The CM_ and CN_ - Messages

(Revised, added new messages up to Delphi 7)
I was looking for a complete reference of the internal VCL messages, but I couldn't find anything.
So I tried to understand the meaning of the messages from the Delphi source-code. This also implies that perhaps I missed some information. Every contribution and correction will therefore be a great help in completing this reference.
You can contact me at my e-mail: webmaster@mh-nexus.de

All messages are declared in **controls.pas**. They are used internally by Delphi to notify controls of WM_-Messages or to provide information specific to Delphi as the dragging of a control.
CN is an abbreviation of control notification and CM stands for control message.
The CX_-Messages can be received by descendants of TControl not only of TWinControl, i.e. you don't need a handle to receive these messages.
Using the method Perform, you can send all kinds of messages to non-handle controls.

Control notifications are messages Windows doesn't send directly to the control but to its parent.
The parent translates the WM_ -Messages to CN_-Messages and sends them (broadcasts) to its child controls.

CM_Messages however, are exclusively "produced" by Delphi. A reference can be found in the table below.
(For an explanation of CN_-Messages have a look at WinAPI-Help; just replace the CN_-prefix with WM_.)

Often in the description there are hints where to find further information: You can look up the keywords denoted with "see also" in the Delphi-Help, the WinAPI-Help or the VCL/CLX-sourcecode.

Message	Params	Descripton
CM_ACTIVATE	None	The form is activated(= got the focus if not minimized). See also TCustomForm.Activate
CM_DEACTIVATE	None	The form is deactivated(= lost the focus). See also TCustomForm.Deactivate
CM_GOTFOCUS	None	The control got the focus.
CM_LOSTFOCUS	None	The control lost its focus.
CM_CANCELMODE	Sender	The control must cancel processing of user input (mouse capture, scrollbars, menu-loop and similiar). Sender points to the control that requested the canceling. See also WM_CANCELMODE.
CM_DIALOGKEY	CharCode, KeyData	A dialog key(i.e. VK_TAB,VK_LEFT, VK_RIGHT, VK_UP, VK_DOWN, VK_RETURN, VK_EXECUTE, VK_ESCAPE, VK_CANCEL) is pressed. See also WM_GETDLGCODE for further explanation and WM_CHAR for the Params.
CM_DIALOGCHAR	CharCode, KeyData	Same as CM_DIALOGKEY but the key that is pressed is a character.
CM_FOCUSCHANGED	Sender	To notify a form that the focus has gone from one control to another. Sender contains the new focusowner.
CM_PARENTFONTCHANGED	None	The parent of the control has changed its font. If ParentFont is set to true, set the Font-property to the parents font.
CM_PARENTCOLORCHANGED	None	The parent of the control has changed its color. If ParentColor is set to true, set the Color-property to the parents color.
CM_HITTEST	xPos, yPos	The control returns some HitTest information of the point (xPos, yPos). In a form this would be if the point is inside the Caption, the Border or the Client-Region. For the Caption it would return HTCAPTION. HitTest is mainly used to adapt a cursor-shape depending on where it points to.
CM_VISIBLECHANGED	None	Notifies the control that its Visible-property has been changed.
CM_ENABLEDCHANGED	None	Notifies the control that its Enabled-property has been changed. Can be used to determine when to draw a deactivated state for the control.
CM_COLORCHANGED	None	Notifies the control that its Color-property has been changed. Can be used to determine when the control needs to be redrawn.
CM_FONTCHANGED	None	Notifies the control that its Font-property has been changed.
CM_CURSORCHANGED	None	Notifies the control that its Cursor-property has been

