

Overview

Overview of PM Options

COOL RIDE Presentation Manager (PM) is used to visually create a frame window, equip it with controls and if necessary modify the window and its controls. For designing such a window equipped with controls—a PM window—you have to use the PM editor the main tool of which is the Gadgets palette. The latter is the source of controls for the frame window that the PM editor provides whenever you open the editor.

A PM window is a frame and controls embedded into it with the help of the Gadgets palette. The frame and the controls are PM objects with their own properties. You create a PM window to implement a graphic user interface within a COOL program. To link the PM window to the source code of the program, you can use events associated with PM objects (controls).

Note

The frame window, or simply the frame, is a single form on which you can add and modify controls picking them up from the Gadgets palette. The frame and controls placed on it are PM objects with specific properties and associated events, if any. The property is an attribute of an object that you set to define one of the object's characteristics or an aspect of its behavior. The event is an action, recognized by an object, for which you can write code to respond. Events can be generated either by a user action (such as clicking the mouse or pressing a key) or by program code.

As a COOL RIDE application developer, you work with the source code editor of the COOL RIDE Development Environment system (Start|Programs|COOL RIDE Application Server|Development Environment). You commonly start designing a PM-based graphic user interface by plugging a Frame Window form into the text of a program module of the PM CLIENT PROCESS category. Such a plug-in has to be part of the WINDOW PM statement.

Definition of WINDOW PM Statement

Handling Frame Window and Gadgets

Standard Tabs of PM Object Dialog

Programming Options for Handling PM Object Properties

Overview

Definition of WINDOW PM Statement

In COOL—the COOL RIDE programming language—the WINDOW PM statement is defined as follows:

```

WINDOW

  [ PM ] { blob_expression | string_expression }

  [ TIMEOUT integer_expression_sec ]

      { ON { event_identifier | event_string_literal | OPEN | TIMEOUT } DO

          statement ; ... }

      [ OTHERWISE { statement ; ... } ]

END_WINDOW

```

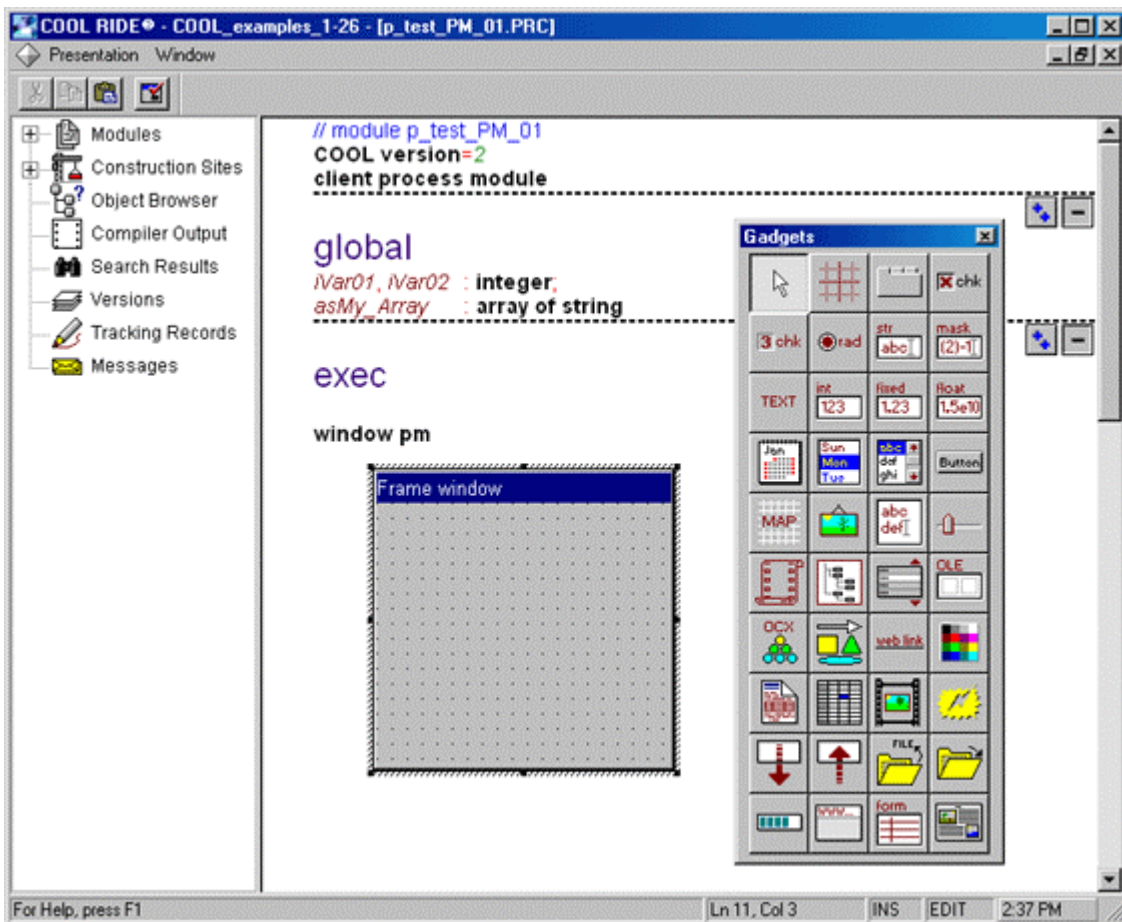
The arguments specified in the above definition have the following meaning:

PM	An optional parameter (modifier) specifying the PM type of the window. The PM modifier indicates that the window's functionality is supported by the COOL RIDE Presentation Manager. When omitted, the use of PM is assumed by default.
<i>blob_expression</i>	A BLOB data type expression, typically a plug-in, that defines the window. Such a blob is actually a .PMB file created by the COOL RIDE Presentation Manager during its development session.
<i>string_expression</i>	A STRING data type expression that defines the window. Such a string is considered to be a PM-specific window definition (The definition syntax is out of the scope of the current document.)
TIMEOUT	An optional parameter for specifying the maximum duration between any two consequent events.
<i>integer_expression_sec</i>	An expression of the INTEGER data type defining the maximum duration, in seconds, between any two consequent events generated by the PM window's gadgets. If it is exceeded, the standard event TIMEOUT is generated.
ON <i>event</i> DO <i>statements</i>	A clause defining the <i>statements</i> to process when the specified <i>event</i> takes place. The <i>event</i> can be specified either as an <i>event_identifier</i> or as an <i>event_string_literal</i> .
<i>event_identifier</i>	A user-defined name of an event that has been defined for the PM gadgets.

Overview

<i>event_string_literal</i>	A string literal naming the event.
OPEN	A standard event that occurs when the window is opened.
TIMEOUT	A standard event that occurs when a specified timeout (if any) between any two subsequent events is exceeded.
<i>statement</i>	COOL statement(s) that are to be preformed upon occurrence of the event.
OTHERWISE <i>statements</i>	A clause defining the <i>statements</i> to process when the event that currently occurs have no appropriate ON ... DO clause. Within the OTHERWISE clause, the built-in function LAST_WINDOW_EVENT might be used to determine the name of the last event.

Note that the above syntax of the WINDOW PM statement is only one of two variants of the WINDOW statement's definition. For complete information about the statement and its usage, refer to "COOL RIDE Language Reference Manual."



To plug a Frame Window form into a WINDOW PM statement within your program module of the PM CLIENT PROCESS category, place mouse pointer just after the PM keyword (or,

Overview

WINDOW if PM is omitted) and then choose **Insert PM Window** from the **Edit** menu. As a result, a blank form of the window with caption “Frame window” appears.

To activate the Presentation Manager window and get access to the PM Gadgets palette, it is enough to double-click on the form. The form remains selected, and the PM Gadgets palette reflects this by showing the Select item being pressed. (The item is located in the upper left corner of the Gadgets palette).

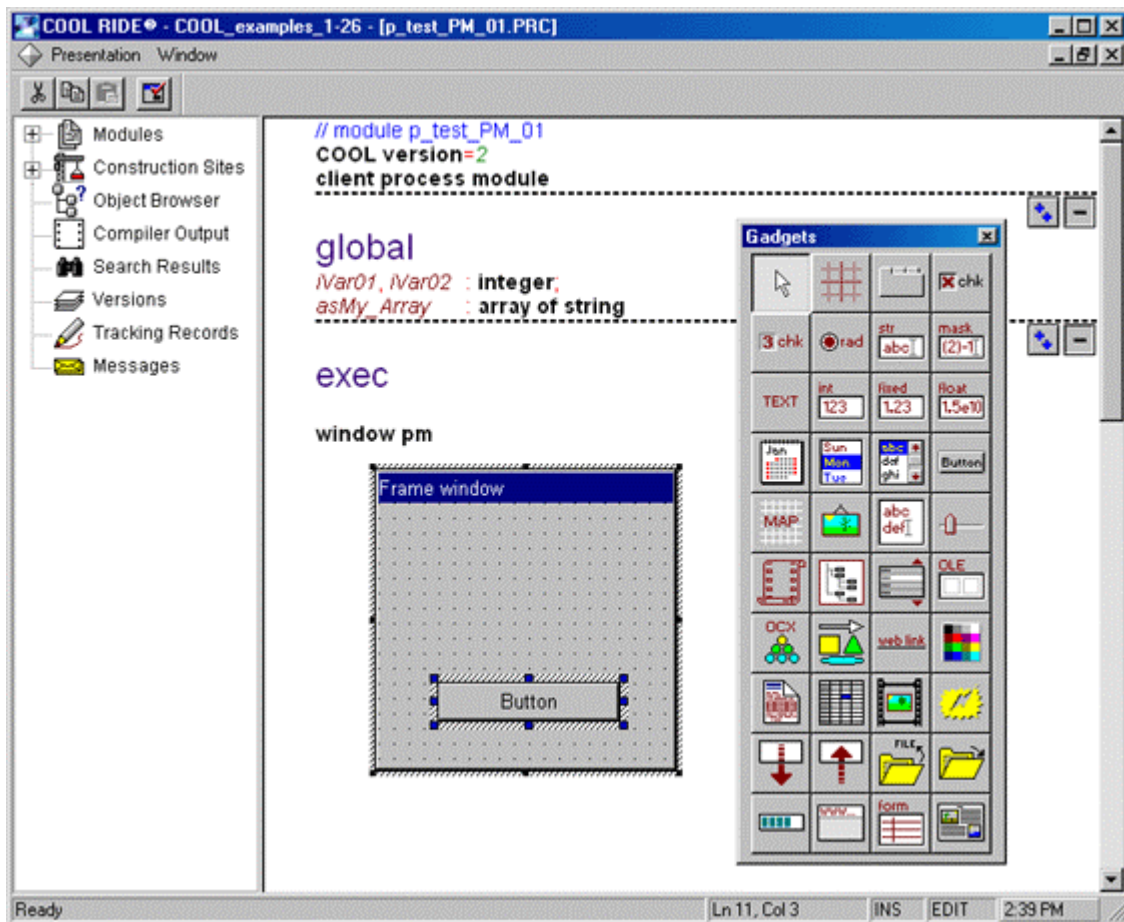
To get the short description of a particular item available on the Gadgets palette, you should move the pointer of your mouse on the item. While the mouse pointer is over the palette’s item, a tiny popup with proper information is shown.

