

# SAP Textedit



**Release 4.6C**



## Copyright

© Copyright 2001 SAP AG. All rights reserved.

No part of this publication may be reproduced or transmitted in any form or for any purpose without the express permission of SAP AG. The information contained herein may be changed without prior notice.

Some software products marketed by SAP AG and its distributors contain proprietary software components of other software vendors.

Microsoft<sup>®</sup>, WINDOWS<sup>®</sup>, NT<sup>®</sup>, EXCEL<sup>®</sup>, Word<sup>®</sup>, PowerPoint<sup>®</sup> and SQL Server<sup>®</sup> are registered trademarks of Microsoft Corporation.

IBM<sup>®</sup>, DB2<sup>®</sup>, OS/2<sup>®</sup>, DB2/6000<sup>®</sup>, Parallel Sysplex<sup>®</sup>, MVS/ESA<sup>®</sup>, RS/6000<sup>®</sup>, AIX<sup>®</sup>, S/390<sup>®</sup>, AS/400<sup>®</sup>, OS/390<sup>®</sup>, and OS/400<sup>®</sup> are registered trademarks of IBM Corporation.

ORACLE<sup>®</sup> is a registered trademark of ORACLE Corporation.

INFORMIX<sup>®</sup>-OnLine for SAP and Informix<sup>®</sup> Dynamic Server<sup>™</sup> are registered trademarks of Informix Software Incorporated.

UNIX<sup>®</sup>, X/Open<sup>®</sup>, OSF/1<sup>®</sup>, and Motif<sup>®</sup> are registered trademarks of the Open Group.




HTML, DHTML, XML, XHTML are trademarks or registered trademarks of W3C<sup>®</sup>, World Wide Web Consortium, Massachusetts Institute of Technology.

JAVA<sup>®</sup> is a registered trademark of Sun Microsystems, Inc.

JAVASCRIPT<sup>®</sup> is a registered trademark of Sun Microsystems, Inc., used under license for technology invented and implemented by Netscape.

SAP, SAP Logo, R/2, RIVA, R/3, ABAP, SAP ArchiveLink, SAP Business Workflow, WebFlow, SAP EarlyWatch, BAPI, SAPPHIRE, Management Cockpit, mySAP.com Logo and mySAP.com are trademarks or registered trademarks of SAP AG in Germany and in several other countries all over the world. All other products mentioned are trademarks or registered trademarks of their respective companies.

## Icons

Icon	Meaning
	Caution
	Example
	Note
	Recommendation
	Syntax

## Contents

<b>SAP Textedit .....</b>	<b>7</b>
<b>Using Controls in a WAN.....</b>	<b>10</b>
Special Considerations for the SAP Textedit .....	12
<b>Methods.....</b>	<b>12</b>
Alphabetical Listing .....	12
AUTO_REDRAW .....	13
COMMENT_LINES .....	13
COMMENT_SELECTION .....	14
CONSTRUCTOR .....	14
DELETE_TEXT .....	17
EMPTY_UNDO_BUFFER.....	17
FIND_AND_REPLACE .....	18
FIND_AND_SELECT_TEXT.....	19
GET_FIRST_VISIBLE_LINE.....	20
GET_LINE_TEXT.....	20
GET_PATH_OF_DROPPED_FILES .....	21
GET_SELECTED_TEXT_AS_R3TABLE.....	21
GET_SELECTED_TEXT_AS_STREAM.....	22
GET_SELECTION_INDEXES.....	23
GET_SELECTION_POS.....	23
GET_TEXTMODIFIED_STATUS.....	24
GET_TEXT_AS_R3TABLE.....	24
GET_TEXT_AS_STREAM.....	25
GO_TO_LINE.....	27
HIGHLIGHT_BREAKPOINT_LINE .....	27
HIGHLIGHT_LINES .....	28
HIGHLIGHT_SELECTION .....	29
INDENT_LINES.....	29
INDENT_SELECTION .....	30
MAKE_SELECTION_VISIBLE.....	31
OPEN_LOCAL_FILE.....	31
PROTECT_LINES.....	31
PROTECT_SELECTION.....	32
REGISTER_EVENT_CONTEXT_MENU.....	33
REGISTER_EVENT_DBLCLICK.....	34
REGISTER_EVENT_F1.....	35
REGISTER_EVENT_F4.....	36
REGISTER_EVENT_FILEDROP.....	37
REPLACE_ALL .....	38
SAVE_AS_LOCAL_FILE .....	39
SELECT_LINES.....	40
SET_AUTOINDENT_MODE .....	40
SET_COMMENTS_STRING.....	41
SET_DRAGDROP .....	41
SET_FILEDROP_MODE .....	42
SET_FIRST_VISIBLE_LINE .....	43
SET_HIGHLIGHT_BREAKPOINTS_MODE .....	43
SET_HIGHLIGHT_COMMENTS_MODE.....	44

SET_LOCAL_CONTEXTMENU_MODE.....	44
SET_NAVIGATE_ON_DBLCLICK.....	45
SET_READONLY_MODE.....	46
SET_SELECTED_TEXT_AS_R3TABLE.....	46
SET_SELECTED_TEXT_AS_STREAM.....	47
SET_SELECTION_INDEXES.....	48
SET_SELECTION_POS.....	48
SET_SELECTION_POS_IN_LINE.....	49
SET_SPACES_ON_INDENT.....	50
SET_STATUSBAR_MODE.....	50
SET_STATUS_TEXT.....	51
SET_TEXTMODIFIED_STATUS.....	51
SET_TEXT_AS_R3TABLE.....	52
SET_TEXT_AS_STREAM.....	53
SET_TOOLBAR_MODE.....	53
SET_WORDBREAK_PROCEDURE.....	54
SET_WORDWRAP_BEHAVIOR.....	54
UNCOMMENT_LINES.....	55
UNCOMMENT_SELECTION.....	56
UNINDENT_LINES.....	57
UNINDENT_SELECTION.....	57
Functional Listing.....	58
Creating a Control.....	58
Setting and Getting Text.....	59
Setting and Getting Text Positions.....	59
Highlighting and Protecting Text.....	60
Finding and Replacing Text.....	61
Status Bar.....	61
Toolbar.....	61
Other Functions.....	62
<b>Events.....</b>	<b>62</b>
<b>Class Constants.....</b>	<b>62</b>
<b>Instance Attributes.....</b>	<b>63</b>
<b>Keyboard and Mouse Control in the Editor Window.....</b>	<b>66</b>
<b>Methods of Class CL_GUI_CFW.....</b>	<b>70</b>
dispatch.....	70
flush.....	71
get_living_dynpro_controls.....	71
set_new_ok_code.....	72
update_view.....	72
<b>Methods of Class CL_GUI_OBJECT.....</b>	<b>72</b>
is_valid.....	72
free.....	73
<b>Methods of Class CL_GUI_CONTROL.....</b>	<b>73</b>
constructor.....	73
finalize.....	74
set_registered_events.....	74
get_registered_events.....	75
is_alive.....	75

---

set_alignment .....	76
set_position .....	76
set_visible .....	77
get_focus .....	77
set_focus .....	77
get_height .....	78
get_width .....	78
<b>Methods of the Class CL_DRAGDROP .....</b>	<b>78</b>
constructor .....	78
add .....	79
clear .....	80
destroy .....	80
get .....	80
get_handle .....	81
modify .....	82
remove .....	83
<b>Methods of the Class CL_DRAGDROPOBJECT .....</b>	<b>83</b>
set_flavor .....	83
abort .....	84

## SAP Textedit

### Use

You can use the SAP Textedit control to implement an editor for entering and working with text. It can be used as a simple multi-line editor and as an editor for ABAP code. It provides temporary additions that allow you to change the display (for example, to highlight text). Temporary in this sense means that the additions are set and managed at the frontend, but not passed back to the controller. The textedit control does not allow you to set permanent format information or use graphics.

### Features

The SAP Textedit has three parts:

- An application toolbar containing predefined pushbuttons
- The editor window for displaying text
- The status bar, containing the following five fields:
  - Text message display
  - Details of the text currently selected
  - Current cursor position and total number of lines
  - Change status ('\*' = changed, ' ' = unchanged)
  - Insert and overwrite modes ('Ins' and 'Ovr')

The application toolbar and status bar display is optional.

### SAP Textedit



```

program SAPTEXTEDIT_DEMO_3.

*****
* This program demonstrates how to create and use the TextEdit Control.
* It shows how to put and retrieve text via internal tables to the
* control.
* The rudimentary error handling brings up popups. Replace this in
* your application with something more usefull to your context.
*****

call screen 100.

DATA:
*   reference to wrapper class of control based on OO Framework
EDITOR TYPE REF TO CL_GUI_TEXTEDIT,
*   reference to custom container: necessary to bind TextEdit Control
TextEdit_Custom_Container TYPE REF TO CL_GUI_CUSTOM_CONTAINER,
*   other variables
OK_CODE LIKE SY-UCOMM,           " return code from screen
repid like sy-repid.

constants: line_length type i value 256.

* define table type for data exchange
TYPES: begin of mytable_line,
        line(line_length) type C,
        end of mytable_line.

* table to exchange text
data mytable type table of mytable_line.

```

INS Ze 1, Sp 1 Ze 1 - Ze 32 von 220 Zeilen

The control provides the following functions:

- Passing text to and from the control using tables
- Display and change modes
- Insert and overwrite mode
- Setting the maximum line length
- Highlighting (also with prefixes at the beginning of the line) and protection against entry for text areas
- Finding out the current cursor position and the position of a selected area
- Finding out and setting the first line displayed in the editor
- Varying line break behavior
  - Code editor:
    - The line break is line-oriented. You can set the maximum line length yourself. The technical implementation uses internal tables.

## Using Controls in a WAN

- Text editor
  - a) Maximum line length set to width of window
  - b) No maximum line length and no line breaks.
- Local context menu containing the same functions as the toolbar
- Optional toolbar for local operations in the control
- A set of keyboard commands for simple navigation within the text Local operations are simplified by the optional toolbar
- Cut, copy, and paste for selected text areas, using either the keyboard or the application toolbar
- Indentation of selected text blocks
- Import or export local files
- Multi-step undo and redo
- Find and replace with the following options:
  - String or part of string
  - Whole word
  - With or without differentiating between upper- and lowercase
  - Within a set of lines
  - Find next
- Events for double-click, F1, F4, drag and drop, and context menus
- Various drag and drop behaviors for files:
  - Display a file without triggering an event
  - Only a single file can be retrieved using drag and drop, and an event is triggered
  - Multiple files can be retrieved using drag and drop, and an event is triggered
- Status indicating whether a text has been changed or not

The control wrapper is implemented in the global class CL\_GUI\_TEXTEDIT in development class SAPTEXTEDIT.



The development class SAPTEXTEDIT contains example and test programs.

## Using Controls in a WAN

When you use controls in your programs, you place an extra load on the communication channel between the frontend and backend. In a LAN, and particularly in a WAN environment, this can be a critical factor.

The problem is alleviated somewhat by buffering mechanisms (see also [Automation Queue \[Ext.\]](#)). Use these points as a guideline to using controls in a WAN.

## Using Controls in a WAN

The documentation for the individual controls also contains more specific notes about using that control in a WAN.

### Using `CL_GUI_CFW=>FLUSH`

The method `CL_GUI_CFW=>FLUSH` [Page 70] synchronizes the automation queue and the ABAP variables in it. Calling it often generates a synchronous RFC call from the application server to the frontend. To optimize the performance of your application, you should call this method as little as possible.

It is often a good idea to read all control attributes in a single automation queue (for example, at the beginning of the PAI) and retrieve them in a single synchronization. You should, in particular, do this when you read attributes that are not necessary in your event handlers or the PAI/PBO cycle.

You do not need to include a "safety flush" at the end of the PBO to ensure that all method calls are transported to the frontend. A flush at the end of the PBO is guaranteed. Consequently, you cannot construct an automation queue spread over several screens.

**There is no guarantee that an automation queue will be sent when you call `CL_GUI_CFW=>FLUSH`. The queue recognizes whether it contains any return values. If this is not the case, it is not sent.**

If you have a queue with no return values, and want to ensure that it is synchronized, you can use the Control Framework method `CL_GUI_CFW=>UPDATE_VIEW` [Page 71]. You should only use this method if you absolutely need to update the GUI. For example, you might have a long-running application in which you want to provide the user with regular updates on the status of an action.

After you have read the attributes of a control, the contents of the corresponding ABAP variables are not guaranteed until after the next flush. The contents of the ABAP variables remain undefined until this call occurs. In the future, there will be cases in which this flush is unnecessary. They will be recognized by the automation queue and the corresponding flush call will be ignored.

