

Standard Commands

The application shell includes a command processor. The command processor is a background dialog which controls the processing of defined commands. You can use the attribute `COMMAND-ID` in the Natural dialog editor to connect some types of dialog elements with a command. The following dialog elements can be attached to commands:

- Menu item. The menu is defined in the dialog editor.
- Tool bar item. The tool bar can be defined in the dialog editor and/or assigned dynamically depending on the dialog type. The dynamic tool bar and the dialog type must be defined in the application shell.
- Push button. Push buttons can be defined in the dialog editor or added dynamically.
- Bitmap. Bitmaps can be defined in the dialog editor or added dynamically.

Standard commands are commands which are predefined in the application shell and can be processed directly by frames.

A standard command can be a local or an internal command. Although the commands of type "action", "start a function", "start a browse" and "start application" can also be processed from most frames, they are not considered standard commands (because of their type).

Not all standard commands are supported by all frames. The respective frame descriptions indicate which frames support which standard commands. The module `Z_CUSTOM_CMD` is called to process any standard command which is not supported by a given frame.

The following overview describes the functionality of each standard command. A complete description of the processing of standard commands can be found in the corresponding frame description.

- Local Standard Commands
 - Internal Standard Commands
 - Tracing a Command
-

Local Standard Commands

Local standard commands are defined in the application shell administration system as local commands. They are invoked by a user action, for example, selection of a menu entry.

The processing of local commands can involve the activation of multiple internal commands in other dialogs.

- Z_APPLSTART
- Z_CALL
- Z_CANCEL
- Z_CLEAR
- Z_CLOSE
- Z_CONFIRM
- Z_EXIT
- Z_HELP
- Z_HELPCNTNT
- Z_HELPUSE
- Z_INFO
- Z_INFOBUFFER
- Z_INITBUFFER
- Z_NEXT
- Z_OK
- Z_OPEN
- Z_PREVIOUS
- Z_READ
- Z_REFRESH
- Z_SAVE
- Z_SAVEAS
- Z_SCRATCH
- Z_SEARCH

Z_APPLSTART

Starts a dialog with icon-based navigation and displays the entries of the entry level dialog. This command has no effect on current processing.

Z_CALL

Opens the dialog for direct call from functions and listings. All functions and listings can be started from this dialog. This command has no effect on current processing.

Z_CANCEL

Closes the current dialog. A confirmation window is optionally possible prior to closing of the dialog, in which the end user is asked to confirm whether or not modifications are to be applied.

Z_CLEAR

Closes current processing and supplies the dialog elements of the dialog with initial values. This is needed for initialization of new areas for mass processing.

Z_CLOSE

Closes the current dialog. A confirmation window is optionally possible prior to closing the dialog, in which the end user is asked to confirm whether or not the modifications are to be applied. In this case, all validity checks applicable for the dialog as well as for the access layer will be executed.

Z_CONFIRM

Applies the modifications resulting from a maintain dialog and/or subdialogs to the preliminary copies. Prior to application, any validity checks defined for the dialog are executed. In conjunction with the command Z_REFRESH, it is possible to save the current processing status so that it can be restored if necessary. Z_CONFIRM does not result in any modifications to the original data.

Z_EXIT

Closes the entire application. It also activates a timer which is used to invoke the standard command Z_EXIT_TMR. The combination of these two commands enables a sequential closing of all open dialogs as well as any required inquiries and key ID entries.

Z_HELP

Invokes the Windows help for the current dialog. If the Help ID is not defined, control is given to the content overview. This command has no effect on current processing.

Z_HELPCNTNT

Invokes the content overview of the Windows help. This command has no effect on current processing.

Z_HELPUSE

Invokes help information on how to use Windows help. This command has no effect on current processing.

Z_INFO

Displays the information screen for the application. This command has no effect on current processing.

Z_INFOBUFFER

Opens a dialog displaying existing initialized data information. This command has no effect on actual processing.

Z_INITBUFFER

Causes a new initialization of the command data. This command has no effect on actual processing.

Z_NEXT

Closes the current processing and reads the next selected data record.

Z_OK

Accepts possible modifications and closes the dialog.

Z_OPEN

Starts a new function with the same action and object type. This command has no effect on actual processing.

Z_PREVIOUS

Closes current processing and reads the previous data record from the previously selected set.

Z_READ

Closes the current processing and reads a new data record. Depending on the frame, either the key of the current dialog will be used for reading the data record, or a key ID dialog will be offered.

Z_REFRESH

Removes the modifications resulting from the dialog (and possibly subdialogs). In addition, the dialog elements are provided anew with the values of the preliminary copies. In conjunction with the command Z_CONFIRM, it is possible to save the processing status for possible restore as necessary.

