4D Write®

Language Reference
Windows and Mac OS Versions

4D Write Language Reference
Version 2003 for Windows® and Mac™ OS

Contents

1

4D Write, Introduction to the language

4D Write, Introduction to the language

version 6.0

4D Write is a plug-in that adds word processing commands and capabilities to 4th Dimension. With these commands, you can automate tasks typically done manually on a document, such as:

- Execute menu commands
- Open and save documents
- Set the margins of a document
- Set display attributes.

All 4D Write commands added to 4th Dimension are preceded by the letters WR. This distinguishes these commands from those of 4th Dimension or any other plug-ins.

4D Write documentation
The documentation available for 4D Write consists of two manuals: 4D Write User reference and 4D Write Language Reference. The purpose of this manual (4D Write Language Reference) is to describe the use of the programming language of 4D Write. For more information about how to use 4D Write, please refer to the 4D Write User Reference manual.

See also

Commands in the Method Editor, Documents in 4D Write Areas, Multi-platform Document Management, Referring to Characters.

Multi-platform Document Management 4D Write, Introduction to the language

version 6.7 (Modified)

4D Write, like 4th Dimension and 4D Server, is a multi-platform program. So, a database created under Mac OS, and that uses 4D Write, can be run under Windows with no modifications, and vice versa. This is possible only if you use the correct versions of the software.

File Equivalents on Mac OS and Windows

The following table indicates the file equivalents of 4D Write documents on Mac OS and Windows.

	MacOS			**Windows**	**Virtual Types (*)**
Document	**Type**	**Creator**	**Extension**		
4D Write document	4WR7	4DW7	4W7	4WR7	
RTF	TEXT	4DW7	RTF	RTF	
Windows Text only	TEXT	4DW7	TXT	ASCW	
MacOS Text only	TEXT	4DW7	TXT	ASCM	
Unicode Text document	TEXT	4DW7	TXT	ASCU	
HTML document	TEXT	MOSS	HTML	HTML	
Word 6/95 document	W6BN	MSWD	DOC	DOC6	
Word 97 PC/98 Mac	W8BN	MSWD	DOC	DOC8	

(*) These types are used by the WR OPEN DOCUMENT and WR SAVE DOCUMENT commands only.

Documents

The following rules must be acknowledged:

• Under Mac OS, 4D Write uses the type and creator to recognize documents. For example, type 4WR7, creator 4DW7 = 4D Write document.
The complete access path includes the disk name, folder names, and document name, each separated by a colon (:). For example, MyDisk:Folder1:Folder2:Mydatabase.

• Under Windows, 4D Write uses the file name extension to recognize documents. For example, .4W7 = 4D Write document. The complete access path includes the disk letter, directory names, and document name, each separated by a backslash (\). For example, D:\Directory1\Directory2\Mydatabase.

• A 4D Write document created under Mac OS and copied onto Windows can be opened directly, provided that it has been saved with its file name extension. For example, the MyDoc document saved as MyDoc.4W7, copied onto a PC volume, can be opened with no further handling.

• A 4D Write document created under Windows and copied onto Mac OS or Power Macintosh can be opened with no further handling.

Templates

To share templates between Mac OS and Windows clients, regardless of the server platform, the procedure is transparent for users, as described below.

If the server is a Windows machine, the name of the template file will be AreaName_.4WT and if the server is a Macintosh machine the name of the template file will be AreaName_.

Templates are saved in the database folder with 4D single-user and 4D Server (if templates are saved on the server, which is the default option).
If, with 4D Server, you decided to store templates locally (on client machines), they are saved:
• On MacOS, in the folder System:Preferences:4D:4D Write Templates:DatabaseName
• On Windows, in the folder Windows\4D\4D Write Templates\DatabaseName

See also

Documents in 4D Write Areas.

Language Conventions in this Manual

4D Write, Introduction to the language

version 6.0

Description

In this manual, 4D Write commands are printed in all uppercase letters using a special font, for example: WR ON COMMAND. 4D Write functions are shown with an initial capital letter, for example: WR Get styled text.

When 4D Write commands or functions appear in methods or object methods, they are displayed in a bold italic typeface to differentiate them from built-in 4th Dimension commands and functions. Non-italic bold text indicates 4th Dimension language terms.

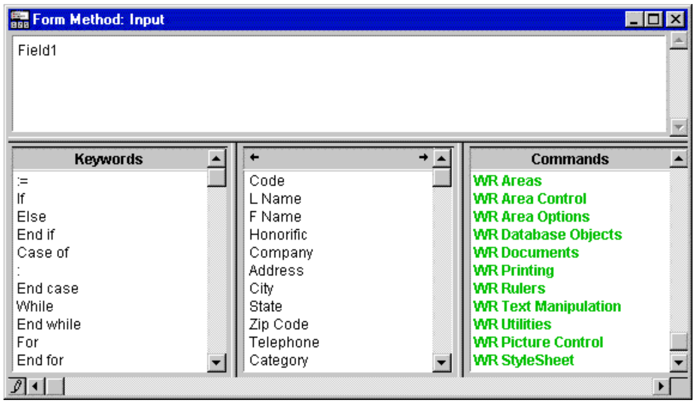
```
QUERY([Templates];[Templates]ID=vNumber) ` 4th Dimension command
If (Records in selection ([Templates])=1)
   WR PICTURE TO AREA (Area;[Templates]Doc) ` 4D Write command
End if
```

In some examples in this manual, a line of code may be continued on a second or third line due to space limitations. However, when you type these examples, keep those lines of code on a single line—do not press the Return key and cause a break in flow.

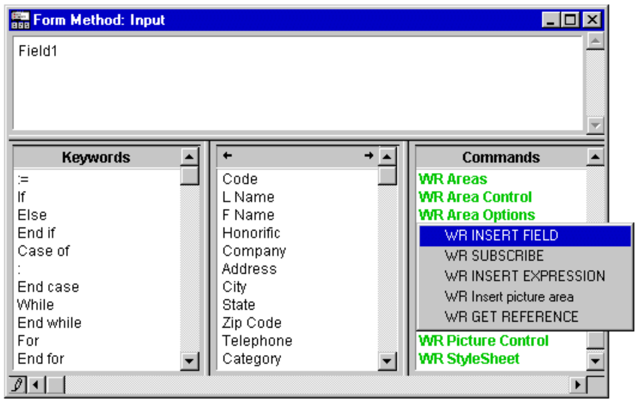
See also

Commands in the Method Editor.

The 4D Write commands are grouped into “themes” in the Method editor. The themes are located at the end of the Routines list.



If you have more than one plug-in installed, the command themes appear in the order in which the plug-ins were installed.
You can place a 4D Write command in a method just as you do any 4th Dimension command. You can either type it directly into the method or choose the command from the pop-up menu in the Routines list.



You can use a 4D Write command in any type of method—project, trigger, form, object or database. The commands are especially useful in object methods activated by objects on the same form as the document area.

See also

Language Conventions in this Manual.

Documents in 4D Write Areas

4D Write, Introduction to the language

version 6.0

There are three types of areas available to you in 4th Dimension:

• External areas in forms

• External windows

• Offscreen areas.

To use a 4D Write document, you either create an external area on a form or open an external window. You create an external area by drawing the area on a form in the Design environment. You open an external window either by choosing 4D Write from the Windows menu in the User environment or by executing the External window function.

In addition to creating visible areas, you can create invisible offscreen areas. For more information, refer to the paragraph “4D Write Offscreen Areas”, later in this section.

4D Write Area ID Number and Variable

4D Write uses variables to store the location of 4D Write areas, external windows, and offscreen areas. You reference the area on which you want to perform an operation by passing the variable containing the area’s ID number as a parameter to the command or function.
In the command descriptions that follow this introduction, the Area parameter refers to the variable identifying the document area.

There are two types of Area variables:

• External object names
When you create and name a 4D Write area, 4th Dimension automatically recognizes the name of the 4D Write area as a variable referring to the area. For example, you would refer to the Letter area by specifying “Letter” as for the Area parameter.

• Variables you create for an external window or offscreen area
When you create an external window or offscreen area using the Open external window or WR New offscreen area functions, you can store the area ID number returned by the function in a variable. You can then use the variable to refer to the external window or offscreen area in other commands and functions. To store the value in a variable, you place the variable name and the assignment operator (:=) to the left of the function in the line of code.

Most 4D Write commands require you to specify an area before they can be executed.

4D Write Plug-in Areas

When you want a 4D Write document to appear in a 4th Dimension form, you must create a plug-in area on the form and assign it a unique name, specifying the plug-in type as 4D Write.

4th Dimension allows you to save this document with the record.

You will probably most often use the plug-in area to store a document or to use it instead of a text field if formatting is important.

4D Write External Window Areas

4th Dimension allows you to create a 4D Write document in an independent area called an external window. External windows are useful when you want the user to have access to a word processor at any time to write letters, memos or other documents.

Issuing the 4th Dimension function, Open external window, from a method opens a specified window and returns an area ID in a long integer variable. You can reference this variable whenever you want to issue a 4D Write command to affect the external window.

For example:

```
    vWrite:=Open external window (50; 50; 350; 450; 8; "Merge Letter"; "_4D Write" )
```

For more information about the Open external window command, please refer to its definition in the 4D Language Reference manual.

4D Write Offscreen Areas

An offscreen area is stored in memory and is not visible to the programmer or user. You can use an offscreen area to modify a document before a user views it or to save the document so a user can revert to the original, if necessary.

WR New offscreen area and WR PICTURE TO AREA are the two commands used to create an offscreen area. Remember to delete the offscreen area after you are done with it to free the memory it uses.

When placed in a global method, the following code creates an offscreen area for saving the document.

```
QUERY([Employee];[Employee]ID=vID)
   If (Records in selection([Employee]=1)
      Area:=WR New offscreen area
      WR PICTURE TO AREA(Area;[Employee]Review_)
         `Store the review in the offscreen area
      MODIFY RECORD([Employee])
         `Modify the employees record
      WR DELETE OFFSCREEN AREA(Area)
         `Free the memory used by the offscreen area
   End if
```

Using a button on a form, you can allow a user to revert to the original saved document.

You can create a button on the input form and assign it the following code:

```
Review:=WR Area to picture(Area)
   `Places the offscreen area that contains the original document into the external
   `area contained in the Review form.
```

See also

Multi-platform Document Management, Referring to Characters.

4D Write Menu Items

You can procedurally gain access to a 4D Write menu and select a menu item. In a method, you can determine the status of a menu or menu item. Each menu item is referenced by a unique integer. See Appendix B: Menu Item Numbers for a listing of menu item integers.

The menu item integers are generally based on the location of the menu and menu item. The menus are numbered from left to right in ascending order. For example, File = 100 and Edit = 200. Likewise, menu items are numbered in ascending order from top to bottom.

The numbers for these menu items always remain the same, even in future versions of 4D Write which may have new menu items. Any new menu items will use different numbers, even if placed between current menu items. This placement will invalidate the general rule of numbering menu items, but the menu references you use in methods will remain accurate, so you will not need to update them.

See also

Appendix B: Menu Item Numbers, Commands in the Method Editor.