### Creating Controls and Passing Data

Creating controls and passing data to them is normally a one-off procedure, which in comparison to using normal screen elements can be very runtime-intensive. You should therefore not use any unnecessary controls, or pass unnecessary data to the controls that you are using.

A typical example is a tabstrip control with several tab pages. If the pages contain controls, you should consider using application server scrolling instead of local scrolling, and not loading the controls until the corresponding page is activated by the user. The same applies to passing data to the controls on tab pages.

If you want to differentiate between LAN and WAN environments when you pass data to a control, you can use the function module `SAPGUI_GET_WANFLAG`. In some applications, you may need to pass different amounts of data or use a complete fallback in a WAN application. The environment affects, for example, the number of same-level nodes that you can transfer to a tree control without having to introduce artificial intermediate levels.

Unlike screen elements, controls only have to be created and filled with data once. From a performance point of view, this means that they become more profitable the longer they exist. In applications that are called repeatedly, and therefore initialized repeatedly, controls can have a negative effect on performance. In applications that use the same screen for a long time, on the other hand, you may find that using controls results in improved performance.

## Special Considerations for the SAP Textedit

You can always use the [performance tools \[Ext.\]](#) to check the advantages and disadvantages in terms of network load that using a control brings.

## Storing Documents, Picture, and Other Data

Release 4.6A sees the introduction of a frontend cache for accessing documents from the Business Document Service (BDS). You are strongly recommended to store desktop documents, images, and other data in the BDS and not in the R/3 database. Documents from the BDS can be cached at the frontend, and therefore only have to be loaded over the network once.

## Special Considerations for the SAP Textedit

Modified texts are always transferred synchronously to and from the frontend as a whole text. This means that, in a WAN, transfer times may be very long, regardless of the available bandwidth.

The worst-affected methods are those in which you pass tables as parameters. These are:

[SET\\_TEXT\\_AS\\_STREAM \[Page 52\]](#)

[SET\\_SELECTED\\_TEXT\\_AS\\_STREAM \[Page 46\]](#)

[GET\\_TEXT\\_AS\\_STREAM \[Page 24\]](#)

[GET\\_SELECTED\\_TEXT\\_AS\\_STREAM \[Page 21\]](#)

[SET\\_TEXT\\_AS\\_R3TABLE \[Page 51\]](#)

[SET\\_SELECTED\\_TEXT\\_AS\\_R3TABLE \[Page 45\]](#)

[GET\\_TEXT\\_AS\\_R3TABLE \[Page 23\]](#)

[GET\\_SELECTED\\_TEXT\\_AS\\_R3TABLE \[Page 20\]](#)

The SAP Textedit uses an attribute cache, which stores information about whether the text has been modified, and the current line and column position. Reading these attributes therefore does not necessarily involve round trips to the frontend, since the relevant data is often available from the attribute cache. The call [CL\\_GUI\\_CFW=>FLUSH \[Page 70\]](#) is still necessary, however, since you cannot tell as the user when the data from the cache is valid.

## Methods

[Arranged Alphabetically \[Page 11\]](#)

[Arranged by Function \[Page 57\]](#)

## Alphabetical Listing

You requested application help. The following help is available for the current R/3 context:

## AUTO\_REDRAW

### Use

This method uses the administration of the reference counter M\_AUTOREDRAW\_REFCOUNTER (a read-only [instance attribute \[Page 62\]](#)) to determine when the control is redrawn. The control is only redrawn when the value of the reference counter is zero. This allows you to nest calls.


### Features

call method textedit->auto\_redraw

exporting  
 enable\_redraw = enable\_redraw

exceptions  
 error\_cntl\_call\_method = 1.

Parameters	Description	Possible values
enable_redraw	Reference counter	<code>false</code> : Reference counter is increased <code>true</code> : Reference counter is decreased

 You should only use this method when absolutely necessary. If another window is open over the text window and you set the focus to the text window, parts of the other window will appear in the text window. The text will not be displayed properly until the reference counter reaches zero.

## COMMENT\_LINES

### Use

Use this method to designate a set of lines as comment lines.

### Features

call method textedit->comment\_lines

exporting  
 from\_line = from\_line  
 to\_line = to\_line  
 enable\_editing\_protected\_text = enable\_editing\_protected\_text

exceptions  
 error\_cntl\_call\_method = 1.

Parameters	Description	Possible values
from_line	First line of block	
to_line	End of block	

**COMMENT\_SELECTION**

enable_editing_protected_text	Editing for protected sections of text	<b>false</b> Protected sections cannot be edited (default value)  <b>true</b> Protected sections can be edited
-------------------------------	--	--

**COMMENT\_SELECTION****Use**

Use this method to make the selected lines into comment lines. The starting and finishing lines are always converted entirely to comment lines, even if the user did not select all of them.

**Features**

call method textedit->comment\_selection

exporting

enable\_editing\_protected\_text = enable\_editing\_protected\_text

exceptions

error\_cntl\_call\_method = 1.

Parameters	Description	Possible values
enable_editing_protected_text	Editing for protected sections of text	<b>false</b> Protected sections cannot be edited (default value)  <b>true</b> Protected sections can be edited

**CONSTRUCTOR****Use**

The constructor creates, initializes, and positions the control.

**CONSTRUCTOR**



The constructor method is called automatically when you instantiate the class (`create object` statement). You generally pass the parameters of the method in the `create object` statement.

**Features**

create object textedit

exporting

parent = parent

lifetime = lifetime

max\_number\_chars = max\_number\_chars

style = style

filedrop\_mode = filedrop\_mode

wordwrap\_mode = wordwrap\_mode

wordwrap\_position = wordwrap\_position

wordwrap\_to\_linebreak\_mode = wordwrap\_to\_linebreak\_mode

exceptions

error\_cntl\_create = 1

error\_cntl\_init = 2


error\_cntl\_link = 3

error\_dp\_create = 4

gui\_type\_not\_supported = 5.

Parameters	Description	Possible values
parent	Parent of the instance, that is, the contain in which the control is to be displayed	
lifetime	Lifetime management parameter specifying the lifetime of the control	<p><b>cntl_lifetime_imode:</b> The control remains alive for the lifetime of the internal session (that is, until a statement such as <code>leave program</code> or <code>leave to transaction</code>)</p> <p><b>cntl_lifetime_dynpro:</b> The control remains alive for the lifetime of the screen (that is, while it remains in the screen stack). It is not destroyed, for example, by a <code>call screen</code> or <code>call transaction</code> statement.</p>
max_number_chars	Sets the maximum number of characters that can be entered in the control	

## CONSTRUCTOR

style	Controls the appearance and behavior of the control 	Constants from the class CL_GUI_CONTROL that begin with WS_* You can combine styles by adding the constants together. The default value sets a suitable combination of style constants internally.
filedrop_mode	Parameter controlling drag and drop behavior	<p><b>dropfile_event_off</b> Only a file that replaces the full existing text can be used (default value).</p> <p><b>dropfile_event_single</b> Only one file can be used. The file is not included in the editor. The file drop event is triggered instead. You can specify the path of the dropped file using the method <a href="#">GET_PATH_OF_DROPPED_FILES [Page 20]</a>.</p> <p><b>dropfile_event_multiple</b> You can use more than one file. The file is not included in the editor. The file drop event is triggered instead. You can specify a list of the paths of the dropped files using the method <a href="#">GET_PATH_OF_DROPPED_FILES [Page 20]</a>.</p>
wordwrap_mode	Line break behavior	<p><b>wordwrap_off</b> No line break</p> <p><b>wordwrap_at_windowborder</b> Line break at the edge of the window (default value)</p> <p><b>wordwrap_at_fixed_position</b> Line break at a fixed position For further information, refer to <a href="#">Class Constants [Page 61]</a></p>

## DELETE\_TEXT

wordwrap_position	If <b>wordwrap_mode = textedit-&gt;wordwrap_at_fixed_position</b> : Position for the automatic line break in a line	Default value: -1
wordwrap_to_linebreak_mode	Converts soft line breaks to hard ones when you save in the R/3 System	<b>false</b> Soft line breaks are ignored when you save <b>true</b> Soft line breaks are converted to hard breaks when you save



The advantage of using containers is that you can set certain essential display attributes (position on the screen, behavior when the window is resized, and so on) when you create them. If you do not use a container, you must set all of the attributes by hand, using the method [SET\\_WINDOW\\_PROPERTY \[Ext.\]](#).

## DELETE\_TEXT

### Use

This method deletes all of the text in the control.

### Features

call method textedit->delete\_text

exceptions  
error\_cntl\_call\_method = 1.

## EMPTY\_UNDO\_BUFFER

### Use

Use this method to empty the undo buffer of the control. After the method has been executed, no previous actions can be undone (including the method call itself). The undo buffer is built up again by subsequent actions in the control. The undo buffer is also implicitly deleted when you protect text areas against input using the [PROTECT\\_LINES \[Page 30\]](#) and [PROTECT\\_SELECTION \[Page 31\]](#) methods.

### Features

call method textedit->empty\_undo\_buffer

**FIND\_AND\_REPLACE**

exceptions  
 error\_cntl\_call\_method = 1.

**FIND\_AND\_REPLACE****Use**

This method allows you to find and replace text.

**Features**

call method textedit->find\_and\_replace

exporting  
 case\_sensitive\_mode = case\_sensitive\_mode  
 replace\_string = replace\_string  
 search\_string = search\_string  
 whole\_word\_mode = whole\_word\_mode

changing  
 string\_found = string\_found

exceptions  
 invalid\_parameter = 1  
 error\_cntl\_call\_method = 2.

Parameters	Description	Possible values
case_sensitive_mode	Upper- and lowercase	<b>true</b> Observe  <b>false</b> Do not observe (default value)
replace_string	Text to replace occurrences of the search string	
search_string	Text to be found/replaced	
whole_word_mode	Only find whole words	<b>true</b> Only find whole words  <b>false</b> Find whole words and parts of words (default value)
string_found	Return value specifying how often the text was found	



If you do not specify REPLACE\_STRING or SEARCH\_STRING, the system calls the *Find and Replace* dialog box.

**FIND\_AND\_SELECT\_TEXT**

If no text is selected, the search starts at the current cursor position. If text is selected, the search starts at the second character of the selection, but is not restricted to the selected text.

When it reaches the end of the text, the method returns to the beginning of the text. The text may thus be searched more than once.



If you want to replace all occurrences of a search string in a text, use the method [REPLACE\\_ALL \[Page 37\]](#).

## FIND\_AND\_SELECT\_TEXT

### Use

This method allows you to find and select text.

### Features

call method textedit->find\_and\_select\_text

```
exporting
  case_sensitive_mode = case_sensitive_mode
  search_string = search_string
  whole_word_mode = whole_word_mode
```

```
changing
  string_found = string_found
```

```
exceptions
  invalid_parameter = 1
  error_cntl_call_method = 2.
```

Parameters	Description	Possible values
case_sensitive_mode	Upper- and lowercase	<b>true</b> Observe <b>false</b> Do not observe (default value)
search_string	Text to be found/replaced	
whole_word_mode	Only find whole words	<b>true</b> Only find whole words <b>false</b> Find whole words and parts of words (default value)
string_found	Return value specifying how often the text was found	

**GET\_FIRST\_VISIBLE\_LINE**

If you do not specify the SEARCH\_STRING parameter, the system calls the *Find* dialog box.

If no text is selected, the search starts at the current cursor position. If text is selected, the search starts at the second character of the selection, but is not restricted to the selected text.

When it reaches the end of the text, the method returns to the beginning of the text. The text may thus be searched more than once.

**GET\_FIRST\_VISIBLE\_LINE****Use**

This method returns the line number of the topmost visible line in the control.

**Features**

call method textedit->get\_first\_visible\_line

importing  
line = line

exceptions  
error\_cntl\_call\_method = 1.

Parameters	Description
line	Line number. The line numbering begins at one.

**GET\_LINE\_TEXT****Use**

This method returns the text in line number LINE\_NUMBER.

**Features**

call method textedit->get\_line\_text

exporting  
line\_number = line\_number

importing  
text = text

exceptions  
invalid\_parameter = 1  
error\_cntl\_call\_method = 2.

Parameters	Description
------------	-------------

GET\_PATH\_OF\_DROPPED\_FILES

line_number	Line number. The line numbering begins at one.
text	Text in the line



Due to restrictions on communication between the frontend and backend, texts can only be transferred with a length of up to 256 characters.



Workaround for texts longer than 256 characters:

Select a whole line and use the method [GET\\_SELECTED\\_TEXT\\_AS\\_R3TABLE \[Page 20\]](#) or [GET\\_SELECTED\\_TEXT\\_AS\\_STREAM \[Page 21\]](#) .

## GET\_PATH\_OF\_DROPPED\_FILES

### Use

This method returns a list of the paths of files used in drag and drop. The list is updated in every drag and drop operation. To set various drag and drop attributes, use the method [SET\\_FILEDROP\\_MODE \[Page 41\]](#).