Z_SAVE

Applies the modifications to the original data. Prior to application, all validity checks defined for the dialog as well as for the access layer are executed. If a new record has been added, the key dialog is opened.

Z_SAVEAS

Opens a key dialog for input of a new key ID. The actual storage takes place as a result of an internal command sent from the key dialog to the maintain dialog.

Z_SCRATCH

Deletes the currently displayed data record.

Z_SEARCH

Accepts the user specified selection criteria and starts the corresponding search for data records. Prior to doing so, all entries in the list box are removed.

Internal Standard Commands

Internal standard commands are not defined in the application shell administration system. They are sent to other dialogs during the processing of a local command using the standard event Z_CMD_EXEC. This is required if more than one dialog is used during the processing of a command.

An internal command can also invoke multiple additional internal commands in other dialogs.

- Z_CANCEL_DLG
- Z_CANCEL_KEY
- Z_CANCEL_TMR
- Z_CONFIRM
- Z_CONF_DLG
- Z_DATA_MOD
- Z_ENTER_SUB
- Z_EXIT
- Z_EXIT_TMR
- Z_GET_DATA
- Z_GET_FOCUS
- Z_GET_GLOBAL
- Z_GET_KEY
- Z_INIT
- Z_ITEM_ADD
- Z_ITEM_DEL
- Z_ITEM_MOD
- Z_ITEM_NEXT
- Z_ITEM_PREV
- Z_KEY_MOD
- Z_LB_CLICK
- Z_LB_DOUBLE
- Z_LB_FILL
- Z_LB_SELECT
- Z_LIST_MOD
- Z_MOD_DLG
- Z_NAV_ERR
- Z_NEW_REC
- Z_REFRESH
- Z_RESET_DLG
- Z_SAVEAS
- Z_SELECT_ALL
- Z_START_NKEY
- Z_START_KEY
- Z_START_SAVE
- Z_START_SEL
- Z_START_SOLO

Z_CANCEL_DLG

Informs the maintain dialog that a subordinate subdialog has been closed.

Z_CANCEL_KEY

Informs the dialog that the key dialog was exited with the command Z_CANCEL.

Z_CANCEL_TMR

Is invoked via a timer event. The timer is activated from frames following a runtime error. Causes current processing to be interrupted.

Z_CONFIRM

Informs a subdialog that, based on a command in the maintain dialog, the processing of the local command Z_CONFIRM is to be performed.

Z_CONF_DLG

Informs the maintain dialog that the modifications of a subdialog have been confirmed with the command Z_CONFIRM.

Z_DATA_MOD

Is invoked by a change or click event for a frame-controlled dialog element. Informes the frames that the dialog data has been modified. Causes change events for frame-controlled dialog elements to be suppressed, and the process status of the frame-controlled commands to be activated or deactivated accordingly.

The frames control by default all dialog elements whose change or click event is not suppressed. The subprograms ZXXCTIGN, ZXXCTKYN and ZXXCTMON can be used to notify the frames that dialog elements are to be handled in a nonstandard manner.

If the variable LZ_CHECK_MODIFY is set to FALSE, this command will not be invoked.

Z_ENTER_SUB

Is invoked in a subdialog when the subdialog receives the focus. It ensures that the corresponding maintain dialog is in the background.

Z_EXIT

Corresponds to the local command Z_EXIT.

Z_EXIT_TMR

This internal command is invoked by an activated timer event via the command Z_EXIT. The actual processing corresponds to the local command Z_CLOSE, with the exception that at the end of processing, the command Z_EXIT is sent to the entry dialog of the next function.

Z_GET_DATA

Is sent from a modal window to the higher level dialog. It is used to return the processed data.

Z_GET_FOCUS

Informs the dialog that it should activate itself.

Z_GET_GLOBAL

Results in the updating of the global data in this dialog.

Z_GET_KEY

Is sent from a key dialog to the higher level dialog. It is used to transfer the selected key ID.

Z_INIT

Informs the subdialog that a new process is to be initialized.

Z_ITEM_ADD

Whenever this command is sent from a foreign dialog to a browse dialog, the browse dialog adds the provided data record to the list box. The data are provided to the browse dialog in the array PZ_DATA of the group PZ_LOCAL. The browse dialog determines the structure of the corresponding list box structure by calling subroutines Z_UPDATE_ITEM and Z_PASS_KEY.

If the new data record is within the value range of the currently displayed data records of the list box, it will be added. If not, it will be integrated into the list box provided that the following conditions are met:

End-of-File has been reached and the key of the data record is greater than the highest key in the list box.