Frame Window and Gadgets

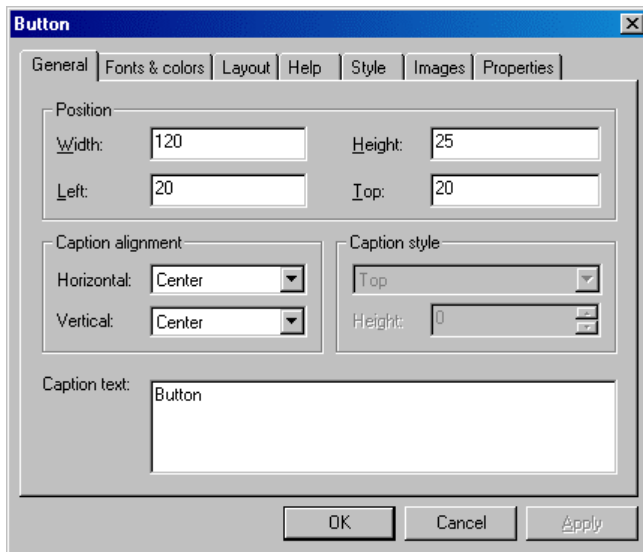
Handling Frame Window and Gadgets

To put a control (for example, Button) provided by the Gadgets palette on the Frame window, you have to do as follows:

1. Select the corresponding gadget by clicking on it within the palette.
2. Move the mouse's pointer onto a proper place within the Frame window.
3. Click with the mouse.



For a PM object—the Frame window or any control placed on it with the help of the Gadgets palette—you can open a multi-tab dialog with the detailed information on the object. The dialog named after the PM object (for example, Frame or Button) lets you not only view but also adjust the object's parameters. To open such a dialog, double-click on the desired object—the window's pane or any control on it. Or, right-click on the object and when a local menu will appear, choose **Properties**.

Standard Tabs of PM Object Dialog

For a particular PM object, the number of the dialog's tabs, the tabs names and tabs contents are determined by the object's type (see the list of gadgets above).

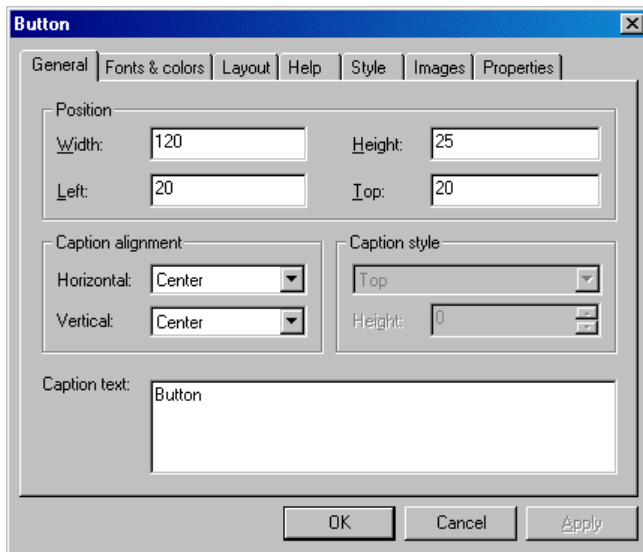
Standard Tabs of PM Object Dialog

No matter how significant the peculiarities of the type of a particular PM object may be, all PM objects—the Frame window and its controls created with the help of the Gadgets palettes—have much in common. As a result, all PM Object dialogs each have five tabs of similar design. These “standard” tabs are **General**, **Fonts & Colors**, **Layout**, **Help**, and **Properties**. They combine the controls (data entry fields, list boxes, checkboxes, etc.) that help to specify the values of the PM object's properties listed on the **Properties** tab. Such a specification can be done on user-interface design stage. As a COOL RIDE application developer, you can also specify the properties' values in the program's code by using the SET PROPERTY statement in the program's source code, provided proper names are already assigned to the properties with the help of options of the **Properties** tab. (For further information on the SET PROPERTY statement and its usage, refer to “COOL RIDE Language Reference Manual.”).