### Features

call method textedit->get\_path\_of\_dropped\_files

exporting  
table = table

exceptions  
error\_dp = 1.

Parameters	Description	Possible values
table	R/3 table containing the paths	

## GET\_SELECTED\_TEXT\_AS\_R3TABLE

### Use

This method allows you to save the selected text as an R/3 table (without information about the line breaks). This depends on the behavior of the line breaks that you have defined in the [CONSTRUCTOR \[Page 13\]](#), especially for soft carriage returns.

### Features

call method textedit->get\_selected\_text\_as\_r3table

**GET\_SELECTED\_TEXT\_AS\_STREAM**

```

importing
  table = table

exceptions
  error_dp = 1
  potential_data_loss = 2.

```

Parameters	Description
table	R/3 table



If you use a table with shorter lines than those in the SAP Textedit, the text from the control is only passed up to the length of the table line. The remaining text is lost.

If the lines of the table are longer than the lines of the control, the table lines are filled with trailing spaces.

**GET\_SELECTED\_TEXT\_AS\_STREAM****Use**

This method allows you to place the entire text from the editor in an R/3 table as a stream (with information about line breaks).

**Features**

call method textedit->get\_selected\_text\_as\_stream

```

importing
  selected_text = selected_text

exceptions
  error_dp = 1.

```

Parameters	Description
selected_text	R/3 table



For an example of how to work with line break information, refer to the demonstration program SAPTEXTEDIT\_TEST\_1, include SAPTEXTEDIT\_TEST\_1F01, subroutine FIND\_CR, which replaces the carriage return line feeds.

## GET\_SELECTION\_INDEXES

### Use

This method returns information (in characters) about the position of a selected text.

### Features

call method textedit->get\_selection\_indexes

```
importing
  from_index = from_index
  to_index = to_index
```

```
exceptions
  error_cntl_call_method = 1.
```

Parameters	Description
from_index	Character at which the selection starts. The index is relative to the first character in the editor.
to_index	Character at which the selection ends.

## GET\_SELECTION\_POS

### Use

This method returns the position of a selected text.

### Features

call method textedit->get\_selection\_pos

```
importing
  from_line = from_line
  from_pos = from_pos
  to_line = to_line
  to_pos = to_pos
```

```
exceptions
  error_cntl_call_method = 1.
```

Parameters	Description
from_line	Line in which the selection starts
from_pos	Position in the line at which the selection starts
to_line	Line in which the selection ends
to_pos	Position in the line at which the selection ends

## GET\_TEXTMODIFIED\_STATUS

## GET\_TEXTMODIFIED\_STATUS

### Use

This method returns the change status of the text. You can use the method [SET\\_TEXTMODIFIED\\_STATUS \[Page 50\]](#) to set the changed status for a text.



If you get text using the methods [GET\\_TEXT\\_AS\\_R3TABLE \[Page 23\]](#) and [GET\\_TEXT\\_AS\\_STREAM \[Page 24\]](#), the changed status is not reset.

### Features

call method textedit->get\_textmodified\_status

importing  
status = status

exceptions  
error\_cntl\_call\_method = 1.

Parameters	Description	Possible values
status	Changed status	<b>true</b> Text changed <b>false</b> Text not changed

## GET\_TEXT\_AS\_R3TABLE

### Use

This method allows you to save the entire text from the editor as an R/3 table (without information about line feeds). This depends on the behavior of the line breaks that you have defined in the [CONSTRUCTOR \[Page 13\]](#), especially for soft carriage returns.

### Features

call method textedit->get\_text\_as\_r3table

exporting  
only\_when\_modified = only\_when\_modified

importing  
table = table  
is\_modified = is\_modified

exceptions  
error\_db = 1  
error\_cntl\_call\_method = 2

GET\_TEXT\_AS\_STREAM

error\_db\_create =3  
potential\_data\_loss = 4.

Parameters	Description	Possible values
table	R/3 table	
only_when_modified	Attribute when you get text	<p><b>true</b> The text is only written to the R/3 table if the changed status IS:MODIFIED = TRUE. Otherwise, an empty table is passed.</p> <p><b>false</b> The text is retrieved regardless of whether it has been changed (default value)</p>
is_modified	Changed status of the text	<p><b>true</b> Text changed</p> <p><b>false</b> Text not changed</p>



If you use a table with shorter lines than those in the SAP Textedit, the text from the control is only passed up to the length of the table line. The remaining text is lost.

If the lines of the table are longer than the lines of the control, the table lines are filled with trailing spaces.



Setting the parameter ONLY\_WHEN\_MODIFIED to TRUE can improve performance, since it minimizes the data exchange between the frontend and backend (and the database).

## GET\_TEXT\_AS\_STREAM

### Use

This method allows you to place the entire text from the editor in an R/3 table as a stream (with information about line breaks).

**GET\_TEXT\_AS\_STREAM****Features**

call method textedit->get\_text\_as\_r3table

exporting

only\_when\_modified = only\_when\_modified

importing

text = text

is\_modified = is\_modified

exceptions

error\_db = 1

error\_cntl\_call\_method = 2.

Parameters	Description	Possible values
text	R/3 table with text	
only_when_modified	Attribute when you get text	<p><b>true</b> The text is only written to the R/3 table if the changed status IS:MODIFIED = TRUE. Otherwise, an empty table is passed.</p> <p><b>false</b> The text is retrieved regardless of whether it has been changed (default value)</p>
is_modified	Changed status of the text	<p><b>true</b> Text changed</p> <p><b>false</b> Text not changed</p>



Setting the parameter ONLY\_WHEN\_MODIFIED to TRUE can improve performance, since it minimizes the data exchange between the frontend and backend (and the database).



For an example of how to work with line break information, refer to the demonstration program SAPTEXTEDIT\_TEST\_1, include SAPTEXTEDIT\_TEST\_1F01, subroutine FIND\_CR, which replaces the carriage return line feeds.

## GO\_TO\_LINE

### Use

Use this method to navigate the cursor to the specified line. The cursor position within the line is the same as it was before the call.

### Features

call method textedit->go\_to\_line

exporting  
line = line

exceptions  
error\_cntl\_call\_method = 1.

Parameters	Description
line	Line to which you want to navigate

## HIGHLIGHT\_BREAKPOINT\_LINE

### Use

Use this method to set a breakpoint.



This method is only used in the ABAP Editor.

### Features

call method textedit->highlight\_breakpoint\_line

exporting  
line = line  
highlight\_mode = highlight\_mode

exceptions  
has\_no\_effect = 1  
error\_cntl\_call\_method = 2  
invalid\_parameter = 3.

Parameters	Description	Possible values
line	Line in which you want to set the breakpoint	

**HIGHLIGHT\_LINES**

highlight_mode	Switches highlighting on or off (true/false)	<b>true</b> Switched on (default value) <b>false</b> Switched off
----------------	--	---

**HIGHLIGHT\_LINES****Use**

Use this method to switch the highlighting mode on and off for a range of lines.

**Features**

call method textedit->highlight\_lines

```
exporting
  from_line = from_line
  to_line = to_line
  highlight_mode = highlight_mode
```

```
exceptions
  has_no_effect = 1
  error_cntl_call_method = 2
  invalid_parameter = 3.
```

Parameters	Description	Possible values
from_line	Starting line for the range	
to_line	Final line (inclusive) of the range	
highlight_mode	Switches highlighting on or off (true/false)	<b>true</b> Switched on (default value) <b>false</b> Switched off



If you used the method [SET\\_HIGHLIGHT\\_COMMENTS\\_MODE \[Page 43\]](#) to highlight all comment lines automatically, the method HIGHLIGHT\_LINES has no effect.

## HIGHLIGHT\_SELECTION

### Use

This method allows you to switch the highlighting mode for a selection of lines on and off.

### Features

call method textedit->highlight\_selection

exporting  
highlight\_mode = highlight\_mode

exceptions  
has\_no\_effect = 1  
error\_cntl\_call\_method = 2  
invalid\_parameter = 3.

Parameters	Description	Possible values
highlight_mode	Switches highlighting on or off (true/false)	<b>true</b> Switched on (default value) <b>false</b> Switched off



If you used the method [SET HIGHLIGHT COMMENTS MODE \[Page 43\]](#) to highlight all comment lines automatically, the method HIGHLIGHT\_SELECTION has no effect.

## INDENT\_LINES

### Use

Use this method to indent a set of lines by a given number of spaces. Use the parameter M\_SPACES\_ON\_INDENT of the method [SET SPACES ON INDENT \[Page 49\]](#) to set the number of spaces.

### Features

call method textedit->indent\_lines

exporting  
from\_line = from\_line  
to\_line = to\_line  
enable\_editing\_protected\_text = enable\_editing\_protected\_text

exceptions  
error\_cntl\_call\_method = 1.

**INDENT\_SELECTION**

Parameters	Description	Possible values
from_line	Beginning of the section you want to indent	
to_line	End of the section you want to indent	
enable_editing_protected_text	Editing for protected sections of text	<b>false</b> Protected sections cannot be edited (default value)  <b>true</b> Protected sections can be edited

**INDENT\_SELECTION****Use**

Use this method to indent a selected area by a given number of spaces. Use the parameter `M_SPACES_ON_INDENT` of the method [SET\\_SPACES\\_ON\\_INDENT \[Page 49\]](#) to set the number of spaces.



Only the text that is actually selected is indented (important if you did not select from the beginning of a line or to the end of a line).

**Features**

call method textedit->indent\_selection

exporting

enable\_editing\_protected\_text = enable\_editing\_protected\_text

exceptions

error\_cntl\_call\_method = 1.

Parameters	Description	Possible values
enable_editing_protected_text	Editing for protected sections of text	<b>false</b> Protected sections cannot be edited (default value)  <b>true</b> Protected sections can be edited

## MAKE\_SELECTION\_VISIBLE

### Use

This method allows you to move a selected area into the visible part of the editor window.

### Features

call method textedit->make\_selection\_visible

exceptions  
error\_cntl\_call\_method = 1.

## OPEN\_LOCAL\_FILE

### Use

This method allows you to open a local file.

### Features

call method textedit->open\_local\_file

exporting  
file\_name = file\_name

exceptions  
error\_cntl\_call\_method = 1.

Parameters	Description
file_name	Name of the file you want to open. You can specify the path in UNC format.



If you do not set the FILE\_NAME parameter, a dialog box appears in which you can enter the name and path of the file.

## PROTECT\_LINES

### Use

You can protect ranges of lines against changes. Use this method to switch the protection on and off.

**PROTECT\_SELECTION**

When a line is protected, its background color changes. If you try to enter text in a protected line, a warning signal sounds.

**Features**

call method textedit->protect\_lines

```
exporting
  from_line = from_line
  to_line = to_line
  protect_mode = protect_mode
  enable_editing_protected_text = enable_editing_protected_text
```

```
exceptions
  error_cntl_call_method = 1
  invalid_parameter = 2.
```

Parameters	Description	Possible values
from_line	Starting line for the range	
to_line	Final line (inclusive) of the range	
protect_mode	Switches input protection on or off	<b>true</b> Protection switched on (default value) <b>false</b> Protection switched off (default value)
enable_editing_protected_text	Editing for protected sections of text	<b>false</b> Protected sections cannot be edited (default value) <b>true</b> Protected sections can be edited

**PROTECT\_SELECTION****Use**

You can protect selections against changes. Use this method to switch the protection on and off.

When a selection is protected, its background color changes. If you try to enter text in a protected line, a warning signal sounds.

REGISTER\_EVENT\_CONTEXT\_MENU

**Features**

call method textedit->protect\_selection

exporting  
 protect\_mode = protect\_mode

exceptions  
 error\_cntl\_call\_method = 1  
 invalid\_parameter = 2.

Parameters	Description	Possible values
protect_mode	Switches input protection on or off	<b>true</b> Protection switched on (default value)  <b>false</b> Protection switched off (default value)
enable_editing_protected_text	Editing for protected sections of text	<b>false</b> Protected sections cannot be edited (default value)  <b>true</b> Protected sections can be edited

**REGISTER\_EVENT\_CONTEXT\_MENU**

**Use**

Use this method to register context menu event.



To make your coding easier to understand, you should call this method directly after you have created the instance (**CREATE OBJECT**).

**Features**

call method textedit->register\_event\_context\_menu

exporting  
 register = register  
 appl\_event = appl\_event  
 local\_entries = local\_entries

**REGISTER\_EVENT\_DBLCLICK**

exceptions  
 error\_regist\_event = 1  
 error\_unregist\_event = 2  
 cntl\_error = 3  
 event\_already\_registered = 4  
 event\_not\_registered = 5.