		changed.
CM_CTL3DCHANGED	None	Notifies the control that its Ctl3D-property has been changed.
CM_PARENTCTL3DCHANGED	None	The parent of the control has changed its Ctl3D-property. If ParentCtl3D is set to true, set the Ctl3D-property to the parents Ctl3D.
CM_TEXTCHANGED	None	Notifies the control that its Caption or Text-property has been changed.
CM_MOUSEENTER	LParam	Notifies the control that the mouse has been moved inside a control. If LParam is 0, the mouse is uppon the control that received the message, else LParam points to the childcontrol the mouse is uppon. This message can be used to make a mouse-sensitive control that changes its state, like the flat SpeedButtons.
CM_MOUSELEAVE	LParam	Same as CM_MOUSEENTER, only that the mouse is moved outside of the control.
CM_MENUCHANGED	None	The menu has been changed. This can occur if items have been added to the menu or in a MDI-Application the MDI-Child's menu has been added to the MDI-Main menu.
CM_APPKEYDOWN	VirtKey, KeyData	To notify the control that an AppKey has been pressed. An AppKey is a MenuShortCut. Parameters are defined as for TWMKey. See also WM_SYSKEYDOWN.
CM_APPSYSCOMMAND	CmdType, xPos, yPos	To notfiy the control that a SystemMenuItem has been choosen or the minimize, maximize, close Buttons have been clicked. The Param CmdType specifies what kind of action occured, xPos and yPos contain the point of the cursor if the action was a "mouseaction". See also WM_SYSCOMMAND
CM_BUTTONPRESSED	GroupIndex, Sender	To inform the parent of a SpeedButton that a SpeedButton has been pressed. The Param GroupIndex is the GroupIndex of the SpeedButton, Sender is the SpeedButton that was pressed. Used to prevent more than one SpeedButton of the same group to be in pressed-state at the same time.
CM_SHOWINGCHANGED	None	The control must adapt its showing state, means to its internal FShowingState-variable that is Boolean and specifies if it is Visible(=True) or not.
CM_ENTER	None	The control got the focus
CM_EXIT	None	The control lost the focus
CM_DESIGNHITTEST	xPos, yPos	See CM_HITTEST. Only difference is that here the control is in designmode. So here you perhaps don't want to do the same as in execution-mode.
CM_ICONCHANGED	None	To notify the control that its Icon-property has been changed
CM_WANTSPECIALKEY	VirtKey, KeyData	A possibility for the control to get special keys when it is focused. That can be a key like ENTER or TAB that would be caught by another control like a Button or a Form. See also TMemo.WantTabs-property.
CM_INVOKEHELP	Command, Data	A possibility for your control to react on a special way if help is invoked. See also OnHelp-Event.
CM_WINDOWHOOK	Hook	A Window is hooked. The Param Hook contains a pointer to a Hook. See also HookMainWindow.
CM_RELEASE	Hook	A Window is unhooked. The Param Hook contains a pointer to a Hook. See also CM_WINDOWHOOK.
CM_SHOWHINTCHANGED	None	Control's ShowHint-property was changed to invisible/visible.
CM_PARENTSHOWHINTCHANGED	None	Control's ParentShowHint-Propert was changed. If it was changed to true the ShowHint-property is set to the parents ShowHint.
CM_SYSCOLORCHANGE	None	System colors have changed. If a control uses system colors like clBtnFace, the control is redrawn to reflect the changings.

CM_WININICHANGE	None	Win.ini was changed.
CM_FONTCHANGE	None	Font-property was changed. Redraw control to reflect the changings.
CM_TIMECHANGE	None	Sytem Time has been changed. See also WM_TIMECHANGED.
CM_TABSTOPCHANGED	None	The control's TabStop-property was changed.
CM_UIACTIVATE	None	UserInterface is activated. Used in ActiveX-controls to know when a control has been activated.
CM_UIDEACTIVATE	None	UserInterface is deactivated. Used in ActiveX-controls to know when a control has been deactivated.
CM_DOCWINDOWACTIVATE	Activate	Notifies the ActiveX-control when the container's document window is activated or deactivated. If Activate is True the window is in the act of activating else it is in the act of deactivating.
CM_CONTROLLISTCHANGE	Control, Inserting	Notifies that the ControlList (= list of child control's) has been changed. The Param Control specifies the Control that was removed/inserted. Inserting decides if a Control was inserted or removed. Can be used to update an internal list of child-controls. See also CM_CONTROLCHANGE.
CM_GETDATALINK	None	Requesting a datalink to the control. i.e. the message returns a DataLink to the control. See also TDataLink.
CM_CHILDKEY	CharCode, Sender	This message is called from a child-control of the control. If you want to catch a key that was sent to a child then return True. This is mainly used with ActiveX. The standard-implementation always returns False. Sender points to the child that sent the message, CharCode contains the Key. See also WantChildKey.
CM_DRAG	DragMessage, DragRec	Whenever an action that concerns Drag&Drop is to be processed by a control. DragMessage specifies the kind of action (dmDragEnter, dmDragLeave, dmDragMove, dmDragDrop, dmDragCancel, dmFindTarget). DragRec contains additional Information: Pos, Source, Target. For an example of how to use look in comctrls.pas for "procedure TCustomTreeView.CMDrag".
CM_HINTSHOW	None	Returns if a Hint can be shown. Your control can return false if the information that is needed to show the hint is not available else it returns true.
CM_DIALOGHANDLE	GetHandle, Value	If GetHandle is True return the handle of a non-modal Dialog associated with your app else set the handle to Value. This message is used to override the default behaviour of TApplication.DialogHandle. See also TApplication.DialogHandle.
CM_ISTOOLCONTROL	None	Returns if a control is a ToolBar-Control. To be a ToolBar-Control means that it can be placed on the OleContainer-Toolbar.
CM_RECREATEWND	None	The control recreates itself to reflect changes like the BorderStyle. The standard implementation sets the focus again after the control was recreated, if the control had the focus before it was destroyed.
CM_INVALIDATE	WParam	Provokes the invalidation of the control. If WParam = 0 then InvalidateRect is called. The control shouldn't be invalidated to often because of flickering.
CM_SYSFONTCHANGED	None	The number of fonts installed in the Operating System has changed. You should check if the font you use is still available.
CM_CONTROLCHANGE	Control, Inserting	A child was inserted or removed. If Inserted is true the child(Param Control) was inserted otherwise it was removed. In a toolbar this can be used to realign the children if one child has been removed and left a gap. See also CM_CONTROLLISTCHANGE.
CM_CHANGED	Sender	Properties of the control have been changed that also concern the control's parent. See also TControl.Changed.