General

The **General** tab of a PM Object dialog contains four control groups: **Position**, **Caption Alignment**, **Caption Style**, and **Caption Text**.

Standard Tabs of PM Object Dialog



Depending on the type of a particular PM object, some of the controls on the **General** tab may be disabled.

The Position control group

Four data entry fields, **Width**, **Height**, **Left**, and **Top**, allow you to specify integer values for four properties $\$HSize$, $\$VSize$, $\$Hloc$, and $\$VLoc$, respectively (see “Standard Properties of PM Objects”).

The Caption Alignment control group

Each of two dropdown list boxes, **Horizontal** and **Vertical**, let you make a choice from three options—Left (1), Center (2), and Right (3) for $\$CaptionHAlign$ and Top (1), Center (2), and Bottom(3) for $\$CaptionVAlign$, respectively (see “Standard Properties of PM Objects”).

The Caption Style control group

Two controls—a dropdown list box and data entry field—are available for customizing the display style of the PM object’s caption. To customize the caption’s location, use the list box with a choice of three options—None (1), Top (2), and Left (3); the chosen value will be assigned to $\$CaptionPos$. To specify the integer value of the size (height or width) of the display field for the caption, use the data entry field; the value will be assigned to property $\$CaptionWidth$ (see “Standard Properties of PM Objects”).

The Caption Text control group

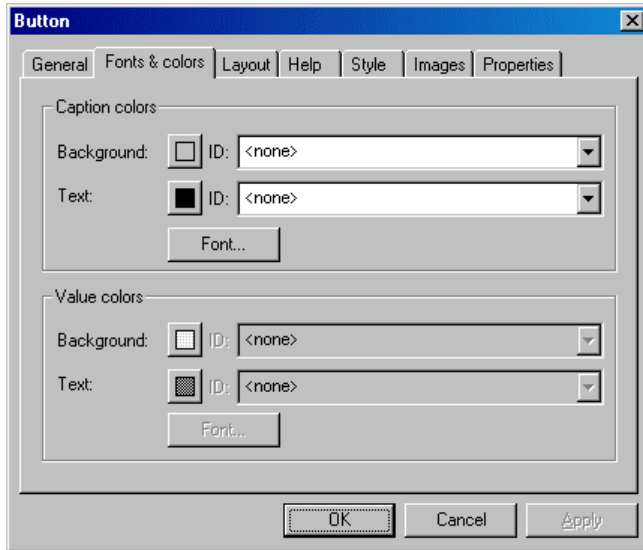
The only control—data entry field—lets you define a string value for the PM object’s caption; the value will be assigned to property $\$Caption$ (see “Standard Properties of PM Objects: $\$Caption$ ”).

Font&Color

The **Fonts&Color** tab of a PM Object dialog contains two control groups: **Caption Colors** and **Value Colors**. The groups agree in functionality—for text and its background, you can specify




Standard Tabs of PM Object Dialog

color and font. The groups help to customize color and font respectively for the PM object's caption and data value entry fields.

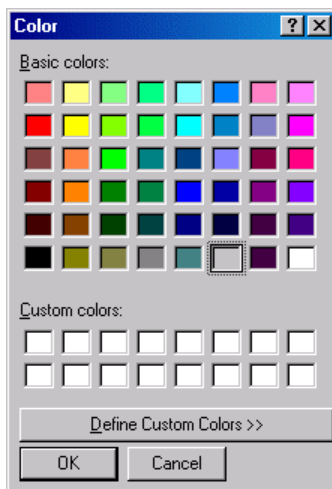


Depending on the type of a particular PM object, some of the controls on the **Fonts&Color** tab may be disabled.

Specifying Color

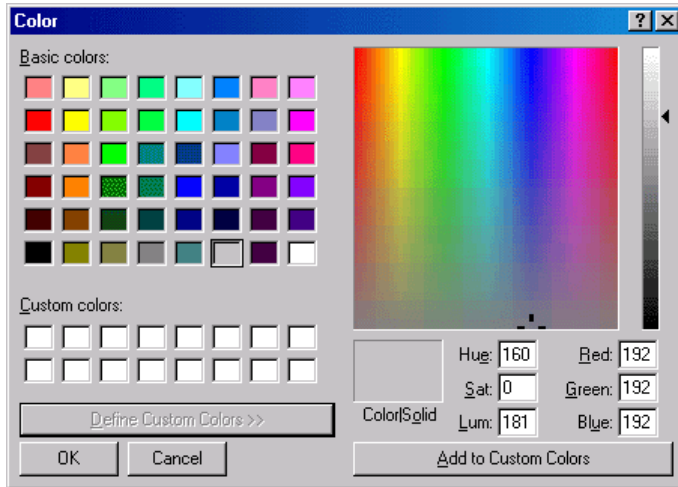
A pair of controls is provided for specifying a value and identifier for a color, and it makes no difference whether it is the text color or the background one, and whether the color is for the PM object's caption, or for the value in a data entry field within the PM object, or for the scale of the Progressbar control, etc. Such a pair of controls consists of a pushbutton like  or , and a dropdown list box like .