Parameters	Description	Possible values
register	Registering the events	<b>true</b> Register events (default value) <b>false</b> Do not register events
appl_event	Choice between system and application events	<b>space</b> System event (default value) <b>'x'</b> Application event
local_entries	Displays the local entries in the context menu	<b>true</b> Display entries (default value) <b>false</b> Do not display entries

**REGISTER\_EVENT\_DBLCLICK****Use**

Use this method to register the event [DBLCLICK \[Ext.\]](#) (double click).



To make your coding easier to understand, you should call this method directly after you have created the instance (**CREATE OBJECT**).

**Features**

call method textedit->register\_event\_dbclick

exporting  
 register = register  
 appl\_event = appl\_event  
 navigate\_on\_dbclick = navigate\_on\_dbclick

REGISTER\_EVENT\_F1

exceptions  
 error\_regist\_event = 1  
 error\_unregist\_event = 2  
 cntl\_error = 3  
 event\_already\_registered = 4  
 event\_not\_registered = 5.

Parameters	Description	Possible values
register	Registering the events	<b>true</b> Register events (default value) <b>false</b> Do not register events
appl_event	Choice between system and application events	<b>space</b> System event (default value) <b>'x'</b> Application event
navigate_on_dbclick	Local navigation	<b>false</b> No local navigation (default value) <b>true</b> local navigation

## REGISTER\_EVENT\_F1

### Use

Use this method to register the F1 (function key) event.



To make your coding easier to understand, you should call this method directly after you have created the instance (**CREATE OBJECT**).

### Features

call method textedit->register\_event\_f1

exporting  
 register = register  
 appl\_event = appl\_event

**REGISTER\_EVENT\_F4**

```

exceptions
  error_regist_event = 1
  error_unregist_event = 2
  cntl_error = 3
  event_already_registered = 4
  event_not_registered = 5.

```

Parameters	Description	Possible values
register	Registering the events	<b>true</b> Register events (default value) <b>false</b> Do not register events
appl_event	Choice between system and application events	<b>space</b> System event (default value) <b>'x'</b> Application event

**REGISTER\_EVENT\_F4****Use**

Use this method to register the F4 (function key) event.



To make your coding easier to understand, you should call this method directly after you have created the instance (**CREATE OBJECT**).

**Features**

call method textedit->register\_event\_f4

```

exporting
  register = register
  appl_event = appl_event

```

```

exceptions
  error_regist_event = 1
  error_unregist_event = 2
  cntl_error = 3
  event_already_registered = 4
  event_not_registered = 5.

```

Parameters	Description	Possible values
------------	-------------	-----------------

**REGISTER\_EVENT\_FILEDROP**

register	Registering the events	<b>true</b> Register events (default value) <b>false</b> Do not register events
appl_event	Choice between system and application events	<b>space</b> System event (default value) <b>'x'</b> Application event

## REGISTER\_EVENT\_FILEDROP

### Use

Use this method to register the filedrop event.



To make your coding easier to understand, you should call this method directly after you have created the instance (**CREATE OBJECT**).

### Features

call method textedit->register\_event\_filedrop

exporting

register = register

appl\_event = appl\_event

exceptions

error\_regist\_event = 1

error\_unregist\_event = 2

cntl\_error = 3

event\_already\_registered = 4

event\_not\_registered = 5.

Parameters	Description	Possible values
register	Registering the events	<b>true</b> Register events (default value) <b>false</b> Do not register events

**REPLACE\_ALL**

appl_event	Choice between system and application events	<b>space</b> System event (default value)  <b>'x'</b> Application event
------------	--	--

**REPLACE\_ALL****Use**

This method allows you to replace found text.

**Features**

call method textedit->replace\_all

exporting

case\_sensitive\_mode = case\_sensitive\_mode

replace\_string = replace\_string

search\_string = search\_string

whole\_word\_mode = whole\_word\_mode

changing

counter = counter

exceptions


error\_cntl\_call\_method = 1

invalid\_parameter = 2.


Parameters	Description	Possible values
case_sensitive_mode	Upper-/lowercase	<b>false</b> Do not observe (default value)  <b>true</b> Observe
replace_string	Text to replace the occurrences of SEARCH_STRING	
search_string	Text to be replaced	

**SAVE\_AS\_LOCAL\_FILE**

whole_word_mode	Only replace whole words	<b>false</b> Find whole words and parts of words (default value)  <b>true</b> Only find whole words
counter	Return value specifying how many times the search string was replaced	

 If you do not set the parameters SEARCH\_STRING and REPLACE\_STRING, the *Find and Replace* dialog box appears.

The method starts at the cursor position and searches to the end of the text. You can use the methods [SET\\_SELECTION\\_INDEXES \[Page 47\]](#), [SET\\_SELECTION\\_POS \[Page 47\]](#), or [SET\\_SELECTION\\_POS\\_IN\\_LINE \[Page 48\]](#) to set the cursor position. If you want to replace all occurrences within a text, you should set the selection to the first character in the text.

 Text replacement is not restricted to the selected area.

## SAVE\_AS\_LOCAL\_FILE


### Use

Use this method to save text in a local file.

### Features

```
call method textedit->save_as_local_file
    exporting
        file_name = file_name
    exceptions
        error_cntl_call_method = 1.
```

Parameters	Description
file_name	Name of the file you want to save. You can specify the path in UNC format.

 If you do not set the FILE\_NAME parameter, a dialog box appears in which you can enter the name and path of the file.

**SELECT\_LINES****SELECT\_LINES****Use**

Use this method to select a range of lines. The range is automatically placed within the visible section of the control.

**Features**

call method textedit->select\_lines

exporting

from\_line = from\_line

to\_line = to\_line

exceptions

error\_cntl\_call\_method = 1.

Parameters	Description
from_line	Starting line for the range
to_line	Final line (inclusive) of the range

You can make the selection invisible to the user as follows:

1. Get the current cursor position
2. Switch off the redraw function by calling the method [AUTO\\_REDRAW \[Page 12\]](#) and setting the ENABLE\_REDRAW parameter to false.
3. Set the required selection. Other actions may follow at this point.
4. If required, set the new position.
5. Reenable the automatic redraw by calling the method [AUTO\\_REDRAW \[Page 12\]](#) and setting the ENABLE\_REDRAW parameter to true.

**SET\_AUTOINDENT\_MODE****Use**

Use this method to switch the automatic indentation mode on or off.

For further information about indenting lines, refer to [INDENT\\_LINES \[Page 28\]](#), [UNINDENT\\_LINES \[Page 55\]](#), [INDENT\\_SELECTION \[Page 29\]](#), and [UNINDENT\\_SELECTION \[Page 56\]](#).

**Features**

call method textedit->set\_autoindent\_mode

exporting

auto\_indent = auto\_indent

exceptions

error\_cntl\_call\_method = 1.

**SET\_COMMENTS\_STRING**

Parameters	Description	Possible values
auto_indent	Switches the automatic mode for line indentation on or off	<b>true</b> Switch on  <b>false</b> Switch off

## SET\_COMMENTS\_STRING

### Use

Use this method to comment out lines beginning with a particular string.



In the ABAP Editor, the comment character is the asterisk (\*).

To switch highlighting on or off for comment lines, use the method [SET\\_HIGHLIGHT\\_COMMENTS\\_MODE \[Page 43\]](#).

### Features

call method textedit->set\_comments\_string

```
exporting
  comments_string = comments_string
```

```
exceptions
  error_cntl_call_method = 1.
```

Parameters	Description
comments_string	String

## SET\_DRAGDROP

### Use

Use this method to set the drag and drop behavior.

### Features

call method textedit->set\_dragdrop

```
exporting
  dragdrop = dragdrop.
```

**SET\_FILEDROP\_MODE**

Parameters	Description
dragdrop	Drag and drop object with type CL_DRAGDROP. You must first create and initialize this object. For further information, refer to <a href="#">Drag and Drop [Ext.]</a>

## SET\_FILEDROP\_MODE

### Use

Use this method to set the mode used in drag and drop operations with files.



You can also set the mode in the [CONSTRUCTOR \[Page 13\]](#) method when you create an instance.

### Features

call method textedit->set\_filedrop\_mode

exporting

filedrop\_mode = filedrop\_mode

exceptions

error\_cntl\_call\_method = 1

invalid\_parameter = 2.

Parameters	Description	Possible values
filedrop_mode	Parameter controlling drag and drop behavior	<p><b>dropfile_event_off</b> Only a file that replaces the full existing text can be used (default value)</p> <p><b>dropfile_event_single</b> Only one file can be used. The file is not included in the editor. The file drop event is triggered instead. You can specify the path of the dropped file using the method <a href="#">GET_PATH_OF_DROPPED_FILES [Page 20]</a>.</p> <p><b>dropfile_event_multiple</b> You can use more than one file. The file is not included in the editor. The file drop event is triggered instead. You can specify a list of the paths of the dropped files using the method <a href="#">GET_PATH_OF_DROPPED_FILES [Page 20]</a>.</p>

## SET\_FIRST\_VISIBLE\_LINE

### Use

Use this method to set the specified line as the first visible line in the editor window.

### Features

call method textedit->set\_first\_visible\_line

exporting  
line = line

exceptions  
error\_cntl\_call\_method = 1.

Parameters	Description
line	Line number

## SET\_HIGHLIGHT\_BREAKPOINTS\_MODE

### Use

This method allows you to switch the highlighting mode for breakpoints on and off.



This method is only used in the ABAP Editor.

### Features

call method textedit->set\_highlight\_breakpoints\_mode

exporting  
highlight\_breakpoints\_mode = highlight\_breakpoints\_mode

exceptions  
error\_cntl\_call\_method = 1  
invalid\_parameter = 2.

Parameters	Description	Possible values
highlight_breakpoints_mode	Switches breakpoint highlighting on or off	<b>true</b> Switch on (default value) <b>false</b> Switch off

**SET\_HIGHLIGHT\_COMMENTS\_MODE**

If this mode is switched on, the methods [HIGHLIGHT\\_SELECTION \[Page 27\]](#) and [HIGHLIGHT\\_LINES \[Page 27\]](#) for highlighting areas of text have no effect.

**SET\_HIGHLIGHT\_COMMENTS\_MODE****Use**

Use this method to switch automatic highlighting for comment lines on or off. The method [SET\\_COMMENTS\\_STRING \[Page 40\]](#) determines the lines that are recognized as comments.

**Features**

call method textedit->set\_highlight\_comments\_mode

```
exporting
  highlight_comments_mode = highlight_comments_mode
```

```
exceptions
  error_cntl_call_method = 1
  invalid_parameter = 2.
```

Parameters	Description	Possible values
highlight_comments_mode	Switches automatic highlighting on or off	<b>true</b> Switch on (default value) <b>false</b> Switch off



If this mode is switched on, the methods [HIGHLIGHT\\_SELECTION \[Page 27\]](#) and [HIGHLIGHT\\_LINES \[Page 27\]](#) for highlighting areas of text have no effect.

**SET\_LOCAL\_CONTEXTMENU\_MODE****Use**

Use this method to make the local context menu visible or invisible.

SET\_NAVIGATE\_ON\_DBLCLICK

**Features**

call method textedit->set\_local\_contextmenu\_mode

exporting  
 visible = visible

exceptions  
 error\_cntl\_call\_method = 1  
 invalid\_parameter = 2.

Parameters	Description	Possible values
visible	Visibility of local context menu	<b>false</b> Context menu invisible (default)  <b>true</b> Context menu visible

**SET\_NAVIGATE\_ON\_DBLCLICK**

**Use**

Use this method to set the reaction to a double-click in the control.

**Features**

call method textedit->set\_navigate\_on\_dblick

exporting  
 navigate\_on\_dblick\_mode = navigate\_on\_dblick\_mode

exceptions  
 error\_cntl\_call\_method = 1  
 invalid\_parameter = 2.

Parameters	Description	Possible values
navigate_on_dblick_mode	Mode for reacting to a double-click	<b>true</b> Forward navigation switched on (default value)  Forward navigation switched off

## SET\_READONLY\_MODE

**SET\_READONLY\_MODE****Use**

Use this method to switch between read mode and input mode in the textedit control.

You can tell the difference between the two modes from the background colors. The colors are taken from the general SAPgui settings.

**Features**

call method textedit->set\_readonly\_mode

exporting  
read\_only = read\_only

exceptions  
error\_cntl\_call\_method = 1  
invalid\_parameter = 2.

Parameters	Description	Possible values
read_only	Input mode for the SAP Textedit	<b>true</b> Input not possible (default value) <b>false</b> Input possible

**SET\_SELECTED\_TEXT\_AS\_R3TABLE****Use**

This method allows you to insert text from an R/3 table as a stream. The text is inserted at the cursor position. Any selected text is overwritten. The text is passed without information about line breaks.