A character in a document is referred to by its sequential number. Commands that refer to characters enable you to specify either a single character or a range of characters. For example, you can specify a word, a sentence, or whole blocks of text to be selected.

You use the WR GET SELECTION command to determine the positions of selected characters in a 4D Write area. The command uses the $First and $Last parameters to refer to the range of selected characters. The $First parameter is always one less than the first character selected. The $Last parameter is equal to the last character selected.

Example

For example, the following expression returns the positions of the selected text in Area into the $First and $Last variables:

```
WR GET SELECTION(Area;$First;$Last)
```

To select text in a 4D Write area, you need to reference characters. In most cases, you must first select text before using a command to manipulate it.

See also

Documents in 4D Write Areas.

2

WR Area Control

Area Control, Introduction

WR Area Control

version 6.7.2 (Modified)

The commands and functions of the theme "WR Area Control" allow you to control the display and the operation of your 4D Write areas.

You can control the screen updates by using the WR SCROLL TO SELECTION, WR UPDATE MODE and WR REDRAW commands.

The WR ON COMMAND and WR Get on command method commands allow you to control the behavior of the menu items of your areas.

You can retrieve menu status info (WR GET COMMAND INFO), as well as activate or lock menu items (WR EXECUTE COMMAND, WR LOCK COMMAND).

Also, the WR SET DOC PROPERTY and WR Get doc property commands provide you with information and control options on interface objects in your 4D Write areas.

WR SCROLL TO SELECTION

version 6.0

WR SCROLL TO SELECTION (area)

Parameter	Type		Description
area	Longint	→	4D Write area

Description

The WR SCROLL TO SELECTION command scrolls area until the selected text is visible. This command is useful when modifications are made through 4D Write commands and the user needs to view the resulting changes.

Note : The WR SCROLL TO SELECTION command has no effect if the screen updates have been frozen beforehand using the WR UPDATE MODE command.

Examples

See the examples for the WR Get font and WR SET CURSOR POSITION commands.

WR EXECUTE COMMAND

WR EXECUTE COMMAND (area; cmdNumber)

Parameter	Type		Description
area	Longint	→	4D Write area
cmdNumber	Longint	→	Number of the command to execute

Description

The WR EXECUTE COMMAND command causes the action associated with a 4D Write menu command or toolbar button to be executed. The most common use for this command is to execute a command after the user has chosen that command and your code has intercepted the user's choice through the WR ON COMMAND command.

Note: The list of commands and their references are available in Appendix B. You can either pass a constant or a value.

See also

Appendix B: Menu Item Numbers, WR GET COMMAND INFO, WR ON COMMAND.

version 6.5

WR GET COMMAND INFO

WR GET COMMAND INFO (area; commandNumber; applied; stringValue; name; status)

Parameter	Type		Description
area	Longint	→	4D Write area
commandNumber	Longint	→	Number of the command to process
applied	Longint	←	0=not applied, 1=applied, 2=partially applied
stringValue	String	←	Selected text value
name	String	←	Command name or text of the Tip
status	Integer	←	0=disabled 1=enabled

Description

The WR GET COMMAND INFO command allows you to get the status of the menu or toolbar command whose number is passed in commandNumber.

Note: The list of commands and their references is available in Appendix D: 4D Write Constants (theme "WR Commands"). You can either pass a value or a constant.

applied returns a value indicating whether the command is applied, not applied, or partially applied, to the current selection of text. applied will equal 0 if the command is not applied, 1 if it is applied, or 2 is it is partially applied. For example, consider the **Bold** menu command (Constant: wr cmd bold , Value: 502). When the following statement is executed:

```
⇒    WR GET COMMAND INFO(area;wr cmd bold;applied;stringValue;name;status)
```

applied=1 if the currently selected text is in bold
applied=0 if the currently selected text is not in bold
applied=2 if only part of the currently selected text is in bold

stringValue contains a text that varies and is specific to each command. For example, consider the **Font** drop-down list (Constant: wr cmd font dropdown, Value: 1002). When the following statement is executed:

```
⇒    WR GET COMMAND INFO(area;wr cmd font dropdown;applied;stringValue;name;
                                                                   status)
```

stringValue="Arial" if this is the currently selected font name.

name contains the name of the command. This is either the text of the menu command or the text of the tip displayed for that command.

status returns the status of the command. status will equal 0 if the command is disabled, and 1 if it is enabled.

Example

A form contains a button switching between hiding or showing invisible characters. The title of the button depends on the current screen settings:

```
⇒      WR GET COMMAND INFO(area;wr cmd view invisibles;vApplied;vStringValue;vName;
                                                                          vStatus)
       Case of
          : (vApplied=1)
             BUTTON TEXT(bStatus;"Hide Invisible Characters")
          : (vApplied=0)
             BUTTON TEXT(bStatus;"Show Invisible Characters")
       End case
```

See also

Appendix B: Menu Item Numbers, WR EXECUTE COMMAND.

WR Get doc property

WR Get doc property (area; property) → Real

Parameter	Type		Description
area	Longint	→	4D Write area
property	Integer	→	Number of the property to read
Function result	Real	←	Value for the property tested

Description

The WR Get doc property command allows you to get the properties of the document currently opened in the 4D Write area referenced by area.

For some properties, WR Get doc property returns 1 (True) or 0 (False). An example is property 2 (wr view ruler).

For other properties, WR Get doc property returns a number expressed in the current default unit. An example is property 37 (wr paper width).

Note : property can be set using constants.

The list of document properties is available in Appendix D: 4D Write Constants. You can either pass the constant or the value.

Examples

See the examples for the WR SET DOC PROPERTY, WR INSERT PAGE NUMBER, WR GET CURSOR POSITION and WR SET PICTURE IN PAGE INFO commands.

See also

WR SET DOC PROPERTY.

WR LOCK COMMAND

version 6.5

WR LOCK COMMAND (area; cmdNumber; locked)

Parameter	Type		Description
area	Longint	→	4D Write area
cmdNumber	Longint	→	Number of the command to process
locked	Integer	→	0=enables the execution 1=does not enable the execution

Description

The WR LOCK COMMAND command allows you to prevent the user from being able to execute the command whose number is passed in cmdNumber. This can concern either a menu command or a palette command. This command affects the user's access to the indicated command only in the 4D Write area referenced by area. Access to the command is unaffected in other 4D Write areas.

• If locked equals 1, the command will not execute when it is called and will be disabled (grayed out) in the menus and palettes where it appears.
• If locked equals 0, the command will be executed when it is called.

Notes:
• Even if a command is locked, your code can still execute it using the WR EXECUTE COMMAND command.
• WR ON COMMAND will not be called if the user tries to select a command that is disabled.
• When a menu or submenu is passed in cmdNumber, the menu and all its commands will be disabled (grayed out).

Although the commands of a disabled menu cannot be selected, keyboard equivalents or toolbar buttons can still be used. If you want to completely lock these commands, you must call WR LOCK COMMAND specifically for each menu item.

Note: The list of constants is available in Appendix D: 4D Write Constants.

Examples

(1) You want the designer to be the only user that can access the Design environment:

```
    If(Current user="Designer")
⇒       WR LOCK COMMAND(Area;wr cmd insert 4D expression;0)
    Else
⇒       WR LOCK COMMAND(Area;wr cmd insert 4D expression;1)
    End if
```

(2) If the user name is not "Guru", the user will not be allowed to create new documents:

```
        If(Form event=On load)
           If (Current user#"Guru")
⇒               WR LOCK COMMAND(Area;wr cmd new;1)
           End if
        End if
```

See also

Appendix B: Menu Item Numbers, Appendix D: 4D Write Constants, WR ON COMMAND.

WR ON COMMAND

WR ON COMMAND (area; 4DRepMethod)

Parameter	Type		Description
area	Longint	→	4D Write area
4DRepMethod	String	→	Replacement method

Description

The WR ON COMMAND command executes the method passed as 4DRepMethod when a 4D Write command is invoked by the user,either by the selection of a menu command or by a click on a button. If area equals zero, 4DRepMethod will apply to each 4D Write area until the database is closed or until the following call to WR ON COMMAND is made: WR ON COMMAND(0;"").

4DRepMethod receives two parameters:
• $1 is a Longint that represents area.
• $2 is a Longint that designates the command number.

When planning to use a compiled database, it is necessary to declare both $1 and $2 as Longints, even if you do not use them.

If you want the initial command to be executed, you need to include the following in the called method: WR EXECUTE COMMAND($1;$2).

Example

You want to save your documents in the “Archive” folder located on your hard disk:

```
    C_LONGINT($1;$2)
    Case of
    : ($2=wr cmd save as )  `When Save As... is selected
       $DocName:=Request("Give a name to your document: ")
       If ((OK=1) & ($DocName#""))
             `Save the document in the selected folder
          WR SAVE DOCUMENT ($1;"HDisk:Archives:"+$DocNom)  `Mac
          WR SAVE DOCUMENT ($1;"D:\Archives\"+$DocNom)   `Win
       Else
          BEEP  `Something is not correct
       End if
    Else   `For any other menu command
       WR EXECUTE COMMAND ($1;$2)
          `Execute the regular action
    End case

       ` Form Method:
    If (Form event=On Load )
⇒      WR ON COMMAND (Area;"TheMethod")
    End if
```

See also

WR EXECUTE COMMAND, WR Get on command method.

WR Get on command method

WR Get on command method (area) → String

Parameter	Type		Description
area	Longint	→	4D Write area
Function result	String	←	Name of installed on command method

Description

The WR Get on command method command returns the name of the method installed by WR ON COMMAND for the 4D Write area.

If no on command method has been installed, an empty string ("") is returned.

See also

WR ON COMMAND.

WR REDRAW

version 6.0

WR REDRAW (area)

Parameter	Type		Description
area	Longint	→	4D Write area

Description

The WR REDRAW command causes area to be redrawn. This command is useful when you have disabled screen updating with the WR UPDATE MODE command and now want to redraw a 4D Write area to show how previously executed code has modified the area.

Example

The following example turns off screen updates, calls the *Reformat* project method that reformats area, and then redraws area without turning screen updating back on.

```
	WR UPDATE MODE (area;0)
	   `Turn off screen updating
	Reformat (area)
	   `area can be passed to a method
⇒	WR REDRAW (area)
	   `Redraw to display changes
```

See Also

WR UPDATE MODE.

WR SET DOC PROPERTY

version 6.5 (Modified)

WR SET DOC PROPERTY (area; property; value)

Parameter	Type		Description
area	Longint	→	4D Write area
property	Integer	→	Number of the property to set
value	Number	→	Value for the selected property

Description

The WR SET DOC PROPERTY command allows you to modify the document properties in the 4D Write area referenced by area.

The meaning given to the value parameter depends on the property value used. property and value can be set using constants.

The list of text properties and their references are available in Appendix D: 4D Write Constants, in the "WR Document properties" theme.