Depending on requirements of your application's design, you have two options for specifying color. The above-mentioned pushbuttons give the way to the first of two options. With a click on such a button, the COOL RIDE Color Palette dialog appears.



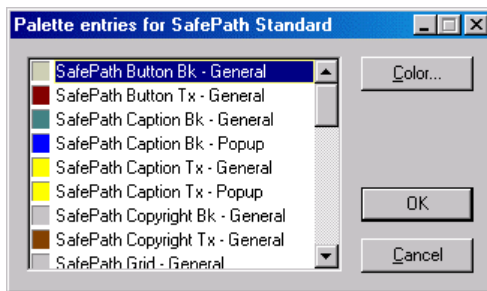
Standard Tabs of PM Object Dialog

By default, the color palette is collapsed with minimum of its options at your disposal. To expand the palette, click the **Define Custom Colors** button.



Appropriate settings of a color via the color palette results in the integer value of the color assigned to a corresponding property (*\$CaptionBackColor*, *\$CaptionTextColor*, *\$ValueBackColor*, or *\$ValueTextColor*).

Requirements of your application's design may imply to use the other option of the color specification provided by the above-mentioned dropdown list box like ID: . The list of color identifiers (*ColorID*) is not empty if a color scheme has been created for use in application design.



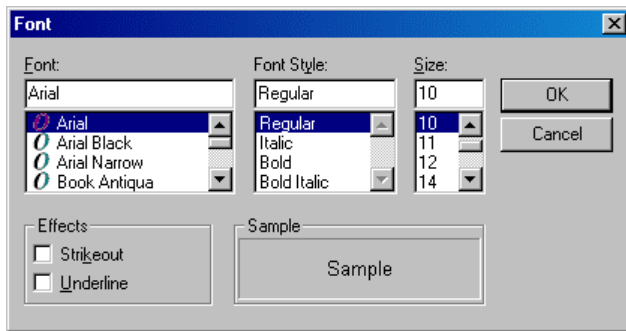
Such a scheme can be created with the help of the COOL RIDE Color Palette designer. (For details, refer to “COOL RIDE Development Environment User Manual”).

If you use the above dropdown list box to choose the desired color from the list of color IDs, the string value of the *ColorID* will be assigned to a corresponding property (*\$CaptionBackColorID*, *\$CaptionTextColorID*, *\$ValueBackColorID*, or *\$ValueTextColorID*).

Specifying Font

The control groups on the **Fonts&Color** tab each contain the **Font** pushbutton helping to open the **Font** dialog where you can make appropriate settings for the font's parameters.

Standard Tabs of PM Object Dialog

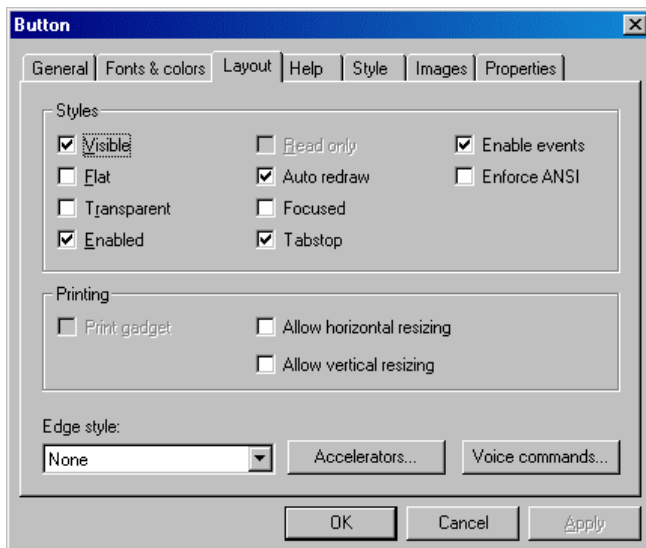


You make font settings independently for the PM object's caption and for the values in data entry field(s) within the object. The results of the settings for **Font**, **Font Style** + **Effects**, and **Size** are respectively

- string values in properties $\$CaptionFontFace$ and $\$ValueFontFace$
- choice values in properties $\$CaptionFontStyle$ and $\$ValueFontStyle$
- integer values in properties $\$CaptionFontSize$ and $\$ValueFontSize$

Layout

The **Layout** tab of a PM Object dialog contains two control groups (**Styles** and **Printing**) and three “standalone” controls—a dropdown list box (**Edge Style**) and two pushbuttons (**Accelerators** and **Voice Commands**).



