINT_POINT_SIZE= width, height
USAGE= <usage>, <usage>
```

where:

Font definition syntax

Variable	**Description of value**
<LOGICAL NAME>	Used as the panel file symbol, all uppercase, no punctuation or spaces. This logical name should not be all numeric except for Report Fonts.
<platform>	Can be any of these: • HTML • Java • Win32
<face name>	Name of the face sent to the underlying operating system. If you are executing your application in the Windows environment, this is the name of the actual font as defined in the Fonts settings for the machine. In other words, Arial is not the same as Arial Bold, which is not the same as Arial Bold Italic.
<point size>	Positive integer size sent to the operating system
<style1>,... <style *n* >	Can be any of these: • BOLD • ITALIC • UNDERSCORE (not supported for Java) • STRIKEOUT (not supported for Java) Multiple, comma-separated styles may be specified. Not all styles may be supported by all platforms
GUI_SIZE= *width, height*	Allows you to override the font size reported by the operating system. Width and height are positive integer numbers. Empty by default. This does not include edit field borders
PRINT_POINT_SIZE= *width, height*	Point size for Report Painter Width and height are positive integer numbers.

<usage>	Specify which tool uses this font (to prevent Window Painter from displaying mainframe Report Writer fonts). These include: • Window (as in Window Painter) • Report (as in Report Painter and Report Writer) Multiple, comma-separated values may be specified.

To add a new entry, simply create a unique logical name and add the platform and usage lines (and any other needed lines).

The PRINT_POINT_SIZE is specified in width, height format. These values specify width and height of a character of this font in points (or pixels) for resolution of 240 dots per inch (DPI, which means one point = 1/240 of an inch). If PRINT_POINT_SIZE is not specified for a particular font, the system font metrics are used. Use this setting to control printing of the report or diagram if the default font metrics are not appropriate in your case. The width of the font is used to calculate the horizontal offset when printing a section horizontally if the horizontal section direction has been specified along with a non-zero number of appearances in a row at design time. If sections that are printed horizontally overlap, specify a greater value for the font width. The height of the font is used to calculate the vertical offset when printing next to the section line. If section lines overlap each other, specify a greater value for the font height.

This is the font used to create font metrics for CharCoordinateSystem.
```
[CCS_FONT]
Java=sansserif,12,BOLD
Win32=System,10
HTML=System,10
Usage=Window
```

This is an example of how to set up Arial Bold Italic 14:
```
[ArialBoldItalic14]
Windows=Arial Bold Italic,16,BOLD,ITALIC
Java=Arial Bold Italic,16,BOLD,ITALIC
HTML=Arial Bold Italic,16,BOLD,ITALIC
Usage= Window
```

Custom Font Definitions

Window Painter honors anycustom font attributes that it finds in a panel. It displays CustomFont in the Font property and saves CustomFont if the panel has been modified except for the CustomFont. If you want to modify the Custom Font, then you must use the Font.ini mechanism.

A script written especially for AppBuilder queries all Window panel files and generates new Font.inientries for those Custom Fonts that use the exact same attributes.

The result is aFont.Custom.ini file containing all custom font definitions from all Window panel files.