The constants which can be used with WR SET DOC PROPERTY and WR Get doc property are the following:

Constants (value)	Allows setting or getting:
wr first page (0)	the first page number (1 by default). If you set, for example, the value 10, the 2nd page will be number 11, etc.
wr view mode (1)	the document view mode (Page = 0, Normal =1)
wr view rulers (2)	the display status of the ruler (Displayed = 1, Hidden = 0)
wr view frames (3)	the display status of text frames (Displayed = 1, Hidden = 0)
wr view headers (4)	the display status of headers (Displayed = 1, Hidden = 0), does not apply to the first page header if it is different from others (use 'wr view first page header')
wr view footers (5)	the display status of footers (Displayed = 1, Hidden = 0), does not apply to the first page footer if it is different from others (use 'wr view first page footer')
wr view pictures (6)	the display status of pictures (Displayed = 1, Hidden = 0)
wr view Hscrollbar (7)	the display status of horizontal scrollbars (Displayed = 1, Hidden = 0)
wr view Vscrollbar (8)	the display status of vertical scrollbars (Displayed = 1, Hidden = 0)

wr view statusbar (9)	the display status of the status bar (Displayed = 1, Hidden = 0)
wr view menubar (10)	the display status of the menu bar (Displayed = 1, Hidden = 0)
wr view standard palette (11)	the display status of the standard tool palette (Displayed = 1, Hidden = 0)
wr view format palette (12)	the display status of the format toolbar (Displayed = 1, Hidden = 0)
wr view style palette (13)	the display status of the style toolbar (Displayed = 1, Hidden = 0)
wr view borders palette (14)	the display status of the borders toolbar (Displayed = 1, Hidden = 0)
wr view invisible chars (15)	the display status of invisible characters (Displayed = 1, Hidden = 0)
wr view references (16)	the display status of references (Displayed = 1, Hidden = 0)
wr view column separators (17)	the presence of a vertical separator between columns in multi-columns mode - corresponds to the Vertical separator option in the Columns dialog box (Vertical separator = 1, No vertical separator = 0)
wr different on first page (18)	if headers and footers are different on first page - corresponds to the 'Different on first page' option in the Preferences dialog box (Yes = 1, No = 0)
wr different left right pages (19)	if headers and footers are different between left and right pages - corresponds to the 'Different on left and right pages' option in the Preferences dialog box (Different = 1, Similar = 0)
wr widow orphan (20)	if widow and orphan are taken into account - corresponds to the 'Widow and Orphan Control' option in the Preferences dialog box (Managed = 1, Ignored= 0)
wr unit (21)	the document current unit - corresponds to the 'Unit' pop up menu in the Preferences dialog box (Centimeters=0, Inches=1, Pixels= 2)
wr default tab (22)	the default "automatic" tab spacing expressed in the current document unit - corresponds to the 'Default Tab Spacing' area in the Preferences dialog box (by default 0.5 inches; 1.3 centimeters; 36 pixels)
wr language (23)	the language associated with the document (American English = 1033, Australian English = 3081, English = 2057, Catalan = 1027, Danish = 1030, Dutch = 1043, Finnish = 1035, French = 1036, French Canadian = 3084, German = 1031, Italian = 1040, Norwegian Bokmal = 1044, Norwegian Nynorsk = 2068, Portuguese Brazil = 1046, Portuguese Iberian = 2070, Spanish = 1034, Swedish = 1053, Russian = 1049, Czech = 1029, Hungarian = 1038, Polish = 1045)

wr number of columns (24)	the number of columns of the document
wr columns spacing (25)	the spacing value between each column expressed in the current document unit - corresponds to the 'Spacing' area of the Columns dialog box.
wr binding (26)	the binding size expressed in the current document unit - corresponds to the 'Binding' area in the Preferences dialog box
wr opposite pages (27)	the opposite pages mode of the document - corresponds to the 'Opposite pages' option in the Preferences dialog box (Opposite pages =1 , Normal pages=0)
wr right first page (28)	if the first page is a left page or a right page - right page by default (right page = 1, left page =0)
wr text inside margin (29)	the margin between the left side of the text and the left side of the paper for a right page, right sides for a left page, expressed in the current document unit
wr text outside margin (30)	the margin between the right side of the text and the right side of the paper for a right page, left sides for a left page, expressed in the current document unit
wr text left margin (29)	the margin between the left side of the page and the left side of the paper expressed in the current document unit
wr text right margin (30)	the margin between the right side of the page and the right side of the paper expressed in the current document unit

If the 'Different on first page' option in the Preferences dialog box has been selected, the following constants should be used for all pages except for the first one:

wr text top margin (31)	the margin between the top of the page body and the top edge of the paper expressed in the current document unit, use 'wr first page top margin' for the first page if different from others
wr text bottom margin (32)	the margin between the bottom of the page body and the bottom edge of the paper expressed in the current document unit, use 'wr first page bottom margin' for the first page if different from others
wr header top margin (33)	the margin between the top of the page header and the top edge of the paper expressed in the current document unit, use 'wr header 1st page top margin' for the first page if different from others
wr header bottom margin (34)	the margin between the bottom of the page header and the top edge of the paper expressed in the current document unit, use 'wr header 1st page bottom mg' for the first page if different from others

wr footer top margin (35)	the margin between the top of the page footer and the bottom edge of the paper expressed in the current document unit, use 'wr footer 1st page top margin' for the first page if different from others
wr footer bottom margin (36)	the margin between the bottom of the page footer and the bottom edge of the paper expressed in the current document unit, use 'wr footer 1st page bottom mg' for the first page if different from others
wr paper width (37)	the paper width expressed in the current document unit (*)
wr paper height (38)	the paper height expressed in the current document unit (*)
wr dead left margin (39)	the non-printable area reserved by the printer on the left of the paper, expressed in the current document unit (this value cannot be set; it can only be read) (*)
wr dead top margin (40)	the non-printable area reserved by the printer at the top of the paper, expressed in the current document unit (this value cannot be set; it can only be read) (*)
wr printable width (41)	the horizontal printable area starting from the dead left margin (this value cannot be set; it can only be read). The right dead margin equals the paper width; the left dead margin-the printable width
wr printable height (42)	the vertical printable area starting from the top left margin (this value cannot be set; it can only be read). The bottom dead margin equals the paper height; the top dead margin-the printable height
wr data size (43)	the size of the document in bytes (this value cannot be set; it can only be read)
wr undo buffer size (44)	the size of the undo buffer in bytes (this value cannot be set; it can only be read)
wr horizontal splitter (45)	the display status of the horizontal splitter (Displayed = 1, Hidden = 0)
wr vertical splitter (46)	the display status of the vertical splitter (Displayed = 1, Hidden = 0)
wr links color (47)	the color of the hyperlinks, while they are not visited
wr visited links color (48)	the color of the hyperlinks once they have been visited
wr view frame area (49)	the presence of a frame around the area in the form (frame = 1, no frame = 0)

The following constants (50 to 57) should be used for the first page of your document when the 'Different on first page' option in the Preferences dialog box has been set.

wr view first page header (50)	the display status of the first page header (Displayed = 1, Hidden = 0), use 'wr view headers' for the other pages
wr view first page footer (51)	the display status of the first page footer (Displayed = 1, Hidden = 0), use 'wr view footers' for the other pages
wr first page top margin (52)	the margin between the top of the first page body and the top edge of the paper expressed in the current document unit, use 'wr text top margin' for the other pages
wr first page bottom margin (53)	the margin between the bottom of the first page body and the bottom edge of the paper expressed in the current document unit, use 'wr text bottom margin' for the other pages
wr header 1st page top margin (54)	the margin between the top of the first page header and the top edge of the paper expressed in the current document unit, use 'wr header top margin' for the other pages
wr header 1st page bottom mg (55)	the margin between the bottom of the first page header and the top edge of the paper expressed in the current document unit, use 'wr header bottom margin' for the other pages
wr footer 1st page top margin (56)	the margin between the top of the first page footer and the bottom edge of the paper expressed in the current document unit, use 'wr footer top margin' for the other pages
wr footer 1st page bottom mg (57)	the margin between the bottom of the first page footer and the bottom edge of the paper expressed in the currentdocument unit, use 'wr footer bottom margin' for the other pages
wr draft mode (58)	the document text entry mode (1 = draft mode, 0 = WYSIWYG mode)
wr column width (59)	the column width expressed in the current document unit (this value cannot be set; it can only be read).

(*) When you set the paper size programmatically, 4D Write will consider that a "virtual" printer device is used. The program will set the dead margins to zero and the printable area will be equal to the paper size. This feature is useful for documents which are not intended to be printed.

Examples

(1) You want to display a 4D Write area on screen without its menus and rulers:

```
   If(Form event=On load)
⇒     WR SET DOC PROPERTY(Area;wr view menubar;0)
⇒     WR SET DOC PROPERTY(Area;wr view rulers;0)
   End if
```

(2) This method allows the user to display or hide the scroll bars:

```
   C_LONGINT(ScrollStatus)
   ScrollStatus:=WR Get doc property(Area;wr Hscrollbar)   `Constant=7
   ScrollStatus:=ScrollStatus+WR Get doc property(Area;wr Vscrollbar)   `Constant=8
   If (ScrollStatus>0)
      CONFIRM("At least one scroll bar is displayed, do you want to hide them?")
      If (OK=1)
⇒        WR SET DOC PROPERTY(Area;wr Hscrollbar;0)
⇒        WR SET DOC PROPERTY(Area;wr Vscrollbar;0)
      End if
   Else
      CONFIRM("Scroll bars are hidden, do you want to display them?")
      If (OK=1)
⇒        WR SET DOC PROPERTY(Area;wr Hscrollbar;1)
⇒        WR SET DOC PROPERTY(Area;wr Vscrollbar;1)
      End if
   End if
```

See also

WR Get doc property.

WR UPDATE MODE

version 6.0

WR UPDATE MODE (area; mode)

Parameter	Type		Description
area	Longint	→	4D Write area
mode	Integer	→	0=No update 1=Update

Description

The WR UPDATE MODE command allows the designer to enable and disable screen updating in area. If mode equals 0, screen updating is off. If mode equals 1, screen updating is on. This command only affects screen updates caused by 4D Write commands. User actions in area will continue to update the screen correctly.

When screen updating is turned off, 4D Write commands execute faster. For example, if you intend to execute a series of modifications to a 4D Write area, turn off updating before beginning the modifications and then turn updating on when you are finished. The commands execute faster as well as the screen redraw.

Example

The following example turns off screen updating, calls the *Reformat* project method that makes several modifications, and then turns screen updating back on:

```
⇒	WR UPDATE MODE (area;0)
	Reformat (Area)
⇒	WR UPDATE MODE (area;1)
```

See Also

WR REDRAW.

3

WR Area Options

Area Options, Introduction

WR Area Options

version 6.5 (Modified)

The commands and functions of the "WR Area Options" theme enable you to set the type of environment available to users. For example, using the WR SET CURSOR POSITION command you can place the cursor at a specific location in a 4D Write document.

You can also prevent users from modifying a 4D Write area (WR TEXT ACCESS) and build a picture preview of an area (WR Build preview).

WR Build preview

WR Build preview (area; page) → Picture

Parameter	Type		Description
area	Longint	→	4D Write area
page	Longint	→	Number of the page to pass as a picture
Function result	Picture	←	Picture of the page

Description

The WR Build preview command converts the page, whose number is passed in page, into a picture. The page number takes into account the page numbering as it was defined in the preferences dialog.