**Features**

call method textedit->set\_selected\_text\_as\_r3table

exporting  
table = table  
enable\_editing\_protected\_text = enable\_editing\_protected\_text

exceptions  
error\_dp = 1  
error\_db\_create = 2.

Parameters	Description	Possible values
table	R/3 table with text	

**SET\_SELECTED\_TEXT\_AS\_STREAM**

enable_editing_protected_text	Editing for protected sections of text	<b>false</b> Protected sections cannot be edited (default value)  <b>true</b> Protected sections can be edited
-------------------------------	--	--



If you use a table with shorter lines than those in the SAP Textedit, the text from the control is only passed up to the length of the table line. The remaining text is lost.

If the lines of the table are longer than the lines of the control, the table lines are filled with trailing spaces.

## SET\_SELECTED\_TEXT\_AS\_STREAM

### Use

This method allows you to insert text from an R/3 table as a stream. The text is inserted at the cursor position. Any selected text is overwritten. The text is passed with information about line breaks.

### Features

call method textedit->set\_selected\_text\_as\_stream

```
exporting
  selected_text = selected_text
  enable_editing_protected_text = enable_editing_protected_text
```

```
exceptions
  error_dp = 1
  error_db_create = 2.
```

Parameters	Description	Possible values
selected_text	R/3 table with text	
enable_editing_protected_text	Editing for protected sections of text	<b>false</b> Protected sections cannot be edited (default value)  <b>true</b> Protected sections can be edited

## SET\_SELECTION\_INDEXES

## SET\_SELECTION\_INDEXES

### Use

Use this method to set a selection for a text area based on character index. The position is automatically placed within the visible section of the control.

### Features

call method textedit->set\_selection\_indexes

exporting

from\_index = from\_index

to\_index = to\_index

exceptions

error\_cntl\_call\_method = 1.

Parameters	Description
from_index	Character index at which the selection begins. The index is relative to the first character in the control.
to_index	Character index at which the selection ends. The index is relative to the first character in the control.



If the “from” and “to” character positions are the same, no text is selected, but the cursor is placed at the corresponding position.

You can make the selection invisible to the user as follows:

1. Get the current cursor position
2. Switch off the redraw function by calling the method [AUTO\\_REDRAW \[Page 12\]](#) and setting the ENABLE\_REDRAW parameter to false.
3. Set the required selection. Other actions may follow at this point.
4. If required, set the old position.
5. Reenable the automatic redraw by calling the method [AUTO\\_REDRAW \[Page 12\]](#) and setting the ENABLE\_REDRAW parameter to true.

## SET\_SELECTION\_POS

### Use

Use this method to set a selection for a text area. The position is automatically placed within the visible section of the control.

## Features

call method textedit->set\_selection\_pos

```
exporting
  from_line = from_line
  from_pos = from_pos
  to_line = to_line
  to_pos = to_pos
```

```
exceptions
  error_cntl_call_method = 1.
```

Parameters	Description
from_line	Line number at which the selection should begin
from_pos	Position within the line at which the selection should begin
to_line	Line number at which the selection should end
to_pos	Position within the line at which the selection should end



If the “from” and “to” line and position numbers are the same, no text is selected, but the cursor is placed at the corresponding position.

You can make the selection invisible to the user as follows:

6. Get the current cursor position
7. Switch off the redraw function by calling the method [AUTO\\_REDRAW \[Ext.\]](#) and setting the ENABLE\_REDRAW parameter to false.
8. Set the required selection. Other actions may follow at this point.
9. If required, set the old position.
10. Reenable the automatic redraw by calling the method [AUTO\\_REDRAW \[Ext.\]](#) and setting the ENABLE\_REDRAW parameter to true.

## SET\_SELECTION\_POS\_IN\_LINE

### Use

Use this method to position the cursor within a line.

### Features

call method textedit->set\_selection\_pos\_in\_line

```
exporting
  line = line
  pos = pos
```

**SET\_SPACES\_ON\_INDENT**

```
exceptions
  error_cntl_call_method = 1.
```

Parameters	Description
line	Line number at which the selection should begin
pos	Position within the line

**SET\_SPACES\_ON\_INDENT****Use**

Use this method to set the number of spaces by which lines or a selected section are indented. For further information, refer to [INDENT\\_LINES \[Page 28\]](#), [UNINDENT\\_LINES \[Page 55\]](#), [INDENT\\_SELECTION \[Page 29\]](#), and [UNINDENT\\_SELECTION \[Page 56\]](#) . You specify the number of spaces in the class constant M\_SPACES\_ON\_INDENT (for further information, refer to [Class Constants \[Page 61\]](#)).

**Features**

call method textedit->set\_spaces\_on\_indent

```
exporting
  number_of_spaces = number_of_spaces

exceptions
  error_cntl_call_method = 1
  invalid_parameter = 2.
```

Parameters	Description
number_of_spaces	Number of spaces

**SET\_STATUSBAR\_MODE****Use**

Use this method to show or hide the status bar of the control.

**Features**

call method textedit->set\_statusbar\_mode

```
exporting
  statusbar_mode = statusbar_mode
```

**SET\_STATUS\_TEXT**

exceptions  
 error\_cntl\_call\_method = 1  
 invalid\_parameter = 2.

Parameters	Description	Possible values
statusbar_mode	Status bar mode	<b>false</b> Status bar invisible (default value) <b>true</b> Status bar visible

## SET\_STATUS\_TEXT

### Use

Use this method to display a text in the status bar of the control. To show or hide the status bar, use the method [SET\\_STATUSBAR\\_MODE \[Page 49\]](#).

### Features

call method textedit->set\_status\_text

exporting  
 status\_text = status\_text

exceptions  
 error\_cntl\_call\_method = 1.

Parameters	Description
status_text	Text to be displayed in the status bar

## SET\_TEXTMODIFIED\_STATUS

### Use

Use this method to set the changed status of the text. The method [GET\\_TEXTMODIFIED\\_STATUS \[Page 23\]](#) returns the current changed status of a text.

**SET\_TEXT\_AS\_R3TABLE**

If you get text using the methods [GET\\_TEXT\\_AS\\_R3TABLE \[Page 51\]](#) and [GET\\_TEXT\\_AS\\_STREAM \[Page 52\]](#), the changed status is not reset.



This method is useful if you want to ignore changes to a text. For example, if you receive text from the control and want to store it permanently in the database, you can set it to unchanged without having to specify any parameters.

**Features**

call method textedit->set\_textmodified\_status

exporting  
status = status

exceptions  
error\_cntl\_call\_method = 1.

Parameters	Description	Possible values
status	Changed status	<b>false</b> Text not changed (default value)  <b>true</b> Text changed

**SET\_TEXT\_AS\_R3TABLE****Use**

This method allows you to place text from the control into an R/3 table. Existing text is overwritten. The text is passed without information about line breaks.

**Features**

call method textedit->set\_text\_as\_r3table

exporting  
table = table

exceptions  
error\_dp = 1  
error\_dp\_create = 2.

Parameters	Description
table	R/3 table with text



If you use a table with shorter lines than those in the SAP Textedit, the text from the control is only passed up to the length of the table line. The remaining text is lost.

**SET\_TEXT\_AS\_STREAM**

If the lines of the table are longer than the lines of the control, the table lines are filled with trailing spaces.

## SET\_TEXT\_AS\_STREAM

### Use

This method allows you to set text from an R/3 table as a stream. Existing text is overwritten. The text is passed with information about line breaks.

### Features

call method textedit->set\_text\_as\_stream

exporting  
text = text

exceptions  
error\_dp = 1  
error\_dp\_create = 2.

Parameters	Description
text	R/3 table with text

## SET\_TOOLBAR\_MODE

### Use

Use this method to show or hide the toolbar.

### Features

call method textedit->set\_toolbar\_mode

exporting  
toolbar\_mode = toolbar\_mode

exceptions  
error\_cntl\_call\_method = 1  
invalid\_parameter = 2.

Parameters	Description	Possible values
toolbar_mode	Toolbar mode	<b>false</b> Toolbar invisible (default value)  <b>true</b> Toolbar visible

## SET\_WORDBREAK\_PROCEDURE

## SET\_WORDBREAK\_PROCEDURE

### Use

Use this method to control line break behavior for normal text and ABAP coding.

### Features

call method textedit->set\_wordbreak\_procedure

exporting  
text\_type = text\_type

exceptions  
error\_cntl\_call\_method = 1.

Parameters	Description	Possible values
text_type	Line break behavior	text_standard Text  text_abap ABAP coding

## SET\_WORDWRAP\_BEHAVIOR

### Use

Use this method to set the behavior of the line break.

### Features

call method textedit->set\_wordwrap\_behavior

exporting  
wordwrap\_mode = wordwrap\_mode  
wordwrap\_position = wordwrap\_position  
wordwrap\_to\_linebreak\_mode = wordwrap\_to\_linebreak\_mode

exceptions  
error\_cntl\_call\_method = 1.

Parameters	Description	Possible values
------------	-------------	-----------------

**UNCOMMENT\_LINES**

wordwrap_mode	Line break behavior	<b>wordwrap_off</b> No line break  <b>wordwrap_at_windowborder</b> Line break at the edge of the window (default value)  <b>wordwrap_at_fixed_position</b> Line break at a fixed position  For further information, refer to <a href="#">Class Constants [Page 61]</a> and the <a href="#">CONSTRUCTOR [Page 13]</a> method.
wordwrap_position	If <code>wordwrap_mode = textedit-&gt;wordwrap_at_fixed_position</code> : Position for the automatic line break in a line	Default value: -1
wordwrap_to_linebreak_mode	Converts soft line breaks to hard ones when you save in the R/3 System	<b>false</b> Soft line breaks are ignored when you save  <b>true</b> Soft line breaks are converted to hard breaks when you save

## UNCOMMENT\_LINES

### Use

Use this method to convert a set of comment lines back into normal lines.

### Features

call method textedit->uncomment\_lines

```
exporting
  from_line = from_line
  to_line = to_line
  enable_editing_protected_text = enable_editing_protected_text
```

```
exceptions
  error_cntl_call_method = 1.
```

Parameters	Description	Possible values
------------	-------------	-----------------

**UNCOMMENT\_SELECTION**

from_line	First line of block	
to_line	End of block	
enable_editing_protected_text	Editing for protected sections of text	<b>false</b> Protected sections cannot be edited (default value)  <b>true</b> Protected sections can be edited

**UNCOMMENT\_SELECTION****Use**

Use this method to convert the selected comment lines into normal lines.

**Features**

call method textedit->uncomment\_selection

exporting

enable\_editing\_protected\_text = enable\_editing\_protected\_text

exceptions

error\_cntl\_call\_method = 1.

Parameters	Description	Possible values
enable_editing_protected_text	Editing for protected sections of text	<b>false</b> Protected sections cannot be edited (default value)  <b>true</b> Protected sections can be edited

## UNINDENT\_LINES

### Use

Use this method to reduce the indentation of a set of lines by a certain number of spaces. Use the parameter M\_SPACES\_ON\_INDENT of the method [SET SPACES ON INDENT \[Page 49\]](#) to set the number of spaces. The indentation is only reduced if there are spaces at the beginning of the lines in question.

### Features

call method textedit->unindent\_lines

```
exporting
  from_line = from_line
  to_line = to_line
  enable_editing_protected_text = enable_editing_protected_text
```

```
exceptions
  error_cntl_call_method = 1.
```

Parameters	Description	Possible values
from_line	Beginning of the section you want to indent	
to_line	End of the section you want to indent	
enable_editing_protected_text	Editing for protected sections of text	<b>false</b> Protected sections cannot be edited (default value)  <b>true</b> Protected sections can be edited

## UNINDENT\_SELECTION

### Use

Use this method to reduce the indentation of the selection by a certain number of spaces. Use the parameter M\_SPACES\_ON\_INDENT of the method [SET SPACES ON INDENT \[Page 49\]](#) to set the number of spaces. The indentation is only reduced if there are spaces at the beginning of the lines in the selection.

### Features

call method textedit->unindent\_selection

## Functional Listing

```
exporting
  enable_editing_protected_text = enable_editing_protected_text

exceptions
  error_cntl_call_method = 1.
```

Parameters	Description	Possible values
enable_editing_protected_text	Editing for protected sections of text	<b>false</b> Protected sections cannot be edited (default value)  <b>true</b> Protected sections can be edited

## Functional Listing

You requested application help. The following help is available for the current R/3 context:

This functional listing allows you to find the right method of the SAP Textedit Control for the task you are trying to implement. The functions are divided up by themes. Within each thematic group, the relevant methods are arranged alphabetically.