The list box is empty and the switch LZ_ADD_ON_EMPTY has been set.

The variable LZ_ADD_EVERY has been set.

Z_ITEM_DEL

Whenever this command is sent from a foreign dialog to a browse dialog, the browse dialog will remove the corresponding data record from the list box. The data are provided in array PZ_DATA of group PZ_LOCAL. The browse dialog determines the structure of the corresponding list box structure by calling subroutines Z_UPDATE_ITEM and Z_PASS_KEY.

Z_ITEM_MOD

Whenever this command is sent from a foreign dialog to a browse dialog, the browse dialog will update the corresponding data record in the list box. The data are provided in array PZ_DATA of group PZ_LOCAL. The browse dialog determines the structure of the corresponding list box structure by calling subroutines Z_UPDATE_ITEM and Z_PASS_KEY.

Z_ITEM_NEXT

Whenever this command is sent from a foreign dialog to a browse dialog, the browse dialog will return the key ID of the next selected list box entry in the variable PZ_LOCAL.PZ_KEY. The calling dialog can determine via the variables PZ_LOCAL.PZ_BOD and PZ_LOCAL.PZ_EOD whether or not additional entries exist before and after the list box entry.

Z_ITEM_PREV

Whenever this command is sent from a foreign dialog to a browse dialog, the browse dialog will return the key ID of the previously selected list box entry in the variable PZ_LOCAL.PZ_KEY. The calling dialog can determine via the variables PZ_LOCAL.PZ_BOD and PZ_LOCAL.PZ_EOD whether or not additional entries exist before and after the list box entry.

Z_KEY_MOD

Is invoked by a change or click event for a frame-controlled dialog element. Informs the frames that the key ID of the dialog has been modified. Causes change events for frame-controlled dialog elements which represent key ID fields to be suppressed, and frame-controlled commands to be activated or deactivated accordingly.

Z_LB_CLICK

Is invoked from a key and browse dialog whenever the user clicks a list box item.

Z_LB_DOUBLE

Is invoked from a key dialog if the user double-clicks on a list box item. The selected list box item is then transferred to the corresponding input dialog element and sent to the higher level dialog.

Z_LB_FILL

Is invoked in a key or browse dialog if the user issues a FILL event for a list box.

Z_LB_SELECT

Is invoked in a key or browse dialog whenever a list box item is selected. For a key dialog, this command causes the data of the selected list box item to be transferred to the input dialog element.

For a browse dialog, the list box selection modification information will be sent to the mass processing dialog (if active). If no mass processing dialog is active and the switch LZ_START_USER_INPUT is not set, the actions of subtype Delete, Display, and Modify will be enabled/disabled depending on whether or not list box entries were selected.

Z_LIST_MOD

Is sent from a listing to the corresponding mass processing dialog. It informs the mass process regarding the modification of the selection set.

Z_MOD_DLG

Is sent from a subdialog to the corresponding maintain dialog, and informs the maintain dialog that the subdialog was modified.

Z_NAV_ERR

The erroneous dialog element or subdialog must be determined. Validation in a high level dialog has resulted in an error condition and the dialog element in which the error was detected is activated.

Z_NEW_REC

Causes a browse dialog to send the key ID of the newly selected data record to the corresponding mass processing dialog.

Z_REFRESH

Corresponds to the local command Z_REFRESH. It is sent from the maintain dialog to its subdialogs whenever the local command Z_REFRESH is invoked and the variable LZ_MAIN_CONFIRM_ALL is set to TRUE.

Z_RESET_DLG

Causes the modifications in a subdialog to be reset same as with the command Z_REFRESH.

Z_SAVEAS

Informs the subdialog that its processing should be switched to a new key.

Z_SELECT_ALL

In a browse dialog, the command causes all list box items to be selected.

Z_START_NKEY

Is provided during the opening of a dialog to indicate that processing is to be started without a key ID.

Z_START_KEY

Is provided during the opening of a dialog to indicate that processing is to be started with a key ID.

Z_START_SAVE

Is provided during the opening of a key dialog to indicate that a new key ID is to be provided.

Z_START_SEL

Is provided during the opening of a key dialog to indicate that an existing key ID is to be selected.

Z_START_SOLO

Is provided during the opening of a mass processing dialog, to indicate that mass processing is to be started independent of a browse dialog.

Tracing a Command

For testing purposes, you can display the current command.

To trace a command

1. Use subroutine Z_INITIALIZE.
2. Move True to LZ_GLOBAL.LZ_CMD_TRACE.
The resulting message box displays the current command.