The picture can be stored, for instance, in a 4D picture field or in a 4D picture variable. The picture is the same size as the page. You can set the size of the picture by using the WR SET DOC PROPERTY command and by passing a value for <u>wr paper width</u> and <u>wr paper height</u>.

Note: unlike when you use WR Area to picture, the picture does not contains any 4D Write data

The returned picture is a vector-based picture. A picture that was created on Windows cannot be directly displayed on Mac OS, nor stored “as is”in a picture file (for example, using the WRITE PICTURE FILE command) since it uses the EMF format. If you want your Windows pictures to be displayed on Mac OS or in another Windows application, you need to convert the picture into a bitmap by using the following statement: myPicture:=myPicture|myPicture.

Unlike EMF (Windows only), Pict and bitmap picture types are not platform dependent.

Note: On the contrary, Mac OS pictures can be used directly.

Example

4D Write documents are saved into BLOB fields. You only want to print only the second page of each document. To do so, insert in the print form a picture variable (named *MyImage* in this example) and attach the following method to the variable:

```
    If(Form event=On Printing Detail)
        WR BLOB TO AREA (NewOffscreen;[MyTable]WriteBlob_)
⇒       MyImage:=WR Build preview (NewOffscreen;2)
    End if
```

Then, create and execute the following project method:

```
QUERY([MyTable])   `Creating the selection to print
OUTPUT FORM([MyTable];"PrintPage2")   `PrintPage2 is the form used for printing
    `Creating the offscreen area used in the previous method
NewOffscreen:=WR New offscreen area
PRINT SELECTION([MyTable])   `Printing the selection
WR DELETE OFFSCREEN AREA(NewOffscreen)    `Deleting the offscreen area
```

See also

WR SET DOC PROPERTY.

WR GET AREA PROPERTY

version 2003 (Modified)

WR GET AREA PROPERTY (area; option; value; stringValue)

Parameter	Type		Description
area	Longint	→	4D Write area
option	Integer	→	Option number
value	Integer	←	0 or 1 depending on the option
stringValue	String	←	Property string depending on the case

Description

The WR GET AREA PROPERTY command allows you to read various options for the 4D Write area referenced by area.

The options that can be read are the following :

		Values		
N°	Options/constants	Num	String	Effect
0	Confirm dialog	0	-	No dialog
	wr confirm dialog	1	-	Dialog
1	Picture preview	0	-	No preview
	wr save preview	1	-	Preview
2	Saving Redo	0	-	No buffer
	wr allow undo	1	-	Actions are stored
3	Dirty bit except if area = 0	0	-	False
	wr modified	1	-	True
4	Variable size printing (unless area = 0)	0	-	Variable size
	wr fixed print size	1	-	Fixed size
5	4D Write 6.0 field conversion dialog (if area = 0)	0	-	No dialog
	wr convert dialog	1	-	Dialog
6	Button title when area is minimized	0	-	Default title
	wr minimized button title	1	Title	
7	4D Write Window title	0	-	area name
	(when going to full screen or in external window)	1	Title	
	wr window title			
8	Minimum area width before switching to button	XX	-	in pixels
	wr minimum width			
9	Minimum area height before switching to button	XX	-	in pixels
	wr minimum height			
10	Saving the templates on the server in C/S	0	-	on client
	wr save template on server	1	-	on server

11	Loading templates from server in C/S wr load template on server	0 1	- -	on client on server
12	Interpreting field references during document conversion wr convert by token	0 1	- -	Names (default) Numbers

Note: option can be set using constants.

The list of text properties and their references are available in Appendix D: 4D Write Constants, in the "WR Area properties" theme. You can either pass the value or the constant.

See also

Appendix D: 4D Write Constants, WR SET AREA PROPERTY.

WR GET CURSOR COORDINATES

version 6.5.2

WR GET CURSOR COORDINATES (area; posHoriz; posVert; height)

Parameter	Type		Description
area	Longint	→	4D Write area
posHoriz	Real	←	Horizontal position in the page
posVert	Real	←	Vertical position in the page
height	Real	←	Height of the cursor

Description

The WR GET CURSOR COORDINATES command returns the coordinates of the cursor in relation to the upper left corner of the page. These values are expressed in the current default unit for the document.

When the command is executed with a text or a picture selected in the area, two cases can occur:
• If the selection has been made programmatically, the cursor is considered to be set at the end of the selection.
• If the selection has been made manually, the cursor is considered to be set at the mouse button release location. For example, if a paragraph has been manually selected by dragging the mouse from the last line to the first line, the cursor position will be set at the beginning of the selection.

The height parameter returns the current height of the cursor. If only a picture is selected, the height of the picture is returned.

See Also

WR GET CURSOR POSITION.

WR GET CURSOR POSITION

version 6.5

WR GET CURSOR POSITION (area; page; column; line; position)

Parameter	Type		Description
area	Longint	→	4D Write area
page	Longint	←	Number of the page where the selection is
column	Longint	←	Number of the column where the selection is
line	Longint	←	Number of the line in the column
position	Longint	←	Position of the selection in the current line

Description

The WR GET CURSOR POSITION command returns the position of the selection in the 4D Write area referenced by area.

• page: page is between the number of the first page and the number of the last page of the document. These numbers take into account the custom page numbering, if any.

• column: This value is between 1 and the total number of columns.

• line: This value is between 1 and the total number of lines in the column.

• position: This value is between 1 and the total number of characters in the line.

If the selection contains several characters, the position of the first character is returned. You can later go back to this location, using the WR SET CURSOR POSITION command with the same parameters.
You can use WR Get frame to determine which area the cursor is in.

Example

You want the user to be able to insert a logo in the header of the document, without losing the current position of the cursor in the text. To do this, attach the following method to the insertion button:

```
         C_LONGINT($frame;$Col;$Line;$Pos)
         C_REAL($PictWidth;$PictHeight;$OrigWidth;$OrigHeight;$HeadTopMargin)
            `Which frame of the document is the cursor in?
         $frame:=WR Get frame(Area)
            `Getting current cursor position
⇒        WR GET CURSOR POSITION (Area;$Page;$Col;$Line;$Pos)
            `Switching the current area to the header of the document
         WR SET FRAME (Area;wr right header)
            `Loading the record that contains the logo to include
         ALL RECORDS([Interface])
            `Inserting the logo
         WR INSERT PICTURE(Area;[Interface]Logo;0)
            `Selecting the logo and getting its size
         WR SELECT(Area;4;1)
⇒        WR GET PICTURE SIZE(Area;$PictWidth;$PictHeight;$OrigWidth;$OrigHeight)
            `The height of the header must fit the picture
         $HeadTopMargin:=WR Get doc property(Area;wr header top margin)
         WR SET DOC PROPERTY(Area;wr text top margin;$HeadTopMargin+$PictHeight)
         WR SET DOC PROPERTY(Area;wr header bottom margin;$PictHeight)
            `Then going back to the frame the cursor was in
         WR SET FRAME(Area;$frame)
            `Putting the cursor back in its original position
         WR SET CURSOR POSITION(Area;$Page;$Col;$Line;$Pos)
```

See also

WR GET CURSOR COORDINATES, WR Get frame, WR SET CURSOR POSITION.

WR Get frame

WR Get frame (area) → Longint

Parameter	Type		Description
area	Longint	→	4D write area
Function result	Longint	←	Page area in which the cursor is

Description

The WR Get frame command returns a number that represents which page area the insertion point or the current selection is in.

The following values can be returned:

Value	Location
0	text area
1	right header
2	right footer
3	left header
4	left footer
5	first header
6	first footer

You can enter these values by number or by using a predefined constant (as shown).

Examples

See the examples for the WR GET CURSOR POSITION and WR SET CURSOR POSITION commands.

See also

Appendix D: 4D Write Constants, WR SET FRAME.

WR SET AREA PROPERTY

version 2003 (Modified)

WR SET AREA PROPERTY (area; option; value{; stringValue})

Parameter	Type		Description
area	Longint	→	4D Write area
option	Integer	→	Option number
value	Integer	→	0 or 1, depending on the option
stringValue	String	→	String for the property, depending on the option

Description

The WR SET AREA PROPERTY command allows you to modify the properties of the 4D Write area referenced by area.

If area equals 0, the WR SET AREA PROPERTY command will apply to each active 4D Write area. In this case, it is recommended that your code should call this command in the On Startup Database Method.

The options that can be set are the following:

		Values		
N°	**Options/constants**	**Num**	**String**	**Effect**
0	Confirm dialog	0	-	No dialog
	<u>wr confirm dialog</u>	1	-	Dialog
1	Picture preview	0	-	No preview
	<u>wr save preview</u>	1	-	Preview
2	Saving Redo	0	-	No buffer
	<u>wr allow undo</u>	1	-	Actions are stored
3	Dirty bit except if area = 0	0	-	False
	<u>wr modified</u>	1	-	True
4	Variable size printing (unless area = 0)	0	-	Variable size
	<u>wr fixed print size</u>	1	-	Fixed size
5	4D Write 6.0 field conversion dialog (if area = 0)	0	-	No dialog
	<u>wr convert dialog</u>	1	-	Dialog
6	Button title when area is minimized	0	-	Default title
	<u>wr minimized button title</u>	1	Title	
7	4D Write Window title	0	-	area name
	(when going to full screen or in external window)	1	Title	
	<u>wr window title</u>			
8	Minimum area width before switching to button	XX	-	in pixels
	<u>wr minimum width</u>			
9	Minimum area height before switching to button	XX	-	in pixels
	<u>wr minimum height</u>			

10	Saving the templates on the server in C/S wr save template on server	0 1	- -	on client on server
11	Loading templates from server in C/S wr load template on server	0 1	- -	on client on server
12	Interpreting field references during document conversion wr convert by token	0 1	- -	Names (default) Numbers

Note: option can be set using constants.

The list of text properties and their references are available in Appendix D: 4D Write Constants, in the "WR Area properties" theme.

Example

You want to disable the automatic picture preview of the area, the display of the confirm dialog and the Undo command from the Edit menu:

⇒ ***WR SET AREA PROPERTY***(Area;wr save preview;0)
⇒ ***WR SET AREA PROPERTY***(Area;wr confirm dialog;0)
⇒ ***WR SET AREA PROPERTY***(Area;wr allow undo;0)

See also

Appendix D: 4D Write Constants, WR GET AREA PROPERTY.

WR SET CURSOR POSITION

WR SET CURSOR POSITION (area; page; column; line; position)

Parameter	Type		Description
area	Longint	→	4D Write area
page	Longint	→	Page number
column	Longint	→	Column number
line	Longint	→	Line number
position	Longint	→	Horizontal position of the cursor in the line

Description

The WR SET CURSOR POSITION command moves the insertion point to a new position specified by page, column , line and position.

• page: The value for page must be between the first and the last page numbers of the document. The page number must take into account the page numbering as it was defined in the preferences dialog.

• column: The value for column must be between 1 and the total number of columns.