CM_DOCKCLIENT	DockSource, MousePos	Client(DockSource.Control) was docked to the control. DockSource contains drag-and-dock information for a control, MousePos the position were to dock. See also CMDockClient.
CM_UNDOCKCLIENT	NewTarget, Client	Client was undocked from control. See also CMUnDockClient.
CM_FLOAT	DockSource	Client(DockSource.Control) floats. DockSource contains drag-and-dock information for a control.
CM_BORDERCHANGED	None	One of these properties has been changed: BevelEdges, BevelInner, BevelKind, BevelOuter, BevelWidth, BorderWidth. A control repaints or recreates itself to reflects the changes, if the ancestor-class doesn't already do it.
CM_BIDIMODECHANGED	None	Notifies the control when BiDiMode property is changed. See also CMBidiModeChanged-method.
CM_PARENTBIDIMODECHANGED	None	The parent of the control has changed its BiDiMode-property. If ParentBiDiMode is set to true, set the BiDiMode-property to the parents BiDiMode.
CM_ALLCHILDRENFLIPPED	None	Flip the control's children; that is, to move children on the left side of the control to the right side and vice versa. See also FlipChildren-method.
CM_ACTIONUPDATE	LParam	Is called as a reaction to the TContainedAction.Update-method. The message-handler updates the action specified in LParam. See also CMActionUpdate in Forms.pas for an example.
CM_ACTIONEXECUTE	LParam	Is called as a reaction to the TContainedAction.Execute-method. The message-handler executes the action specified in LParam. See also CMActionExecute in Forms.pas for an example.
CM_HINTSHOWPAUSE	WasActive, Pause	Is called to set the duration(Pause) the controls hint is to be showed. WasActive is true if the hint is already visible. See also HintShowPause.
CM_DOCKNOTIFICATION	Client, NotifyRec	This message is called when the client (i.e. the control that is docked) changed. The control can react if for example the Visibility was changed and hide/show the client. NotifyRec contains the message and its params that caused CM_DOCKNOTIFICATION to be called. This could be for example: CM_VISIBLECHANGED. See also SendDockNotification.
CM_MOUSEWHEEL	ShiftState, WheelDelta, XPos, YPos	The Mousewheel was moved above the control that receives this message. ShiftState specifies the keys and mousebuttons that were pressed. WheelDelta provides the distance that the weel is rotated. XPos and YPos contain the cursor position in screen-coordinates. See also DoMouseWheel-method and WM_MOUSEWHEEL.
CM_ISSHORTCUT	CharCode, KeyData	Returns 1 if the control uses the shortcut formed by CharCode and KeyData. A Menu-control for example returns 1 if CTRL+S is used by a MenuItem for saving. See also CMIsShortCut.
CM_RAWX11EVENT	Differs	Raw X11 Events (Linux pendants to the WM_-messages, but not as high-level). Params depend on the specific X11 event. See also X11 documentation.