Depending on the type of a particular PM object, some of the controls on the **Layout** tab may be disabled.

Standard Tabs of PM Object Dialog**The Style control group**

Nine checkboxes help to customize display and behavior of the PM object. Each checkbox has an associated property the Boolean (BOOL) value of which is determined by the checkbox state (see “Standard Tabs of PM Object Dialog: Properties”).

<u>Checkbox</u>	<u>Meaning</u>	<u>Property ID</u>
Visible	A visibility of the PM object (control) on the pane within the Frame window.	<i>Visible</i>
Flat	A two-dimensional presentation of the PM object (control) that also shows the third dimension when the checkbox is not marked.	<i>Flat</i>
Transparent	A transparency of the PM object (control); that is, the underlying objects are visible if the checkbox is marked.	<i>Transparent</i>
Enabled	An accessibility of the PM object.	<i>Enabled</i>
Read Only	The Read Only mode for the data value entry field(s) of the PM object.	<i>ReadOnly</i>
Auto Redraw	The Auto Redraw mode for the PM object display.	<i>AutoRedraw</i>
Focused	The In Focus mode for the PM object display. (On a particular frame only one control (PM object) may be in focus.) Setting the mode for the given object (control) discards the similar setting that has been done for another PM object(s) if any.	<i>Focused</i>
Tabstop	The Tabstop mode for the PM object display. (When the mode is enabled, you can place the PM object in focus by repeatedly pressing the Tab key on the keyboard).	<i>TabStop</i>
Enable Events	The Enable Events mode for the PM object.	<i>EventsEnabled</i>
Enforce ANSI	The Enforce ANSI mode for the PM object. When the mode is enabled, a symbol with the matching counterpart from the ANSI standard of symbols is accepted as an entry; otherwise, the entry is impossible.	<i>AnsiEnforced</i>

Standard Tabs of PM Object Dialog

The Printing control group

Three checkboxes constitute the control group. Each checkbox has an associated property the Boolean (BOOL) value of which is determined by the checkbox state (see “Standard Properties of PM Objects”).

To enable the Print Gadget option, you should mark the corresponding checkbox; this will assign *TRUE* to property *\$Print*.

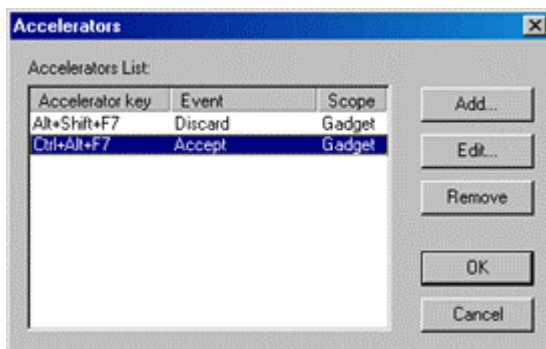
To enable the Gadget Resizing options—horizontal and vertical—you should mark the corresponding checkboxes; as a result, properties *\$HResizeForPrint* and *\$VResizeForPrint* will be of the *TRUE* value.

The Edge Style list box

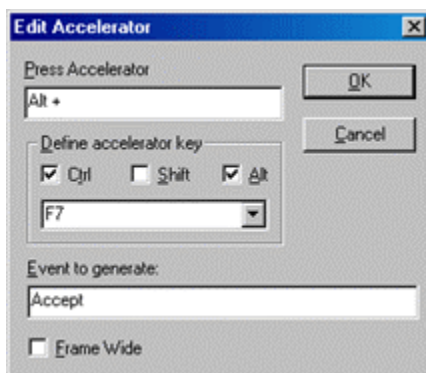
The dropdown list box let you make a choice from six options—None (1), Inset (2), Raised (3), Raised and Inset (4), Wireframe (5), and NoGap (6)—for property *\$Edge*.

The Accelerator button

The button opens the **Accelerators** dialog where you can define one or more keyboard accelerators associated with the PM object (control). The defined accelerators as a string value will be assigned to property *\$Accelerators* (see “Standard Properties of PM Objects: *\$Accelerators*”).



To define/edit an accelerator parameters, open the **Add/Edit Accelerator** dialog with the help the **Add** or **Edit** button, respectively.