• line: The value for line must be contained between 1 and the total number of lines of the column (or page, if there is only one column).

• position: This value must be contained between 1 and the total number characters in the line. To move the insertion point to the first position in the line, set position to 1.

If you want to place the cursor in an area other than the body area, you need to use the WR SET FRAME command before using the WR SET CURSOR POSITION command.

Example

You want to move the insertion point to the beginning of the 10th line of the 4th page:

```
        `Making sure that we are in the body area of the document
    If (WR Get frame (Area)#0)
            `Otherwise, moving to the body area
        WR SET FRAME (Area;wr body)
    End if
        `Moving the cursor
⇒   WR SET CURSOR POSITION(Area;10;1;10;1)
        `Scrolling area to display the insertion point
    WR SCROLL TO SELECTION(Area)
```

See also

WR GET CURSOR POSITION, WR SET FRAME.

WR SET FRAME

WR SET FRAME (area; frame)

Parameter	Type		Description
area	Longint	→	4D Write area
frame	Integer	→	Frame number

Description

The WR SET FRAME command places the insertion point at its previous location in the part of the 4D Write area area indicated by the frame parameter. This position was previously memorized by 4D Write. If the Normal view mode is selected and the insertion point is placed in an header or footer area, 4D Write automatically switches to Page view mode.

You can pass the following values or constants in frame:

Value	Constants
0	wr text frame
1	wr right header
2	wr right footer
3	wr left header
4	wr left footer
5	wr first header
6	wr first footer

Values 3 and 4 are to be used when you use different headers and footers for left and right pages.
Values 5 and 6 are to be used when you use different headers and footers for the first page.

Note: The list of values is also available in Appendix D: 4D Write Constants.

Examples

See the examples for the following commands: WR GET CURSOR POSITION, WR SET CURSOR POSITION and WR INSERT PAGE NUMBER.

See also

WR Get frame.

WR TEXT ACCESS

version 6.0

WR TEXT ACCESS (area; mode)

Parameter	Type		Description
area	Longint	→	4D Write area
mode	Integer	→	0=Allow access 1=Restrict access

Description

The command WR TEXT ACCESS enables you to control access to the text in Area. If mode equals 0, 4D Write allows modifications to area. If mode equals 1, 4D Write displays area in read-only mode.
When an area is displayed in read-only mode, the menus, rulers, and Zoom box are not present. The text can be seen and scrolled but not modified. When access to a formerly restricted area is changed, you must call WR SET DOC PROPERTY (Area;wr view menubar;1) and WR SET DOC PROPERTY (Area;wr view rulers;1) to display the ruler and menu bar.

Example

The following example is the form method of the form that contains area. It sets area to read-only when the form is loaded.

```
	If (Form event=On load)
⇒		WR TEXT ACCESS (area;1)
	End if
```

See Also

WR SET DOC PROPERTY.

4

WR Areas

Areas, Introduction

WR Areas

version 6.5 (Modified)

The commands and functions of the "WR Areas" theme allow you to manage 4D Write areas, wherever they are located — in 4D forms and stored in BLOBs or Picture fields, or in offscreen areas.

For example, the WR PICTURE TO AREA command loads the picture passed as parameter from a field or places a 4D Write document in an offscreen area.

WR Area to blob

version 6.5

WR Area to blob (area{; savedDoc}) → BLOB

Parameter	Type		Description
area	Longint	→	4D Write area
savedDoc	Integer	→	1=If document is not saved, no dialog 0=If document is not saved, the dialog is displayed
Function result	BLOB	←	Contents of area

Description

The WR Area to blob command places the contents of the area referenced by area into a BLOB field or variable. WR Area to blob returns a Blob that can be assigned to a BLOB field or a BLOB variable.

• If savedDoc equals 0, and the document has been modified since it was last saved, a dialog will be displayed asking the user if they wish to save the document.
• If savedDoc equals 1, the document will be considered as saved and the user will not be prompted to save it.
• If savedDoc is omitted, default settings will be applied.

Example

You want to save Area in the BLOB field “WriteBlobSave” :

⇒ [Texts]WriteBlobSave:=***WR Area to blob***(Area;1)

See also

WR Area to picture, WR BLOB TO AREA.

WR Area to picture

WR Area to picture (area{; savedDoc{; preview}}) → Picture

Parameter	Type		Description
area	Longint	→	4D Write area
savedDoc	Integer	→	1 = if document is not saved, no dialog 0 = if document is not saved, the dialog is displayed
preview	Integer	→	1 = the picture is created 0 = the picture is not created
Function result	Picture	←	Picture of the contents of area

Description

The WR Area to picture command allows you to place the contents of the area referenced by area in a picture field or variable. Passing a 4D Write area to the WR Area to picture command returns a picture that can later be assigned to a picture field or a picture variable.

savedDoc
• If savedDoc equals 0, and the document has been modified since it was last saved, a dialog will be displayed asking the user if they wish to save the document.
• If savedDoc equals 1, the document will be considered as saved and the user will not be prompted to save it.

preview
• If preview equals 0, no picture preview will be created.
• If preview equals 1, a picture preview will be created.

Note: If no picture preview is created, the picture cannot be displayed.

If optional parameters are omitted, the default settings for area will be applied.

Examples

(1) You want to save Area as well as its preview picture in the Picture field “WritePictSave”:

⇒ [Texts]WritePictSave:=***WR Area to picture***(Area;1;1)

(2) You want to save the current text selection in a record of the [Templates] table:

```
    WR EXECUTE COMMAND(Area;wr cmd copy)   `Copying the selection
    CREATE RECORD([Templates])   `Creating a record in [Templates]
    Tempo:=WR New offscreen area   `Creating an offscreen area
    WR EXECUTE COMMAND(Tempo;wr cmd paste)   `Pasting selection in the area
       `Saving the result in the [Templates]Text_ field
⇒   [Templates]Text_:=WR Area to picture(Tempo)
    WR DELETE OFFSCREEN AREA (Tempo)   `Deleting the temporary area
    SAVE RECORD([Templates])   `Saving the record in [Templates]
```

See also

WR Area to blob, WR PICTURE TO AREA.

WR BLOB TO AREA

WR BLOB TO AREA (area; blob)

Parameter	Type		Description
area	Longint	→	4D Write area
blob	BLOB	→	Variable or field that contains 4D Write data

Description

The WR BLOB TO AREA command loads into the 4D Write area area the contents of blob. The contents of the BLOB are assumed to be 4D Write data.

The contents of the Blob can either be data that was automatically saved from a 4D Write area associated by name with a BLOB, or data that was saved using the WR Area to blob command.

Examples

(1) You want to load a template of letter which is stored in the "[Templates]Reference_" BLOB field and use it as the current template:

```
        QUERY([Templates];[Templates]Texts=Ref)
        If(Records in selection([Templates])>0)
⇒           WR BLOB TO AREA(Area;[Templates]Reference_)
        End if
```

(2) You want to copy the text stored in the "[Templates]TheText_" BLOB field and paste it in the current area on screen. This example shows you how to create an advanced glossary system:

```
        Temp:=WR New offscreen area
⇒       WR BLOB TO AREA (Temp;[Templates]TheText_)    `Expanding the field
        WR EXECUTE COMMAND(Temp;wr wmd select all)
        WR EXECUTE COMMAND(Temp;wr cmd copy)
        WR DELETE OFFSCREEN AREA (Temp)    `Deleting the area
        WR EXECUTE COMMAND(Area;wr cmd paste)    `Executing the Paste menu command
```

Note: If you store the 4D Write areas into Picture fields, please refer to the description of the command WR PICTURE TO AREA.

See also

WR Area to blob.

WR DELETE OFFSCREEN AREA

WR DELETE OFFSCREEN AREA (area)

Parameter	Type		Description
area	Longint	→	4D Write area

Description

The command WR DELETE OFFSCREEN AREA deletes the 4D Write area that was created with WR New offscreen area and frees the memory used by the offscreen area.
area must be an offscreen area and not an area on a form or in a window. Issue the WR DELETE OFFSCREEN AREA command when you no longer need the offscreen area.

Example

The following example illustrates the need to pair every call to WR New offscreen area with a corresponding call to WR DELETE OFFSCREEN AREA.

```
	NewArea:=WR New offscreen area
		`Create a new offscreen area
		`Do Something
⇒	WR DELETE OFFSCREEN AREA (NewArea)
		`Remove the offscreen area
```

See Also

WR New offscreen area.

WR New offscreen area

WR New offscreen area → Longint

Parameter	Type		Description
This command does not require any parameters			
Function result	Longint	←	Reference of 4D Write area

Description

The command WR New offscreen area reserves space in memory for a 4D Write area that is invisible to you and the user. This function also returns a value that can be used to access the invisible area. The value returned by WR New offscreen area can be used in any 4D Write command that requires a 4D Write area.
Remember to delete the offscreen area created by this function when you are finished with it.

Example

The following example creates a temporary offscreen area, prints it and the deletes it.

```
⇒	Temporary:=WR New offscreen area
	WR INSERT TEXT(Temporary;MyText)
	WR PRINT(Temporary;0)
	WR DELETE OFFSCREEN AREA(Temporary)
```

See Also

WR DELETE OFFSCREEN AREA.

WR PICTURE TO AREA

version 6.5

WR PICTURE TO AREA (area; picture)

Parameter	Type		Description
area	Longint	→	4D Write area
picture	Picture	→	Field or variable

Description

The WR PICTURE TO AREA command allows you to read a picture variable or a picture field that contains a 4D Write document and to open it in the 4D Write area referenced by area. area can either be an area currently displayed or an offscreen area.

This command allows you, for instance, to read 4D Write documents that were saved in different tables.

Note: This command also reads the 4D Write version 6.0.x file format.

Examples

(1) You want to load a letter template stored in the "[Templates]Reference" Picture field and use it as the current template:

```
     QUERY([Templates];[Templates]Reference=Ref)
     If(Records in selection([Templates])>0)
⇒        WR PICTURE TO AREA(Area;[Templates]Reference_)
     End if
```

(2) You want to copy the text stored in the "[Templates]TheText_" Picture field and paste it in the current area on screen. This example shows you how to create an advanced glossary system:

```
     Temp:=WR New offscreen area
⇒    WR PICTURE TO AREA (Temp;[Templates]TheText_)    `Expanding the field
     WR EXECUTE COMMAND(Temp;wr cmd select all)
     WR EXECUTE COMMAND(Temp;wr cmd copy)
     WR DELETE OFFSCREEN AREA (Temp)    `Deleting the area
     WR EXECUTE COMMAND(Area;wr cmd paste)    `Executing the Paste menu command
```

Note: If you store 4D Write areas in BLOB fields, please refer to the description of the command WR BLOB TO AREA.

See also

WR Area to picture.

5

WR Database Objects

Database Objects, Introduction

The commands and functions of the "WR Database Objects" theme allow you to access 4th Dimension objects. These objects can be methods, variables, functions, fields, page numbers, or 4D Write picture areas.

You can also retrieve information on these objects, when they are placed in a 4D Write area, by using the WR GET REFERENCE command.