### Areas

[Creating and Destroying a Control \[Ext.\]](#)

[Setting and Getting Text \[Ext.\]](#)

[Setting and Getting Text Positions \[Ext.\]](#)

[Highlighting and Protecting Text \[Ext.\]](#)

[Finding and Replacing Text \[Ext.\]](#)

[Status Bar \[Ext.\]](#)

[Toolbar \[Ext.\]](#)

[Other Functions \[Ext.\]](#)

## Creating a Control

[CONSTRUCTOR \[Page 13\]](#)

## Setting and Getting Text

There are two different ways of exchanging data with tables. One way includes information about line breaks, the other does not. The sets of methods have the suffix AS\_STREAM (containing line break information) or AS\_R3TABLE (without line break information).

Getting Text	Setting Text
<a href="#">GET_LINE_TEXT [Page 19]</a>	
<a href="#">GET_SELECTED_TEXT_AS_R3TABLE [Page 20]</a>	<a href="#">SET_SELECTED_TEXT_AS_R3TABLE [Page 45]</a>
<a href="#">GET_SELECTED_TEXT_AS_STREAM [Page 21]</a>	<a href="#">SET_SELECTED_TEXT_AS_STREAM [Page 46]</a>
<a href="#">GET_TEXT_AS_R3TABLE [Page 23]</a>	<a href="#">SET_TEXT_AS_R3TABLE [Page 51]</a>
<a href="#">GET_TEXT_AS_STREAM [Page 24]</a>	<a href="#">SET_TEXT_AS_STREAM [Page 52]</a>
<a href="#">OPEN_LOCAL_FILE [Page 30]</a>	<a href="#">SAVE_AS_LOCAL_FILE [Page 38]</a>

Deleting Text	
<a href="#">DELETE_TEXT [Page 16]</a>	



If you use GET methods with the type AS\_R3TABLE and use a table with shorter lines than those in the SAP Textedit, the text from the control is only passed up to the length of the table line. The remaining text is lost.

If the lines of the table are longer than the lines of the control, the table lines are filled with trailing spaces.

## Setting and Getting Text Positions

There are two ways of accessing text positions. You can either use a line and position within the line, or the absolute character index. Character index means that all of the characters in the text are sequentially numbered.

The main difference is in the way the data passed to the control (as a table) is visualized. If you are using automatic line breaks in the textedit control, the line numbers in the table may not be the same as the line numbers in the control. The same applies, of course, to positions within a line. You define the behavior of the line break in the constructor (for further information, refer to [CONSTRUCTOR \[Page 13\]](#)).

The difference is particularly apparent when you set the line break to WORDWRAP\_AT\_WINDOWBORDER (automatic line break when the edge of the window is reached). If the user changes the size of the SAPgui and the control container can also be resized, the size of the textedit control will change. If the user makes the SAPgui window smaller, the textedit control will become narrower. Consequently, lines containing words that collide with the window border are automatically split. So even without the user entering any data, the lines and positions within a line are no longer the same as those stored in the internal table at the backend.

## Highlighting and Protecting Text

It is possible to create a similar example for the line break behavior using WORDWRAP\_AT\_FIXED\_POSITION.



If you need a direct correspondence between a position in the internal table and a position in the textedit control, you should, because of the behavior described above, use the character index.

## Access Using Line and Position Numbers

Line and position numbers in a line begin at 1, so the first character in a text is at line 1, position 1.

Lines and position numbers outside the text area have a logical value. If a number is less than 1, it is set to 1. If it is greater than the number of lines or positions, it is set to the number of the greatest existing line or position.

Getting Text Positions	Setting Text Positions
<a href="#">GET_FIRST_VISIBLE_LINE [Page 19]</a>	<a href="#">SET_FIRST_VISIBLE_LINE [Page 42]</a>
<a href="#">GET_SELECTION_POS [Page 22]</a>	<a href="#">SET_SELECTION_POS [Page 47]</a>
<a href="#">GO_TO_LINE [Page 26]</a>	
	<a href="#">SELECT_LINES [Page 39]</a>
	<a href="#">SET_SELECTION_POS_IN_LINE [Page 48]</a>

## Access Using the Character Index

The character index starts at 0. Position zero is the position before the first character. Information about the text position is independent of the line break behavior (see above).

Getting Text Positions	Setting Text Positions
<a href="#">GET_SELECTION_INDEXES [Page 21]</a>	<a href="#">SET_SELECTION_INDEXES [Page 47]</a>

## Highlighting and Protecting Text

### Redrawing the Control

[AUTO\\_REDRAW \[Page 12\]](#)

### Switching Highlighting on and off

[HIGHLIGHT\\_LINES \[Page 27\]](#)

[HIGHLIGHT\\_SELECTION \[Page 27\]](#)

### Switching Input Protection on and off

[PROTECT\\_LINES \[Page 30\]](#)

Finding and Replacing Text

<a href="#">PROTECT_SELECTION [Page 31]</a>
---

<b>Comment Lines</b>
<a href="#">SET_COMMENTS_STRING [Page 40]</a>
<a href="#">SET_HIGHLIGHT_COMMENTS_MODE [Page 43]</a>

<b>Switching Between Display and Change Mode</b>
<a href="#">SET_READONLY_MODE [Page 45]</a>

<b>Moving a Selection to the Visible Part of the Editor Window</b>
<a href="#">MAKE_SELECTION_VISIBLE [Page 30]</a>

## Finding and Replacing Text

<b>Finding and Replacing Text</b>
<a href="#">EMPTY_UNDO_BUFFER [Page 16]</a>
<a href="#">FIND_AND_REPLACE [Page 17]</a>
<a href="#">FIND_AND_SELECT_TEXT [Page 18]</a>
<a href="#">REPLACE_ALL [Page 37]</a>

## Status Bar

<b>Setting the Status Bar</b>	<b>Read-Only Instance Attributes</b>
<a href="#">SET_STATUSBAR_MODE [Page 49]</a>	<a href="#">Instance Attributes [Page 62]</a>
<a href="#">SET_STATUS_TEXT [Page 50]</a>	

## Toolbar

<b>Setting the Toolbar</b>	<b>Read-Only Instance Attributes</b>
<a href="#">SET_TOOLBAR_MODE [Page 52]</a>	<a href="#">Instance Attributes [Page 62]</a>

## Other Functions

## Other Functions

<b>Setting the Line Break</b>	
<a href="#">SET_WORDWRAP_BEHAVIOR [Page 53]</a>	

## Events

To react to an event in your ABAP program, you must have registered it. To register events, use the methods [REGISTER\\_EVENT\\_CONTEXT\\_MENU \[Page 32\]](#), [REGISTER\\_EVENT\\_DBLCLICK \[Page 33\]](#), [REGISTER\\_EVENT\\_F1 \[Page 34\]](#), [REGISTER\\_EVENT\\_F4 \[Page 35\]](#), and [REGISTER\\_EVENT\\_FILEDROP \[Page 36\]](#). Events that are triggered but for which you are not registered are filtered by the presentation server, and not passed to the application server. For further information, refer to [Event Handling \[Ext.\]](#)

Event	Description / Use
CONTEXT_MENU	Calls the context menu using the right-hand mouse button. For further information, refer to <a href="#">Context Menu [Ext.]</a> .
CONTEXT_MENU_SELECTED	Item selection in the context menu. For further information, refer to <a href="#">Context Menu [Ext.]</a> .
DBLCLICK	Double click
F1	Function key F1 pressed
F4	Function key F4 pressed
FILEDROP ON_DRAG ON_DROP ON_DROP_COMPLETE ON_GET_FLAVOR	Drag and drop events. For further information, refer to <a href="#">Drag and Drop [Ext.]</a>




For an example of event handling, refer to the demonstration program SAPTEXTEDIT\_TEST\_EVENTS in development class SAPTEXTEDIT.

## Class Constants

Class Constant	Type / Value	Description / Use
----------------	--------------	-------------------

Instance Attributes

ABAP_COMMENTLINE_IDENTIFIER	TYPE C VALUE *	Used to highlight comment lines when you use the control as a program editor
BOOL_INITIAL FALSE TRUE	TYPE I VALUE -1 TYPE I VALUE 0 TYPE I VALUE 1	Simulation of type BOOL
STRING_LENGTH	TYPE I VALUE 256	Type for strings. When you pass parameters to the control, this definition corresponds to the maximum number of characters that can be transferred between the frontend and application server.  There is no warning if you try to pass a longer string.
WORDWRAP_OFF	TYPE I VALUE 0	No automatic word-wrap in control. You only need this definition when you create an object in the constructor.
WORDWRAP_AT_WINDOWBORDER	TYPE I VALUE 1	Automatic word-wrap at the edge of the control window. You only need this definition when you create an object in the constructor.
WORDWRAP_AT_FIXED_POSITION	TYPE I VALUE 2	Automatic word-wrap at a fixed position in the control. You only need this definition when you create an object in the constructor.



ABAP does not contain any constants with type BOOL. Instead, this type is simulated using the integer constants `false` and `true`. You can address them using `cl_gui_textedit=>>true` and `cl_gui_textedit=>>false`.

## Instance Attributes

The following instance attributes describe the state of the control. They are all flagged as Read Only, so you can use, but not change them.

The frontend attributes are not set from within the control. This means that you can only change them using SET methods from the ABAP proxy. This allows the values of the frontend properties to be reflected in attributes on the ABAP side. To set an attribute, you need to call the corresponding set method, which implicitly updates the corresponding instance attribute. This

**Instance Attributes**

means that you do not need to call the get methods, thus avoiding flushes, which affect performance.


The instance attributes are initialized in the [constructor \[Page 13\]](#), and are updated when you call the corresponding set methods.

Instance Attributes	Type / Value	Description / Use
M_AUTOREDRAW_REFCONTAINER	TYPE I VALUE 0 READ-ONLY	Reference counter: Determines when the control is redrawn
M_AUTO_INDENT	TYPE I READ-ONLY	Value for automatic line indentation
M_COMMENTS_STRING(STRING_LENGTH)	TYPE C READ-ONLY	Character which, when placed at the beginning of a line, signals that the entire line is a comment. To set the value, use the method <a href="#">SET_COMMENTS_STRING [Page 40]</a> . To set the highlighting, use the method <a href="#">SET_HIGHLIGHT_COMMENTS_MODE [Page 43]</a> .
M_FILEDROP_MODE	TYPE I VALUE DROPFILE_EVENT_OFF READ-ONLY	File mode in drag and drop. You set it using the <a href="#">SET_FILEDROP_MODE [Page 41]</a> method.
M_HIGHLIGHT_BREAKPOINTS_MODE	TYPE I READ-ONLY	Breakpoint highlighting on or off (true or false). You set this using the method <a href="#">SET_HIGHLIGHT_BREAKPOINTS_MODE [Page 42]</a> .
M_HIGHLIGHT_COMMENTS_MODE	TYPE I READ-ONLY	Comment line highlighting on or off (true or false). To set the attribute, use the method <a href="#">SET_HIGHLIGHT_COMMENTS_MODE [Page 43]</a> .
M_LOCAL_CONTEXTMENU_MODE	TYPE I READ-ONLY	Visibility of context menu (True: Visible, False: Invisible). You set the attribute using the method <a href="#">SET_LOCAL_CONTEXTMENU_MODE [Page 43]</a> .
M_READONLY_MODE	TYPE I READ-ONLY	Text displayed in protected or input mode (true or false) To set the attribute, use the method <a href="#">SET_READONLY_MODE [Page 45]</a> .