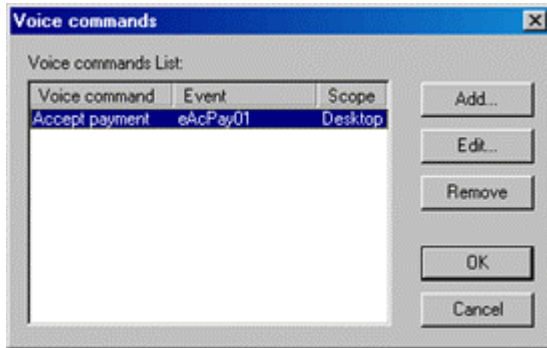


Standard Tabs of PM Object Dialog

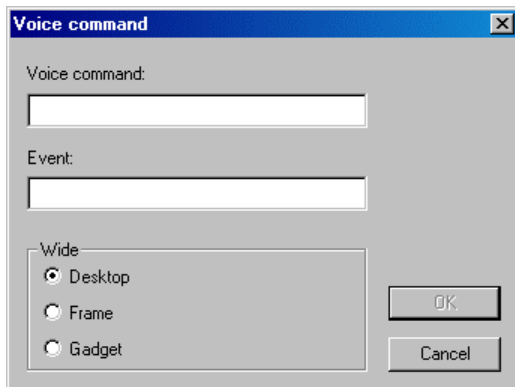
For an accelerator, you have to define not only hot key(s) but also the name of the event handler. Note that if you mark the **Frame Wide** checkbox, the scope of the accelerator validity will be extended on the Frame pane where the PM object (control) is located.

The Voice Commands button

The button opens the **Voice Commands** dialog where you can define text for one or more voice command associated with the control. The defined commands as a string value will be assigned to property `$Commands` (see “Standard Properties of PM Objects: `$Commands`”).



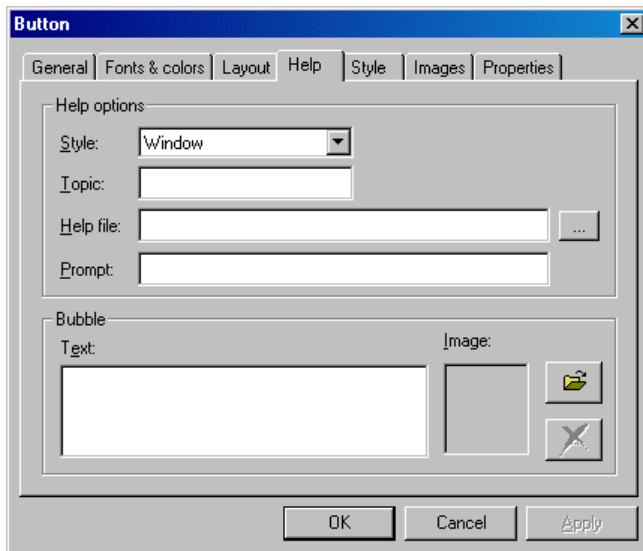
To define/edit a command parameters, open the Voice Command Parameters dialog with the help the **Add/Edit** button.



For a voice command, you have to define not only the command's text but also the name of the event handler. Note that the Scope radio buttons lets you choose the scope of the command validity—the PM object (control within the Frame window), the Frame pane where the PM object is located, or the entire desktop of your computer (the window with the PM object may be hidden from view).

Standard Tabs of PM Object Dialog**Help**

The **Help** tab of a PM Object dialog contains two control groups: **Help Options** and **Bubble**.

**The Help Options control group**

The control group combines four controls. The **Style** dropdown list box lets you make a choice from two options—Window (1) and Bubble (2)—for property *\$HelpStyle* (see “Standard Properties of PM Objects: *\$HelpStyle*”). The first of two options enforces display of the help for the PM object in a standard window; the second option makes the help be shown in a tiny popup window.

The **Topic** data entry field lets you specify the context-identifier of the help topic for the current PM object. Such an identifier is a unique numeric expression; to use it properly you must add the **#** symbol as a prefix. Such a string value will be assigned to property *\$HelpTopic*. Note that the above-mentioned help topic is commonly part of some help file of the application to which the PM object (control) belongs as an element of user interface.

The **Help File** control is a data entry field with button opening a filer. The control allows you to specify the name of a proper help file. Once it is done, the corresponding string value is assigned to property *\$HelpFile*. The full path to the file is used for the testing purposes. During the run-time, only the file’s name and extension are in use.

The **Prompt** data entry field lets you define a prompt that will appear on the status bar of the Frame window when the mouse pointer is hovering over the control (PM object) located on the window. Once defined, the string value of the prompt becomes the value of property *\$Prompt*.

The Bubble control group

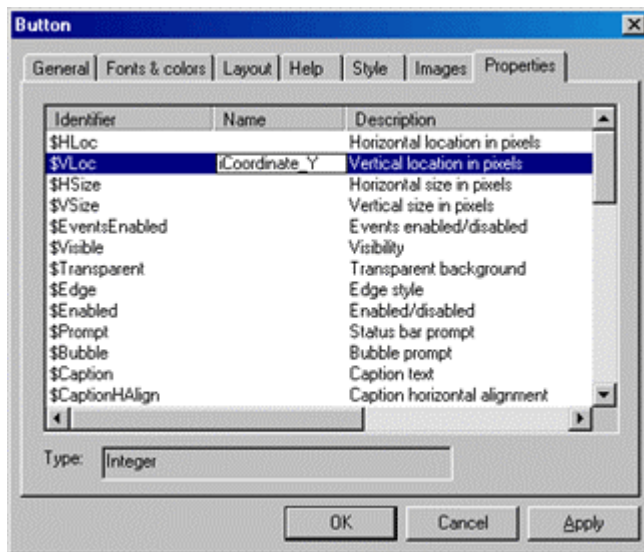
The control group helps to define a popup that appears whenever a mouse pointer is hovering over the current PM object (control). The **Text** entry field lets you type the text you wish to be shown in the popup. The string value of the entered data will be assigned to property *\$Bubble* (see “Standard Tabs of PM Object Dialog: Properties”).

