ContextMenu Function

Description

The ContextMenu function creates and executes Context Menus. A Context Menu is a right mouse click menu specific to a control.

Syntax

status = **ContextMenu** (CtrlEntId, Message, Override, , , ,*CtrlOverride)*

Parameters

The ContextMenu function has the following parameters.

Parameter	Description
CtrlEntId	Has the format WindowName.ControlName, where WindowName is the identifier of the window that contains the affected control, and ControlName is the identifier of the control. Notice that a period separates the two values.
Message	This parameter must be "CREATEMENU". This will create the menu for the affected control. This function should be used during either the window's CREATE event or the control's GOTFOCUS event.
Override	(Optional) An @fm-delimited record containing the specification for the menu. If null, the function will create the menu for the control from the SYSREPOSMENUCONTEXT repository record. The repository record is created through the ContextMenu Builder within the User Interface Workspace tool.
param2	Reserved.
param3	Reserved.
param4	Reserved.
CtrlOverride	(Optional) Control Override. When valued with a fully qualified control name, the menu created for the CtrlEntId will be the same as the CtrlOverride control's menu.

Notes:

- ***The right-click on a control does not generate the GOTFOCUS event for a control.***
- ***The CREATEMENU message will create the menu for the control from the SYSREPOSMENUCONTEXT repository record. The repository record is created through the ContextMenu Builder within the User Interface Workspace tool.***
- ***When referencing the current control within source code, the SYSTEM properties will return null values, you will need to use the WINDOW properties. For example, suppose your menu item executes a stored procedure. If you needed to know which control has focus, then use Get_Property(@window,"FOCUS") rather than Get_Property("SYSTEM","FOCUS").***
- ***Using the CtrlOverride allows you to create a single menu in the ContextMenu Builder and then place this menu on more than one control.***

GOTFOCUS Event Example

```
* Create a context menu on the GOTFOCUS event of the State control on the CUST_ENTRY form.
* (The CUST_ENTRY form can be found in the Examples application.)

declare function ContextMenu
status = ContextMenu( CtrlEntId, "CREATEMENU" )
```

CREATE Event Example

```
* Create ContextMenus for all controls on the CREATE event of a form.

declare function ContextMenu
ctrlmap = Get_Property( CtrlEntId, "CTRLMAP" )
ctrlPos = 0
ctrlFlag = ""

loop
   remove thisControl from ctrlmap at ctrlPos setting ctrlFlag
      ctrlType = Get_Property( thisControl, "TYPE" )
      cMenu = ContextMenu( thisControl, "CREATEMENU" )
   while ctrlFlag
repeat
```

Using Another Control's Menu Example

```
To create a menu from a different control:
status = ContextMenu(CtrlEntId,'CREATEMENU','','','','','CUSTOMERS.NAME')
```

Modify and Creating Context Menus Programmatically

An existing Context Menu may be programmatically modified for current use. Call the ContextMenu using the CREATEMENU message and a Menu Structure parameter.

The Menu Structure parameter contains the same layout as the SYSREPOSMENUCONTEXT record. To override an existing menu pass in the fields that need to be overridden. To create a menu pass in a fully defined record. Menus changed or created programmatically are not saved to the repository. They are saved to the SYSREPOSMENUCONTEXT with a key of TEMP_CONTEXTMENU_*STATION* where *STATION* is the station id of the user.

The key to the SYSREPOSMENUCONTEXT record has a format of: Application name**Form name*Control name

The layout of the record is as follows:

Field Position	Name	Description
1	Menu Text	An @vm-delimited array of strings to be displayed on the menu.
2	Item ID	The same @vm-delimited array of strings in the Menu Text parameter.
3	Null	Reserved
4	Null	Reserved
5	Event Type	An @vm-delimited array of single character type values.

Value	Action
E	Execute an Entity.
M	Send an Event.

6	Event Name	An @vm-delimited array of Event Type dependent values.

Event Type	Value
E	The fully qualified name of the Entity to execute.
M	The fully qualified control id (ctrlEntId) of the control on which to raise an event.

7	Event Message	An @vm-delimited array of Event Type dependent values.

Event Type	Value
E	EXECUTE
M	The event to be raised, i.e. OPTIONS.

8	Event Parameters	An @vm-delimited array y parameters that need to be sent to the entity or control event.
9	Disabled Flags	An @vm-delimited array of Boolean values. If true values are passed then the corresponding menu items are disabled.
10	Checked Flags	An @vm-delimited array of Boolean values. If true values are passed then the corresponding menu items are marked as checked.
11	Hidden Flags	An @vm-delimited array of Boolean values. If true values are passed then the corresponding menu items are not visible.
12	Bitmaps	An @vm-delimited array of repository BMP entities to be used as the left side image for the corresponding menu items.