Instance Attributes

M_SPACES_ON_INDENT	TYPE I VALUE 2 READ-ONLY	Number of spaces that you want to insert or delete for indentations. To set the attribute, use the method <a href="#">SET_SPACES_ON_INDENT [Page 49]</a> .
M_STATUS_TEXT(StringLength)	TYPE C READ-ONLY	Text in first element of the status line. To set it, use the method <a href="#">SET_STATUS_TEXT [Page 50]</a> .
M_STATUSBAR_MODE	TYPE I READ-ONLY	Display or suppress the status bar (false or true). To set it, use the method <a href="#">SET_STATUSBAR_MODE [Page 49]</a> .
M_TOOLBAR_MODE	TYPE I READ-ONLY	Display or suppress the toolbar (false or true). To set it, use the method <a href="#">SET_TOOLBAR_MODE [Page 52]</a> .
M_WORDBREAK_PROCEDURE	TYPE I READ-ONLY	Line break behavior for normal text or ABAP code (TEXT_STANDARD or TEXT_ABAP)
M_WORDWRAP_MODE	TYPE I READ-ONLY	<p>Line break behavior:</p> <ul style="list-style-type: none"> <li>• No line break (WORDWRAP_OFF)</li> <li>• Line break at edge of window (WORDWRAP_AT_WINDOW_BORDER)</li> <li>• Line break at a fixed position (WORDWRAP_AT_FIXED_POSITION)</li> </ul> <p>You set these values when you create the instance (<a href="#">CONSTRUCTOR [Page 13]</a> method) or explicitly using the method <a href="#">SET_WORDWRAP_BEHAVIOR [Page 53]</a>.</p>

## Keyboard and Mouse Control in the Editor Window

M_WORDWRAP_POSITION	TYPE I READ-ONLY	<p>Position in a line at which the line break automatically occurs. You set this value when you create the SAP Textedit instance (<a href="#">CONSTRUCTOR [Page 13]</a> method).</p>  <p>This value is only used if you have set M_WORDWRAP_MODE to 2 or WORDWRAP_AT_FIXED_POSITION.</p>
M_WORDWRAP_TO_LINEBREAK_MODE	TYPE I READ-ONLY	<p>Converts soft line breaks to hard ones when you save in the R/3 System:</p> <ul style="list-style-type: none"> <li>• Soft line breaks are ignored when you save. Line ends at the next hard line break (false).</li> <li>• Soft line breaks are converted to hard breaks on saving (true)</li> </ul> <p>You set this value when you create the SAP Textedit instance (<a href="#">CONSTRUCTOR [Page 13]</a> method).</p>
M_NAVIGATE_ON_DOUBLECLICK	TYPE I READ-ONLY	<p>Mode for reacting to a double-click To set the value, use the method <a href="#">SET_NAVIGATE_ON_DOUBLECLICK [Page 44]</a></p>

## Keyboard and Mouse Control in the Editor Window

### Insert and overwrite mode

Keyboard command	Description
Ins	Switches between insert and overwrite mode

Keyboard and Mouse Control in the Editor Window

**Clipboard Control**

Keyboard command	Description
Ctrl + Ins Shift + Ctrl + Ins Ctrl + C Shift + Ctrl + C	Copies the selected text to the clipboard
Ctrl + Del Shift + Ctrl + Del Ctrl + X Shift + Ctrl + X	Cuts the selected text to the clipboard
Ctrl + Ins Shift + Ctrl + Ins Ctrl + V Shift + Ctrl + V	Pastes the contents of the clipboard at the cursor position

**Undo / Redo**

Keyboard command	Description
Ctrl + Z Alt + Backspace	Undo
Ctrl + Y	Redo

**Navigating in the Text**

Keyboard command	Description
-	Cursor moves one line higher
-{}↓	Cursor moves one line lower
-{}←	Cursor moves one character to the left
-{}→	Cursor moves one character to the right
PgUp	Cursor moves up one page
PgDn	Cursor moves down one page
Home	Cursor moves to beginning of line
End	Cursor moves to end of line
Ctrl + ↑	Cursor moves to beginning of the current paragraph

**Keyboard and Mouse Control in the Editor Window**

Ctrl + ↓	Cursor moves to beginning of the next paragraph
Ctrl + ←	Cursor moves to beginning of next word to the left
Ctrl + →	Cursor moves to beginning of next word to the right
Ctrl + PgUp	Cursor moves to the first fully-visible character in the editor window
Ctrl + PgDn	Cursor moves to the last fully-visible character in the editor window
Ctrl + Home	Cursor moves to the first character of the text
Ctrl + End	Cursor moves to the last character of the text

**Selecting Text**

<b>Keyboard command</b>	<b>Description</b>
Ctrl + A Ctrl + Shift + A	Selects entire text
Shift + ↑	Selects the text from the current cursor position one whole line upwards
Shift + ↓	Selects the text from the current cursor position one whole line downwards
Shift + ←	Selects the text from the current cursor position one character to the left
Shift + →	Selects the text from the current cursor position one character to the right
Ctrl + PgUp	Selects one whole editor page upwards from the current cursor position
Ctrl + PgDn	Selects one whole editor page downwards from the current cursor position
Ctrl + Home	Selects the text from the current cursor position to the beginning of the line
Ctrl + End	Selects the text from the current cursor position to the end of the line
Shift + Ctrl + ↑	Selects the text from the current cursor position to the beginning of the current paragraph
Shift + Ctrl + ↓	Selects the text from the current cursor position to the end of the current paragraph
Shift + Ctrl + ←	Selects the text from the current cursor position to the beginning of the next word to the left
Shift + Ctrl + →	Selects the text from the current cursor position to the beginning of the next word to the right
Shift + Ctrl + PgUp	Selects the text from the current cursor position to the first fully visible line in the editor window

**Keyboard and Mouse Control in the Editor Window**

Shift + Ctrl + PgDn	Selects the text from the current cursor position to the last fully visible line in the editor window
Shift + Ctrl + Home	Selects the text from the current cursor position to the beginning of the text
Shift + Ctrl + End	Selects the text from the current cursor position to the end of the text
Press and hold left mouse button and drag mouse	Selects the text block
Single click the left-hand edge of a line with the left mouse button	Selects the entire line
Click the left-hand edge of a line with the left mouse button and drag up or down	Selects a successive set of lines
Ctrl + Single click on the left-hand edge of a line with the left mouse button	Selects entire text

**Find and Replace**

If text is selected, the actions affect the selected area. However, this only applies to actions triggered at the frontend. The selection does not affect find and replace functions started from the ABAP program.

Keyboard command	Description
Ctrl + F Shift + Ctrl + F Alt + Ctrl + F	Displays the find and replace dialog
Ctrl + G Shift + Ctrl + G Alt + Ctrl + G	Finds the next occurrence of the search string

**Drag & Drop Control**

Keyboard command	Description
Left mouse button on selected text	Drag and drop: move
Ctrl + Left mouse button on selected text	Drag and drop: Copy

**Context Menu**

Keyboard command	Description
Ctrl + F10	Context menu
Right mouse button	Context menu

**Indentation**

Keyboard command	Description
Tab	Indents the current line or selected text

## Methods of Class CL\_GUI\_CFW

Shift + Tab	removes the indent for current line or selected text
-------------	--

### Deleting Text

Keyboard command	Description
Backspace	Deletes from right to left: Any selected text is deleted in a single operation
Ctrl + Backspace Shift + Ctrl + Backspace	Deletes a whole word at a time from right to left
Del	Deletes from left to right: Any selected text is deleted in a single operation
Ctrl + Del Shift + Ctrl + Del	Deletes a whole word at a time from left to right

## Methods of Class CL\_GUI\_CFW

The class `CL_GUI_CFW` contains static methods that apply to all instantiated custom controls when you call them.

### dispatch

Use this method to dispatch application events ([see Event Handling \[Ext.\]](#)) to the event handlers registered for the events. If you do not call the method within the PAI event of your application program, it is called automatically by the system after the PAI has been processed. The method returns a return code from which you can tell if the call was successful.

```
CALL METHOD cl_gui_cfw=>dispatch
  IMPORTING return_code = return_code.
```

Parameters	Description
return_code	<p><code>cl_gui_cfw=&gt;rc_found</code>: The event was successfully directed to a handler method.</p> <p><code>cl_gui_cfw=&gt;rc_unknown</code>: The event was not registered in the event list.</p> <p><code>cl_gui_cfw=&gt;rc_noevent</code>: No event was triggered in a control. The function code was therefore a normal one (for example, from a menu entry).</p> <p><code>cl_gui_cfw=&gt;rc_nodispatch</code>: No handler method could be assigned to the event.</p>



An event can only be dispatched once. After that, it is "spent". Consequently, attempting to dispatch the events a second time does not trigger the handler events again.

## flush

Use this method to synchronize the [automation queue \[Ext.\]](#). The buffered operations are sent to the frontend using GUI RFC. At the frontend, the automation queue is processed in the sequence in which you filled it.

If an error occurs, an exception is triggered. You must catch and handle this error. Since it is not possible to identify the cause of the error from the exception itself, there are tools available in the Debugger and the SAPgui to enable you to do so.

**Debugger:** Select the option *Automation Controller: Always process requests synchronously*. The system then automatically calls the method `cl_gui_cfw=>flush` after each method called by the Automation Controller.

**SAPGUI:** In the SAPgui settings, under *Trace*, select *Automation*. The communication between the application server and the Automation Controller is then logged in a trace file that you can analyze at a later date.

```
CALL METHOD cl_gui_cfw=>flush
    EXCEPTIONS CNTL_SYSTEM_ERROR = 1
              CNTL_ERROR = 2.
```



Do not use any more synchronizations in your program than are really necessary. Each synchronization opens a new RFC connection to the SAPgui.

## get\_living\_dynpro\_controls

This method returns a list of reference variables to all active custom controls.

```
CALL METHOD cl_gui_cfw=>get_living_dynpro_controls
    IMPORTING control_list = control_list.
```

Parameters	Description
<code>control_list</code>	List of reference variables of active custom controls. The list has the type <code>CNTO_CONTROL_LIST</code> (defined in class <code>CL_GUI_CFW</code> ).

**set\_new\_ok\_code****set\_new\_ok\_code**

You may only use this method in the handler method of a system event. It sets an **OK\_CODE** that triggers PAI processing. This means that data is transferred from the screen to the program, and you can take control of the program in your PAI modules.

```
CALL METHOD cl_gui_cfw=>set_new_ok_code
  EXPORTING new_code = new_code
  IMPORTING rc = rc.
```

Parameters	Description
new_code	Function code that you want to place in the <b>OK_CODE</b> field ( <b>SY-UCOMM</b> ).
return_code	<p><b>cl_gui_cfw=&gt;rc_posted</b>: The <b>OK_CODE</b> was set successfully and the automatic field checks and PAI will be triggered after the event handler method has finished.</p> <p><b>cl_gui_cfw=&gt;rc_wrong_state</b>: The method was not called from the handler method of a system event.</p> <p><b>cl_gui_cfw=&gt;rc_invalid</b>: The <b>OK_CODE</b> that you set is invalid.</p>

**update\_view**

Calling the [flush \[Page 70\]](#) method only updates the automation queue if the queue contains return values.

If you have a queue with no return values, and want to ensure that it is synchronized, you can use the Control Framework method **CL\_GUI\_CFW=>UPDATE\_VIEW**. You should only use this method if you absolutely need to update the GUI. For example, you might have a long-running application in which you want to provide the user with regular updates on the status of an action.

```
CALL METHOD cl_gui_cfw=>update_view
  EXCEPTIONS CNTL_SYSTEM_ERROR = 1
             CNTL_ERROR       = 2.
```

**Methods of Class CL\_GUI\_OBJECT**

The class **CL\_GUI\_OBJECT** contains important methods for custom control wrappers. The only one relevant for application programs is the [is\\_valid \[Page 71\]](#) method.

**is\_valid**

This method informs you whether a custom control for an object reference still exists at the frontend.

```
CALL METHOD my_control->is_valid
IMPORTING result = result.
```

Parameters	Description
result	0: Custom control is no longer active at the frontend 1: Custom control is still active

## free

Use this method to destroy a custom control at the frontend. Once you have called this method, you should also initialize the object reference (**FREE my\_control**).

```
CALL METHOD my_control->free
EXCEPTIONS cntl_error = 1
cntl_system_error = 2.
```

## Methods of Class CL\_GUI\_CONTROL

The class **CL\_GUI\_CONTROL** contains methods that you need to set control attributes (for example, displaying the control), register events, and destroy controls.

## constructor

This method is called by the control wrapper when you instantiate a control.



To instantiate a SAP control, always call the constructor of its class.

```
CREATE OBJECT my_control
EXPORTING clsid = clsid
lifetime = lifetime
shellstyle = shellstyle
parent = parent
autoalign = autoalign
EXCEPTIONS cntl_error = 1
cntl_system_error = 2
create_error = 3
lifetime_error = 4.
```

Parameters	Description
clsid	ID of the class

**finalize**

lifetime	<p>Lifetime management parameter. The following values are permitted:</p> <p><b>my_control-&gt;lifetime_imate</b>: The control remains alive for the duration of the internal session (that is, until the session is ended by one of the following statements: <code>leave program. leave to transaction. set screen 0, leave screen.</code>). After this, the <a href="#">finalize [Page 73]</a> method is called.</p> <p><b>my_control-&gt;lifetime_dynpro</b>: The control remains alive for the lifetime of the screen instance, that is, for as long as the screen remains in the stack. After this, the <a href="#">free [Page 72]</a> method is called.</p> <p>Using this mode automatically regulates the visibility of the control. Controls are only displayed when the screen on which they were created is active. When other screens are active, the controls are hidden.</p> <p><b>my_control-&gt;lifetime_default</b>: If you create the control in a container, it inherits the lifetime of the container. If you do not create the control in a container (for example, because it is a container itself), the lifetime is set to <b>my_control-&gt;lifetime_imate</b>.</p>
Shellstyle	<p>Controls the appearance and behavior of the control</p> <p>You can pass any constants from the ABAP include <code>&lt;CTLDEF&gt;</code> that begin with <code>WS</code>. You can combine styles by adding the constants together. The default value sets a suitable combination of style constants internally.</p>
parent	<p>Container in which the SAP Picture Control can be displayed (<a href="#">see also SAP Container [Ext.]</a>).</p>
autoalign	<p>' ': Control is not automatically aligned</p> <p>'X': Control is automatically aligned. This uses the maximum available space within a container.</p>

## finalize

This method is redefined by the relevant control wrapper. It contains specific functions for destroying the corresponding control. This method is called automatically by the [free \[Page 72\]](#) method, before the control is destroyed at the frontend.

```
CALL METHOD my_control->finalize.
```

## set\_registered\_events

Use this method to register the events of the control. [See also: Event Handling \[Ext.\]](#)

```
CALL METHOD my_control->set_registered_events
  EXPORTING events      = events
  EXCEPTIONS cntl_error   = 1
             cntl_system_error = 2
             illegal_event_combination = 3.
```

**get\_registered\_events**

Parameters	Description
events	Table of events that you want to register for the custom control <code>my_control</code> .