Standard Tabs of PM Object Dialog

In addition to text you may specify an image that will be displayed within the same popup. The Image control—preview window and two pushbuttons—helps to find the desired image (.BMP or .GIF file). Once the image is specified, the corresponding BLOB value will be assigned to property *\$BubbleImage*.

Properties

The **Properties** tab of a PM Object dialog lists the PM object's properties and allows you to name them for use in the application's source code as program variables whose values are generally modifiable.



More than two dozens of items listed on the **Properties** tab are marked as standard properties—that is, mandatory for almost all types of PM objects (the frame window and its controls). The identifiers of such properties start with the \$ sign—the marker of the standard. The properties specific for a particular type of an object have no \$ in the first position of the property's identifier.

Note that for a selected item of the Properties list, the **Type** display field on the **Properties** tab shows the type of data feasible for the current property.

To get control over a particular property of a PM object in a programming module of your application, you have to assign a name to the property. Through the name you gain access to the property's value.

To name a property, double-click on the corresponding item in the Properties list. It will activate the data entry field in the **Name** column. The field lets you assign the property a name that later can be used in your program to set the property some value and then get it when necessary. Two statements—`SET_PROPERTY` and `GET_PROPERTY`—are used for these purposes. The property exists within the scope of the closest pair of window statements (`Window` and `End_Window`). You can change the property's value; the value is accessible—statements `SET_PROPERTY` and `GET_PROPERTY` are applicable to it—for the client process's main thread only.

Handling PM Object Properties

Programming Options for Handling PM Object Properties

In a program written in the COOL RIDE programming language (COOL) two statements, SET_PROPERTY and GET_PROPERTY, are used for handling the properties of PM objects—the Frame window and its controls created with the help of the Gadgets palette.

Set Property

To set a property value, COOL provides you with the SET_PROPERTY statement of the following syntax:

```

SET_PROPERTY

        { { property_name_identifier | property_name_string_literal | (string_expression) }

          TO expression } ,...

[ DELAYED ]

```

The arguments specified in the above definition have the following meaning:

<i>property_name_identifier</i>	An user-defined identifier of a property of a PM object.
<i>property_name_string_literal</i>	A string literal that names the property.
<i>string_expression</i>	An expression of the STRING data type that evaluates to the name of the property.
<i>expression</i>	An expression defining the value to be set for the property (see below “Types of PM Object Properties”).
DELAYED	Specifies that the actual execution of the statement should be delayed until the execution of an EXCHANGE statement or the next non-DELAYED SET_PROPERTY/GET_PROPERTY statement.

The following example illustrates a possible use of the SET_PROPERTY statement.

```

...
set_property "wpLastName"           to sLastName           delayed;
set_property "wpFirstName"         to sFirstName         delayed;
set_property "wpDate"              to iDate              to iDate
                                     delayed;
...
// Three commands set properties wpLastName, wpFirstName, and wpDate to the values of
// sLastName, sFirstName, and iDate, respectively.
// Note:
// wpLastName, wpFirstName, and wpDate are the names defined by the programmer via
// the COOL RIDE Presentation Manager
// sLastName, sFirstName, iDate are program variables with specified values.
...

```

Handling PM Object Properties

For further information, see the SET_PROPERTY statement in “COOL RIDE Language Reference Manual”

Get Property

To get a property value, COOL provides you with the GET_PROPERTY statement of the following syntax:

GET_PROPERTY

```

      { { property_name_identifier | property_name_string_literal | ( string_expression ) }
        INTO l-value } , ...
[ DELAYED ]

```

The arguments specified in the above definition have the following meaning:

<i>property_name_identifier</i>	An user-defined identifier of a property of a PM object.
<i>property_name_string_literal</i>	A string literal that names the property.
<i>string_expression</i>	An expression of the STRING data type that evaluates to the name of the property
<i>l-value</i>	The data object that receives the value of the property (see below “Types of PM Object Properties”).
DELAYED	Specifies that the actual execution of the statement should be delayed until the execution of an EXCHANGE statement or the next non- DELAYED SET_PROPERTY/GET_PROPERTY statement.

The following example illustrates a possible use of the SET_PROPERTY statement.

```

...
get_property "wpLastName"           into sLastName delayed;
get_property "wpFirstName"          into sFirstName delayed;
get_property "wpDate"               into iDate;
...

```

For further information, see the SET_PROPERTY statement in “COOL RIDE Language Reference Manual.”