WR GET REFERENCE

version 6.7 (Modified)

WR GET REFERENCE (area; info1; info2; name; type{; numFormat{; dateFormat{; timeFormat}}})

Parameter	Type		Description
area	Longint	→	4D Write area
info1	Integer	←	First information regarding the reference
info2	Integer	←	Second information regarding the reference
name	String	←	Receives reference name
type	Integer	←	Receives reference type
numFormat	String	←	Numeric format
dateFormat	Integer	←	Number of the date format
timeFormat	Integer	←	Number of the time format

Description

The WR GET REFERENCE command gets information about the selected reference in the 4D Write area.
Information about the selected reference is returned into the info1, info2, name and type parameters. You can also find out the display format of numeric, Date or Time inserted references.
Values returned in info1, info2, and name depend on the value in type. If the selected object is not a reference, type returns 0.

• If type=1, the reference is a field. info1 indicates the table number. info2 indicates the field number. name is empty.

• If type=2, the reference is an expression. info1 and info2 contain the value 0. name contains the name of the variable or expression.

The numFormat parameter returns a string indicating the format of the selected numeric field/expression (i.e., Real, Integer, or Longint). If no format is associated with the expression or if it is not a numeric type expression, an empty string is returned.

The dateFormat parameter returns the number of the Date format associated with the selected field/expression, if it is a date type. Should this not be the case, the value 0 is returned.

Following are the format codes for dates:

Date format	Name	Number
1/6/00	(Short)	1
Thu, Jan 6 2000	(Abbreviated)	2
Thursday, January 6 2000	(Long)	3
01/06/2000	(MM DD YYYY)	4
January 6, 2000	(Month Day Year)	5
Jan 6, 2000	(Abb Month Day)	6
01/06/2000	(MM DD YYYY Forced)	7

The timeFormat parameter returns the number of the time format associated with the selected field/expression, if it is a time type. Should this not be the case, the value 0 is returned.
Following are the format codes for times:

Time format	Number
HH:MM:SS	1
HH:MM	2
HH hours MM minutes SS seconds	3
HH hours MM minutes	4
HH:MM AM PM	5

Example

This example determines if the user selected an object that is a reference. It also tells the user if the selected object is a field or an expression.

```
⇒      WR GET REFERENCE (Letter;$Table;$Field;$Name;$Type)
       Case of
          : ($Type=0)    `Text or nothing
             ALERT("Selected text or nothing")
          : ($Type=1)
             ALERT("Selected the field "+Field name($Table;$Field))
          : ($Type=2)
             ALERT("Selected the expression named "+$Name)
       End case
```

See Also

WR INSERT EXPRESSION, WR INSERT FIELD.

WR INSERT EXPRESSION

WR INSERT EXPRESSION (area; expression{; numFormat{; dateFormat{; timeFormat}}})

Parameter	Type		Description
area	Longint	→	4D Write area
expression	String	→	Expression to insert
numFormat	String	→	Numeric format
dateFormat	Integer	→	Number of the date format
timeFormat	Integer	→	Number of the time format

Description

The WR INSERT EXPRESSION command inserts a reference to expression into area, replacing any currently selected text.

expression must be a valid 4th Dimension expression that returns a value. expression can be a 4th Dimension variable, function, or statement that returns a value; an external function or a user-defined function (project method); or a picture variable. If expression is a variable, you should pass its name between double quotes ("").
If expression returns a value that includes carriage returns and tabs, 4D Write formats the text according to the ruler of the paragraph in which expression resides.

The numFormat optional parameter indicates the format of numeric expressions (i.e. Real, Integer, or Longint). It can contain any numeric display format, whether it exists or not (for example “###,##”). Put an empty string when this parameter is not appropriate or omit it if the following two parameters have been omitted.

The dateFormat optional parameter indicates the format of Date type expressions. It must contain a number that indicates an existing date format. Put 0 when this parameter is not appropriate or omit it if the following parameter has been omitted.
Following are the format codes for dates:

Date format	Name	Number
1/6/00	(Short)	1
Thu, Jan 6 2000	(Abbreviated)	2
Thursday, January 6 2000	(Long)	3
01/06/2000	(MM DD YYYY)	4
January 6, 2000	(Month Day Year)	5
Jan 6, 2000	(Abb Month Day)	6
01/06/2000	(MM DD YYYY Forced)	7

The timeFormat optional parameter indicates the format of Time type expressions. It must contain a number indicating an existing time format. Put 0 when this parameter is not appropriate or omit it.

Following are the format codes for times:

Time format	Number
HH:MM:SS	1
HH:MM	2
HH hours MM minutes SS seconds	3
HH hours MM minutes	4
HH:MM AM PM	5

Example

The following two-part example shows a reference to a 4th Dimension project method inserted into a 4D Write area. The project method finds a customer's related invoices and concatenates the invoice numbers and amounts.

```
    `Project method SHOW INVOICES
$Tab:=Char(Tab Key)
$CR:=Char(Return Key)
RELATE MANY ([Customers])
FIRST RECORD ([Invoices])
$0:=""
For ($i;1;Records in selection([Invoices]))
    $0:=$0+[Invoices]Number+$Tab+String([Invoices]Amount;"$###,##0.00")+$CR
    NEXT RECORD ([Invoices])
End for
```

The second part of this example shows the insertion of the *SHOW INVOICES* project method into area. When 4D Write displays or prints area, each invoice will appear in a separate line.

⇒ ***WR INSERT EXPRESSION*** (area;"SHOW INVOICES")

See Also

WR GET REFERENCE, WR INSERT FIELD.

WR INSERT FIELD

version 6.7 (Modified)

WR INSERT FIELD (area; table; field{; numFormat{; dateFormat{; timeFormat}}})

Parameter	Type		Description
area	Longint	→	4D Write area
table	Integer	→	Table number
field	Integer	→	Field number
numFormat	String	→	Numeric format
dateFormat	Integer	→	Number of the date format
timeFormat	Integer	→	Number of the time format

Description

The WR INSERT FIELD command inserts a reference to a field into area, replacing any selected text. The field is described by the table and field numbers. You can also specify the display format of inserted numeric, Date or Time fields.

The numFormat optional parameter indicates the format of numeric fields (i.e., Real, Integer, or Longint). It can contain any numeric display format, whether it exists or not (for example, "###,##"). Put an empty string when this parameter is not appropriate, or omit it if the following two parameters have been omitted.

The dateFormat optional parameter indicates the format of Date type fields. It must contain a number that indicates an existing date format. Put 0 when this parameter is not appropriate, or omit it if the following parameter has been omitted.
Following are the format codes for dates:

Date format	Name	Number
1/6/00	(Short)	1
Thu, Jan 6 2000	(Abbreviated)	2
Thursday, January 6 2000	(Long)	3
01/06/2000	(MM DD YYYY)	4
January 6, 2000	(Month Day Year)	5
Jan 6, 2000	(Abb Month Day)	6
01/06/2000	(MM DD YYYY Forced)	7

The timeFormat optional parameter indicates the format of Time type fields. It must contain a number indicating an existing time format. Put 0 when this parameter is not appropriate or omit it.

Following are the format codes for times:

Time format	Number
HH:MM:SS	1
HH:MM	2
HH hours MM minutes SS seconds	3
HH hours MM minutes	4
HH:MM AM PM	5

See Also

WR GET REFERENCE, WR INSERT EXPRESSION.

WR Insert picture area

WR Insert picture area (area; picture; where) → Longint

Parameter	Type		Description
area	Longint	→	4D Write area
picture	Picture	→	4D Write area picture to insert
where	Integer	→	1=Document end 0=Insertion point
Function result	Longint	←	Error code

Description

The WR Insert picture area command inserts the 4D Write document in Picture into area.

where describes the position at which the new text will be inserted.

If where equals 1, the text will be inserted at the end of the document.

If where equals 0, the text will be inserted at the current insertion point or will replace any currently selected text.

WR Insert picture area returns a long integer containing an error code.
If the insertion is successful, the value returned is 0. See Appendix C: Error Codes for error codes.

Example

The following example adds the signature of the sender to the end of the document:

```
	QUERY([Sender]; [Sender]Name=[Letter]Sender)
⇒	ErrorNum:=WR Insert picture area(area;[Sender]Signature_; 1)
```

See Also

WR Area to picture.

WR INSERT PAGE NUMBER

version 6.5.3 (Modified)

WR INSERT PAGE NUMBER (area; format{; typeNum})

Parameter	Type		Description
area	Longint	→	4D Write area
format	Integer	→	Format type
typeNum	Integer	→	Number to insert 0 = Page number, 1 = Total number of pages

Description

The WR INSERT PAGE NUMBER command allows you to insert, at the cursor location, a reference that displays the current page number or the total number of pages. This reference can be placed in the main text, footer or header area. You can use the WR SET FRAME command to place the cursor in whichever area you choose.

format allows you to choose the display format for the reference to insert. These formats are identical to the formats available in the Insert page number dialog.

Format Type	Value
1, 2, 3...	0
a, b, c...	1
A, B, C...	2
i, ii, iii...	3
I, II, III...	4

The typeNum optional parameter allows you to insert either the current page number or the total   page count of the current documet. If you pass 0 or if you omit this parameter, the current page number will be inserted. If you pass 1, the total number of pages of the document will be inserted.

Example

The following method (*OddPages)* is attached to a variable inserted in the footer of the current document:

```
       `Checking if the "Different on left and right pages" mode is already activated
    If(WR Get doc property(Area;wr different left right pages)#1)
          `If not, activating this mode
       WR SET DOC PROPERTY(Area;wr different left right pages;1)
       ALERT("Warning: the document is now in 'Different on left and right pages' mode!")
    End if
       `Setting the cursor in the left footer
    WR SET FRAME(Area;wr left footer)
       `Inserting 'Page X' in roman uppercase
    WR INSERT TEXT(Area;"Page ")
⇒   WR INSERT PAGE NUMBER(Area;4)
    WR INSERT TEXT(Area;" on ")
    WR INSERT EXPRESSION(Area;"WR Count(Area;11)")
```

See also

WR GET PAGE NUMBER FORMAT, WR SET FRAME.

WR GET PAGE NUMBER FORMAT

WR GET PAGE NUMBER FORMAT (area; format; numType)

Parameter	Type		Description
area	Longint	→	4D Write area
format	Integer	←	Type of format
numType	Integer	←	Type of page numbering 0 = Page number, 1 = Total number of pages

Description

The command WR GET PAGE NUMBER FORMAT allows you to determine the display format and the type of numbering used in an inserted page number reference. The reference should be already selected.

The format parameter returns the display format number of the reference, to indicate which option had been chosen in the “Insert page number...” dialog box:

Format type	Value
123	0
abc	1
ABC	2
Roman numerals (lower case)	3
Roman numerals (upper case)	4

The numType parameter returns 0 if the reference is the page number and 1 if the reference is the total number of pages.

See Also

WR INSERT PAGE NUMBER.