The table `events` is a list of the events that you want to register. It is defined with reference to table type `CNTL_SIMPLE_EVENTS`. The table type is based on the structure `CNTL_SIMPLE_EVENT`, which consists of the following fields:

Field	Description
EVENTID	Event name
APPL_EVENT	Indicates whether the event is a system event (initial) or an application event (X).

The values that you assign to the field `EVENTID` are control-specific and therefore described in the documentation of the individual controls.

## get\_registered\_events

This method returns a list of all events registered for custom control `my_control`.

```
CALL METHOD my_control->get_registered_events
    IMPORTING events = events
    EXCEPTIONS cntl_error = 1.
```

Parameters	Description
events	Table of events that you want to register for the custom control <code>my_control</code> .

The table `events` is a list of the events that you want to register. It is defined with reference to table type `CNTL_SIMPLE_EVENTS`. The table type is based on the structure `CNTL_SIMPLE_EVENT`, which consists of the following fields:

Field	Description
EVENTID	Event name
APPL_EVENT	Indicates whether the event is a system event (initial) or an application event (X).

The values that you assign to the field `EVENTID` are control-specific and therefore described in the documentation of the individual controls.



For general information about event handling, refer to the [Event Handling \[Ext.\]](#) section of the SAP Control Framework documentation.

## is\_alive

This method informs you whether a custom control for an object reference still exists at the frontend.

**set\_alignment**

CALL METHOD my\_control->is\_alive  
RETURNING state = state.

Parameters	Description
state	<b>my_control-&gt;state_dead</b> : Custom control is no longer active at the frontend <b>my_control-&gt;state_alive</b> : Custom control is active on the current screen. <b>my_control-&gt;state_alive_on_other_dynpro</b> : Custom control is not active on the current screen, but is still active (but invisible) at the frontend.

**set\_alignment**

Use this method to align the custom control within its container:

CALL METHOD my\_control->set\_alignment  
EXPORTING alignment = alignment  
EXCEPTIONS cntl\_error = 1  
cntl\_system\_error = 2.

Parameters	Description
alignment	Control alignment

The `alignment` parameter may consist of combinations of the following alignments:

Name	Description
my_control->align_at_left	Alignment with left-hand edge
my_control->align_at_right	Alignment with right-hand edge
my_control->align_at_top	Alignment with top edge
my_control->align_at_bottom	Alignment with bottom edge

You can combine these parameters by adding the components:

alignment = my\_control->align\_at\_left + my\_control->align\_at\_top.

**set\_position**

Use this method to place the control at a particular position on the screen.



The position of the control is usually determined by its container.

CALL METHOD my\_control->set\_position  
EXPORTING height = height  
left = left  
top = top  
width = width

```
EXCEPTIONS cntl_error      = 1
           cntl_system_error = 2.
```

Parameters	Description
height	Height of the control
left	Left-hand edge of the control
top	Top edge of the control
width	Width of the control

## set\_visible

Use this method to change the visibility of a custom control.

```
CALL METHOD my_control->set_visible
  EXPORTING visible      = visible
  EXCEPTIONS cntl_error  = 1
           cntl_system_error = 2.
```

Parameters	Description
visible	<b>x</b> : Custom control is visible <b>' '</b> : Custom control is not visible

## get\_focus

This static method returns the object reference of the control that has the focus.

```
CALL METHOD cl_gui_control=>get_focus
  IMPORTING control      = control
  EXCEPTIONS cntl_error  = 1
           cntl_system_error = 2.
```

Parameters	Description
control	Object reference ( <b>TYPE REF TO cl_gui_control</b> ) to the control that has the focus.

## set\_focus

Use this static method to set the focus to a custom control.

**get\_height**

```
CALL METHOD cl_gui_control=>set_focus
  EXPORTING control      = control
  EXCEPTIONS cntl_error  = 1
             cntl_system_error = 2.
```

Parameters	Description
control	Object reference ( <b>TYPE REF TO cl_gui_control</b> ) to the control on which you want to set the focus.

**get\_height**

This method returns the height of the control.

```
CALL METHOD control->get_height
  IMPORTING height      = height
  EXCEPTIONS cntl_error  = 1.
```

Parameters	Description
height	Current height of the control

**get\_width**

This method returns the width of the control.

```
CALL METHOD control->get_width
  IMPORTING width      = width
  EXCEPTIONS cntl_error  = 1.
```

Parameters	Description
width	Current width of the control

**Methods of the Class CL\_DRAGDROP**

The class **CL\_DRAGDROP** contains methods that describe the [drag and drop \[Ext.\]](#) behavior of a custom control.

**constructor**

The constructor creates an instance for the description of the drag and drop behavior of a control.

CREATE OBJECT dragdrop.

## add

This method adds a new description to the drag and drop behavior. You can store any number of descriptions, but you may not add the same description more than once.

```
CALL METHOD dragdrop->add
  EXPORTING flavor      = flavor
           dragsrc     = dragsrc
           droptarget  = droptarget
           effect      = effect
           effect_in_ctrl = effect_in_ctrl
  EXCEPTIONS already_defined = 1
             obj_invalid    = 2.
```

Parameters	Description
flavor	Description of the new flavor
dragsrc	'x': The description is a drag source
droptarget	'x': The description is a drop target
effect	Drop effect of the description between different custom controls. The following effects are supported:  <b>dragdrop-&gt;copy</b> : Appearance of the mouse when using drag and drop to copy.  <b>dragdrop-&gt;move</b> : Appearance of the mouse when using drag and drop to move.  <b>dragdrop-&gt;none</b> : Drag and drop is not possible.
effect_in_ctrl	Drop effect of the description in the same custom control. The following effects are supported:  <b>dragdrop-&gt;copy</b> : Appearance of the mouse when using drag and drop to copy.  <b>dragdrop-&gt;move</b> : Appearance of the mouse when using drag and drop to move.  <b>dragdrop-&gt;none</b> : Drag and drop is not possible.  <b>dragdrop-&gt;use_default_effect</b> : Uses the same effect specified in the <b>effect</b> parameter.

Exceptions	Description
already_defined	The specified flavor has already been defined.
obj_invalid	The object has already been destroyed using the method <a href="#">destroy [Page 79]</a> .

## clear



If you use the `copy` and `move` effects when you define the flavor, the system uses the `move` effect when the user drags an object normally, and the `copy` effect when the user presses and holds the CTRL key while dragging.

## clear

Deletes the contents of the instance. Once you have called this method, you cannot perform any more drag and drop operations on the corresponding custom control.

CALL METHOD dragdrop->clear  
EXCEPTIONS obj\_invalid = 1.

Exceptions	Description
obj_invalid	The object has already been destroyed using the method <a href="#">destroy [Page 79]</a> .

## destroy

Deletes the contents of the instance. The instance itself is also destroyed. Once you have called this method, you cannot perform any more drag and drop operations on the corresponding custom control.

CALL METHOD dragdrop->destroy.

## get

Returns the complete description of a flavor.

CALL METHOD dragdrop->get  
EXPORTING flavor = flavor  
IMPORTING isdragsrc = isdragsrc  
isdroptarget = isdroptarget  
effect = effect  
effect\_in\_ctrl = effect\_in\_ctrl  
EXCEPTIONS not\_found = 1  
obj\_invalid = 2.

Parameters	Description
flavor	Name of the flavor
dragsrc	'x': The description is a drag source
droptarget	'x': The description is a drop target

**get\_handle**

effect	Drop effect of the description between different custom controls. The following effects are supported:  <b>dragdrop-&gt;copy</b> : Appearance of the mouse when using drag and drop to copy.  <b>dragdrop-&gt;move</b> : Appearance of the mouse when using drag and drop to move.  <b>dragdrop-&gt;none</b> : Drag and drop is not possible.
effect_in_ctrl	Drop effect of the description in the same custom control. The following effects are supported:  <b>dragdrop-&gt;copy</b> : Appearance of the mouse when using drag and drop to copy.  <b>dragdrop-&gt;move</b> : Appearance of the mouse when using drag and drop to move.  <b>dragdrop-&gt;none</b> : Drag and drop is not possible.  <b>dragdrop-&gt;use_default_effect</b> : Uses the same effect specified in the <b>effect</b> parameter.

Exceptions	Description
already_defined	The specified flavor has already been defined.



If you use the **copy** and **move** effects when you define the flavor, the system uses the **move** effect when the user drags an object normally, and the **copy** effect when the user presses and holds the CTRL key while dragging.

## get\_handle

This method returns the handle of the drag and drop position. In most cases, you will not need to use this method. However, for tabular mass data interfaces (such as the SAP Tree), you must copy this handle into the interface table.

```
CALL METHOD dragdrop->get_handle
IMPORTING handle = handle
EXCEPTIONS obj_invalid = 1.
```

Parameters	Description
handle	Handle of the drag and drop description

Exceptions	Description
obj_invalid	The object has already been destroyed using the method <a href="#">destroy [Page 79]</a> .

## modify

## modify

Use this method to change an existing flavor.

```
CALL METHOD dragdrop->modify
  EXPORTING flavor      = flavor
           dragsrc     = dragsrc
           droptarget   = droptarget
           effect       = effect
           effect_in_ctrl = effect_in_ctrl
  EXCEPTIONS not_found = 1
            obj_invalid = 2.
```

Parameters	Description
flavor	Name of the flavor
dragsrc	'x': The description is a drag source
droptarget	'x': The description is a drop target
effect	Drop effect of the description between different custom controls. The following effects are supported:  <b>dragdrop-&gt;copy</b> : Appearance of the mouse when using drag and drop to copy.  <b>dragdrop-&gt;move</b> : Appearance of the mouse when using drag and drop to move.  <b>dragdrop-&gt;none</b> : Drag and drop is not possible.
effect_in_ctrl	Drop effect of the description in the same custom control. The following effects are supported:  <b>dragdrop-&gt;copy</b> : Appearance of the mouse when using drag and drop to copy.  <b>dragdrop-&gt;move</b> : Appearance of the mouse when using drag and drop to move.  <b>dragdrop-&gt;none</b> : Drag and drop is not possible.  <b>dragdrop-&gt;use_default_effect</b> : Uses the same effect specified in the <b>effect</b> parameter.

Exceptions	Description
not_found	The specified flavor does not exist
obj_invalid	The object has already been destroyed using the method <a href="#">destroy [Page 79]</a> .



If you use the `copy` and `move` effects when you define the flavor, the system uses the `move` effect when the user drags an object normally, and the `copy` effect when the user presses and holds the CTRL key while dragging.

## remove

Use this method to delete a flavor.

```
CALL METHOD dragdrop->remove
  EXPORTING flavor = flavor
  EXCEPTIONS not_found = 1
             obj_invalid = 2.
```

Parameters	Description
flavor	Name of the flavor

Exceptions	Description
not_found	The specified flavor does not exist
obj_invalid	The object has already been destroyed using the method <a href="#">destroy [Page 79]</a> .

## Methods of the Class CL\_DRAGDROPOBJECT

The class CL\_DRAGDROPOBJECT describes the context of a [drag and drop operation \[Ext.\]](#). It contains information about the source object, the flavor of the drag and drop operation, and information about the source and target.

### set\_flavor

You can only use this method within event handling for the ONGETFLAVOR event. Use the `newflavor` parameter to determine the flavor that you want to use in the drag and drop operation. You receive a list of available flavors as an event parameter.

```
CALL METHOD dragdropobject->set_flavor
  EXPORTING newflavor = newflavor
  EXCEPTIONS illegal_state = 1
             illegal_flavor = 2.
```

Parameters	Description
newflavor	Name of the flavor

**abort**

Exceptions	Description
invalid_state	You did not call the method from within event handling for <b>ONGETFLAVOR</b> .
obj_invalid	You used a flavor that is not supported by the current drag and drop situation.

**abort**

Terminates the drag and drop operation immediately. No further events are triggered.

CALL METHOD dragdropobject->abort.