WR INSERT DATE AND TIME

version 6.5.4

WR INSERT DATE AND TIME (area; dateFormat; timeFormat)

Parameter	Type		Description
area	Longint	→	4D Write area
dateFormat	Integer	→	Number of the date format
timeFormat	Integer	→	Number of the time format

Description

The command WR INSERT DATE AND TIME allows you to insert at the cursor location a reference that displays the dynamic date and/or time. If there is a current text selection in your document, it will be replaced with the inserted reference.

The dateFormat parameter allows you to set a display format for the date reference. Here are the values you can use:

dateFormat	Name	Value
<no date>		0
6/01/00	(Short)	1
Thu 6 Jan 2000	(Abbreviated)	2
Thursday 6 January 2000	(Long)	3
01/06/2000	(MM DD YYYY)	4
January 6, 2000	(Month Day Year)	5
Jan 6, 2000	(Abbr Month Day)	6
06/01/2000	(MM DD YYYY forced)	7

The timeFormat parameter returns the time format number for the inserted reference.

timeFormat	Value
<no hour>	0
HH:MM:SS	1
HH:MM	2
HH hours MM minutes SS seconds	3
HH hours MM minutes	4
HH:MM AM PM	5

See Also

WR GET DATE AND TIME FORMAT.

WR GET DATE AND TIME FORMAT

version 6.5.4

WR GET DATE AND TIME FORMAT (area; dateFormat; timeFormat)

Parameter	Type		Description
area	Longint	→	4D Write area
dateFormat	Integer	←	Number of the date format
timeFormat	Integer	←	Number of the time format

Description

The command WR GET DATE AND TIME FORMAT allows you to determine the display format of a selected dynamic date and/or time.

The dateFormat parameter returns the date format number for the inserted reference.

dateFormat	Name	Value
<no date>		0
6/01/00	(Short)	1
Thu 6 Jan 2000	(Abbreviated)	2
Thursday 6 January 2000	(Long)	3
01/06/2000	(MM DD YYYY)	4
January 6, 2000	(Month Day Year)	5
Jan 6, 2000	(Abbr Month Day)	6
06/01/2000	(MM DD YYYY forced)	7

The timeFormat parameter returns the time format number for the inserted reference.

timeFormat	Value
<no hour>	0
HH:MM:SS	1
HH:MM	2
HH hours MM minutes SS seconds	3
HH hours MM minutes	4
HH:MM AM PM	5

See Also

WR INSERT DATE AND TIME.

WR INSERT HTML EXPRESSION

version 6.7

WR INSERT HTML EXPRESSION (area; htmlExpression)

Parameter	Type		Description
area	Longint	→	4D Write area
htmlExpression	Text	→	HTML expression

Description

The command WR INSERT HTML EXPRESSION inserts in area the HTML expression put into the htmlExpression parameter. The expression is inserted where the cursor is located. If text was selected at the moment of insertion, the text is replaced by the expression.

The HTML expression will not appear in the original 4D Write document but will be inserted as a HTML expression when the document is saved in HTML format. The HTML text will be interpreted directly through a Web browser; it can therefore contain any kind of HTML tag (URLs, style markers, images, etc.).

When the 4D Write document is exported in HTML, the expression will be saved in the generated HTML document.

See Also

WR Get HTML expression.

WR Get HTML expression

WR Get HTML expression (area) → Text

Parameter	Type		Description
area	Longint	→	4D Write area
Function result	Text	←	Content of the HTML expression

Description

The WR Get HTML expression command allows recuperating the text of the HTML expression currently selected within area.

To select HTML expressions contained in a 4D Write document, you should use the WR Count(Area;wr nb HTML expressions) statement and then make a loop for WR SELECT(Area;13;$loop).

Example

You want to get HTML expressions contained in your 4D Write document:

```
        C_LONGINT(Area;$i;$NbHTMLExp)
        C_TEXT($MyExp)

        $NbHTMLExp:=WR Count(Area;wr nb HTML expressions)
        For($i;1;$NbHTMLExp)
           WR SELECT(Area;13;$i)
⇒          $MyExp:=WR Get HTML expression(Area)
        End for
```

See Also

WR INSERT HTML EXPRESSION.

WR INSERT RTF EXPRESSION

version 6.7

WR INSERT RTF EXPRESSION (area; rtfExpression)

Parameter	Type		Description
area	Longint	→	4D Write area
rtfExpression	Text	→	RTF expression

Description

The command WR INSERT RTF EXPRESSION inserts in area the RTF expression put into the rtfExpression parameter. The expression is inserted where the cursor is located. If text was selected at the moment of insertion, the text is replaced by the expression.
When the 4D Write document is exported in RTF, the expression will be saved in the generated RTF document.

The RTF (*Rich Text Format*) is an exchange file format that saves most format attributes within a document (size, style and character color, margins, etc.) between different word processing softwares. This format is based on the use of specific markers interpreted at the time of RTF import.

See Also

WR Get RTF expression.

WR Get RTF expression

WR Get RTF expression (area) → Text

Parameter	Type		Description
area	Longint	→	4D Write area
Function result	Text	←	Content of the RTF expression

Description

The WR Get RTF expression command allows recuperating the text of the RTF expression currently selected within area.

To select RTF expressions contained in a 4D Write document, you should use the WR Count(Area;wr nb RTF expressions) command and then make a loop for WR SELECT(Area;14;$loop).

Example

You want to get RTF expressions contained in your 4D Write document:

```
    C_LONGINT(Area;$i;$NbRTFExp)
    C_TEXT($MyExp)

    $NbRTFExp:=WR Count(Area;wr nb RTF expressions)
    For($i;1;$NbRTFExp)
       WR SELECT(Area;14;$i)
⇒      $MyExp:=WR Get HTML expression(Area)
    End for
```

See Also

WR INSERT RTF EXPRESSION.

version 6.7

WR INSERT HYPERLINK (area; linkType; urlStyle; linkLabel; linkContent; methodRef)

Parameter	Type		Description
area	Longint	→	4D Write area
linkType	Integer	→	Hyperlink type: 0 = Method, 1 = URL, 2 = 4D Write Document
urlStyle	Integer	→	URL appearance: 1 = Default style, 0 = Custom style
linkLabel	Text	→	Link's visible text (View/Values mode)
linkContent	Text	→	Hyperlink value
methodRef	Longint	→	Value for $3, 3rd parameter of the method (if the link type is Method)

Description

The command WR INSERT HYPERLINK inserts a "hyperlink" reference within area, at the current cursor location or in place of the current text selection.

linkType

The linkType parameter defines the type of hypertext link to insert. 4D Write allows for three types of hypertext links: Method type links, URL type links, and Document type links.

• A **Method** type link executes a 4D method once the reference has been clicked. The method cannot be a function and it is not possible to pass parameters. However, it can receive two or three values in $1, $2, and, optionally $3:

- $1 (Longint) contains the 4D Write area reference,
- $2 (Text) contains the link label,
- $3 (Longint) contains an arbitrary numeric value that you can associate with a link using the methodRef parameter or in the 4D Write "User mode".

In light of the database compiling, it is necessary to declare $1 and $3 as Longints and $2 as Text even if you do not use them.
To insert a Method type link, put 0 in linkType.

• A **URL** type link opens the default browser and accesses a specific URL defined within the linkContent parameter. To insert a URL type link, put 1 in linkType.

• A **Document** type link replaces, once the link has been clicked, the current document by another document whose path was set in the linkContent parameter. Of course, the format of the document to be opened must be recognized by 4D Write. To insert a Document type link, put 2 in linkType.

urlStyle
The urlStyle parameter allows you to define the appearance of the inserted hypertext link:
• If you would like to keep the default hyperlink appearance (blue and underlined), put 1 in the urlStyle paramater. Default colors can be modified programmatically, using the WR SET DOC PROPERTY command.
• If you would like to use a customized appearance, put 0. In this case, you can select the link and define the style using the WR SET TEXT PROPERTY command.
If you put 0 and do not set any link style, the link will appear as current text (it will not be graphically materialised).

linkLabel
The linkLabel parameter sets the link's visible text (in View/Values mode).

linkContent
The linkContent parameter contains the hypertext link value. The nature of this value depends on the type of link:
• For a 4D Method type link, put the name of the method (for example "Order_Clients"),
• For an URL type link, put the complete URL (for example "http://www.4D.com/")
• For a Document type link, put the full path to the document (for example, "C:\MyFolder\MyDoc.4w7" under Windows, or "HardDrive:MyFolder:MyDoc" under MacOS).

methodRef
The methodRef parameter allows you, when the link is a 4D method type, to add a supplementary value to the called method. The method will receive this value in the $3 parameter (Longint type).

Examples

(1) In your 4D Write documents, you want to provide hypertext navigation based on document type links. The following method manages pathnames dynamically, whatever the platform:

```
	$Doc:=Structure file
	Doc:=$Doc
	While (Position(":";$Doc)#0)
		$Doc:=Substring($Doc;1+Position(":";$Doc);Length($Doc))
		$Long:=Length($Doc)
	End while
	Doc:=Substring(Doc;1;Length(Doc)-$Long)
	PLATFORM PROPERTIES($Platf;$Syst;$Computer)
	If ($Platf=Windows )
		$name:=Doc+"Documentation"+"/"+"01_Introduction.4W7"
	Else
		$name:=Doc+"Documentation"+":"+"01_Introduction.4W7"
	End if
	$title:="See Documentation"
⇒	WR INSERT HYPERLINK (Writearea;2;1;$title;$name)
```

(2) You want to insert the URL of your Web site in the 4D Write area:

```
⇒       WR INSERT HYPERLINK(area;1;"Visit that great site";"http:/www.MySite.com/")
```

(3) This example illustrates method type links. In your document, you want the user to be able to enter information, for example his/her name and first name in a particular place. You will insert a hyperlink calling a method named *Hyperlink_Method*. This method asks the user to enter either her/his name or first name, depending on the value passed in $3. The entered data will then replace the link:

```
           `Hyperlink_Method
        C_LONGINT($1;$3)
        C_TEXT($2)
        Case of
           : ($3=1)
              WR INSERT TEXT ($1;Request("Enter your first name"))
           : ($3=2)
              WR INSERT TEXT ($1;Request("Enter your last name"))
        End case
        WR GET SELECTION ($1;$deb;$end)
        WR SET SELECTION ($1;$deb;$end+1)
        WR EXECUTE COMMAND ($1;wr cmd clear)
```

Inserting the method type hyperlink in the 4D Write area:

```
        $title:="Click to enter"
        $method:="Hyperlink_Method"
        WR INSERT TEXT (Area;"Last name: ")
⇒       WR INSERT HYPERLINK (Area;0;1;$title;$method;1)
        WR INSERT TEXT (Area;Char(Carriage Return )+"First name: ")
⇒       WR INSERT HYPERLINK (Area;0;1;"Click to enter";"Hyperlink_Method";2)
```

See Also

WR GET HYPERLINK.

version 6.7

WR GET HYPERLINK (area; linkType; urlStyle; linkLabel; linkContent; methodRef)

Parameter	Type		Description
area	Longint	→	4D Write area
linkType	Integer	←	Hyperlink type: 0 = Method, 1 = URL, 2 = 4D Write Document
urlStyle	Integer	←	URL appearance: 1 = Default style, 0 = Custom style
linkLabel	Text	←	Link’s visible text (View/Values mode)
linkContent	Text	←	Hyperlink value
methodRef	Longint	←	Value for $3, 3rd parameter of the method (if the link type is Method)

Description

The WR GET HYPERLINK command returns the properties of the selected hyperlink within area.

linkType
- If the link is a 4D Method type, linkType returns 0.
- If the link is a URL type, linkType returns 1.
- If the link is a Document type, linkType returns 2.

urlStyle
- If the link style is set to the default, urlStyle returns 1.
- If the link style is customized, urlStyle returns 0. In this case, you can use the WR GET TEXT PROPERTY command for style information.

linkLabel
linkLabel returns the link’s visible text (in View/Values mode).

linkContent
linkContent returns the hypertext value, in other words:
- for a 4D Method type link, the name of the method,
- for a URL type link, the complete URL,
- for a Document type link, the complete document path.

methodRef
methodRef returns the value put in the called method (if the link is a 4D Method type).

To select hyperlinks contained in a 4D Write document, you should use the WR Count(Area;wr nb hyperlinks) command and then make a loop for WR SELECT(Area;12;$loop).

See Also

WR INSERT HYPERLINK.

6

WR Documents

Documents, Introduction

WR Documents

version 6.5 (Modified)

The 4D Write commands and functions of the "WR Documents" theme allow you to manipulate 4D Write documents that are saved to disk.

Using these commands, you can procedurally save, open or lock 4D Write documents.

Also, these commands allow you to set and get document information such as the subject or author.

WR GET DOCUMENT INFO

version 6.5

WR GET DOCUMENT INFO (area; string; subject; author; company; notes; creationDate; creationTime; modifDate; modifTime; lock)

Parameter	Type		Description
area	Longint	→	4D Write area
string	String	←	Title of the document
subject	String	←	Subject of the document
author	String	←	Author of the document
company	String	←	Company name
notes	String	←	Document notes
creationDate	Date	←	Creation date
creationTime	Time	←	Creation time
modifDate	Date	←	Last modification date
modifTime	Time	←	Last modification time
lock	Integer	←	0=unlocked 1=locked

Description

The WR GET DOCUMENT INFO command allows you to retrieve document information as displayed in the Document information dialog. The Document information dialog is displayed by selecting **Document information** from the **Tools** menu.

Some of this information such as the document subject, the author's name, the company name and the notes can be set using the WR SET DOCUMENT INFO command.

lock can be set using the WR LOCK DOCUMENT command. It is a logical lock that prevents the user from modifying the document. It affects user operations such as Paste, Cut, text entry, modify or replace attributes. The user can still browse the document, copy text, perform some character searches or print the document.

creationDate, creationTime, modifDate, modifTime are automatically updated by 4D Write when the document is saved.

Example

See the example for the WR SET DOCUMENT INFO command.

See also

WR SET DOCUMENT INFO.

WR LOCK DOCUMENT

WR LOCK DOCUMENT (area; status)

Parameter	Type		Description
area	Longint	→	4D Write area
status	Integer	→	0=unlocked 1=locked

Description

The WR LOCK DOCUMENT prevents users from modifying the 4D Write area referenced by area. Once the document is locked, users cannot paste text, cut text, enter or modify text. Scrolling, copying, searching and printing the document are still possible.

To determine the lock status of the current document, you can use the WR GET DOCUMENT INFO command. This information is also displayed in the Document information dialog. You can access that dialog by selecting **Document information** from the **Tools** menu.

• If status equals 1, the document will be locked
• If status equals 0, the document will be unlocked

Example

You want to close records definitively and prevent users from editing them.

```
	`It will not be possible to edit the document
⇒	WR LOCK DOCUMENT(Area;1)
	`Users will not be able to select the menu command Tools>Document Information
	`to open the dialog box and enable the option
	WR LOCK COMMAND(Area;wr cmd doc information;0)
```

See also

WR LOCK COMMAND.

WR OPEN DOCUMENT

WR OPEN DOCUMENT (area; document{; type})

Parameter	Type		Description
area	Longint	→	4D Write area
document	String	→	Path of document to open
		←	Path of the opened document
type	String	→	Type of the document to open (4 characters)
		←	Type of the open document (4 characters)

Description

The WR OPEN DOCUMENT command opens the document specified by document and places it in the 4D Write area referenced by area.

document is the name or the complete access path of the document file. On Windows, you must include the file extension in addition.
Examples:
• on Windows, you must include the "\" character between directories: "D:\directory1\directory2\file.4W7").
• on MacOS, you must include the ":" character between folders: "MacintoshHD:Folder:Document".

If document contains only the name of the file, WR OPEN DOCUMENT will look for the document in the folder of the database's structure file.

If document is an empty string, WR OPEN DOCUMENT displays the standard Open file dialog.
When the Open button of the Open file dialog is clicked the *OK* system variable is set to 1, and the document variable will be assigned the complete access path of the file the user selects.
In this case the type parameter returns the type selected by the user in the type drop-down list or the document type if no type was selected by the user.
If the user clicks the Cancel button, document returns an empty string and the *OK* system variable is set to 0.
The type optional parameter allows you to filter the document types displayed by default in the standard Open Document dialog box— except for HTML documents. For HTML documents, the type parameter is used for displaying either the HTML source code (if type "TEXT" is passed) or the HTML page (if type contains "HTML" or is omitted).

The file formats supported by this command are:

Type	Files
4WR7	4D Write
4WR6	4D Write version 6.0
4WT7	Template 4D Write
RTF	RTF file
DOC6	Word document
ASCW	Windows text file
ASCM	Mac OS text file
ASCU	Unicode text file
HTML	HTML text file

Example

The following example opens a file located in the database's directory.

```
⇒   WR OPEN DOCUMENT(area;"HD:Folder:database folder:File")   'On Mac OS
⇒   WR OPEN DOCUMENT(area;"D:\directory\Basedirectory\file.4W7")   'On Windows
```

See also

WR SAVE DOCUMENT.

WR SAVE DOCUMENT

version 6.7 (Modified)

WR SAVE DOCUMENT (area; document{; type})

Parameter	Type		Description
area	Longint	→	4D Write area
document	String	→	Pathname of the document to create
		←	Pathname of the created document
type	String	→	Type of document

Description

The WR SAVE DOCUMENT command saves the document located in the 4D Write area referenced by area, using the access path passed in document.

document is the name or the complete access path of the document file. On Windows, you must include the file extension, in order to determine the file type.
Examples:
• on Windows or for crossplatform compliance, you must include the "\" character between directories: "D:\directory1\directory2\file.4W7".
• on Mac OS, you must include the ":" character between folders: "MacintoshHD:Folder:Document".

If document contains only the name of the file, WR SAVE DOCUMENT will save the document in the folder of the database's structure file.

If document is an empty string, WR SAVE DOCUMENT displays the standard Save file dialog.
When the **Save** (Mac OS) or **OK** (Windows) button of the Save file dialog is clicked, the OK system variable is set to 1, and the document variable will be assigned the complete access path of the file the user selects.
In this case the type parameter returns the type selected by the user in the type drop-down list, or the document type if no type was selected by the user.
If the user clicks the **Cancel** button, document returns an empty string and the OK system variable is set to 0.

File formats can be selected from the Type drop-down list (on Windows) or from the type pop-up menu in the Save file dialog.

The available file formats are:

Type	Files	Windows extension
4WR7	4D Write	.4W7
4WT7	Template 4D Write	.4WT
RTF	RTF file	.RTF
ASCW	Windows text file	.TXT
ASCM	Mac OS text file	.TXT
ASCU	Unicode text file	.TXT
HTML	HTML text file	.HTML
DOC8	Word 97 Win/Word 98 MacOS	.DOC

The type parameter is used for the document encoding only. It corresponds neither to a MacOS file type, nor to a Windows extension.
However, the parameter is used by 4D Write to determine the appropriate value for the Windows file extension or the Mac OS file creator/type:

• ***Windows***

4D Write format	Extension
4D Write	.4W7
RTF	.RTF
HTML	.HTM
ASCII PC/Mac	.TXT
ASCII unicode	.TXT
Word	.DOC

The file extension is defined according to the type parameter value, even if the name already has an extension. For example, if you pass "Report.RTF" in the document parameter and "HTML" in type, the file will be named "Report.HTM".

• ***MacOS***

4D Write format	Creator	Type
4D Write	4DW7	4WR7
RTF	4DW7	RTF
HTML	MOSS	TEXT
ASCII PC/Mac	4DW7	TEXT
ASCII unicode	4DW7	TEXT
Word	MSWD	W8BN

Note: Exporting in Word format does not support pictures (of any type), bullets or hypertext links.

Examples

(1) You want to save the document named 'LetterClient' in the 4D Write file format. This document will be saved into the "WriteDocuments" folder located at the same level as the database's structure file:

```
	`Getting the full pathname to the database structure file
$Doc:=Structure file
Doc:=$Doc
$long:=0
	`Getting position of the last separator to remove structure name from full pathname
While((Position(":";$Doc)#0)
	$Doc:=Substring($Doc;1+Position(":";$Doc);Length($Doc))
	$Long:=Length($Doc)
End while
	`Concatenating names to build the full pathname of the document
	`Adding an extension to the document allows cross-platform document management
Doc:=Substring(Doc;1;Length(Doc)-$Long)+"WriteDocuments:LetterClient.4W7"
⇒	WR SAVE DOCUMENT(Area;doc;"4WR7")
```

(2) You want to give the user the ability to choose both the name and type of the document to save. Then, you want to retrieve the chosen values:

```
DocName:=""
DocType:=""
⇒	WR SAVE DOCUMENT (Area;DocName;DocType)
If (OK=1)
	...	`Using the DocName and DocType values
End if
```

See also

WR OPEN DOCUMENT.

WR SET DOCUMENT INFO

WR SET DOCUMENT INFO (area; title; subject; author; company; comment)

Parameter	Type		Description
area	Longint	→	4D Write area
title	String	→	Title of the document
subject	String	→	Document subject
author	String	→	Author of the document
company	String	→	Company name
comment	Text	→	Comment

Description

The WR SET DOCUMENT INFO stores in the document the information that is passed in the parameters. From a user standpoint, the information is displayed in the Document information dialog box. You can access that dialog by selecting **Document information** in the **Tools** menu.

To manage the document lock status, refer to WR LOCK DOCUMENT.

Example

You want users to be able to modify only the Title, Subject and Comment of the document information. You need to implement a method that intercepts the selection of menu commands and display your own customized form when users select **Document information** from the **Tools** menu.

1. In the form method of the form that contains the 4D Write area, place the following code to intercept the menu command:

```
Case of
   : (Form event=On Load)
      WR ON COMMAND(WArea;"z65OnCmd")
End case
```

2. The method 'z65OnCmd' is the following:

```
C_LONGINT($1;$2;$3)
Case of
   : ($2=wr cmd doc information)
         `=801, if the user selects Tools>Document Information...
      DIALOG([TheTable];"InfoArea")   `Custom Information form
   Else
      WR EXECUTE COMMAND($1;$2)   `If the user selects any other menu command
End case
```

3. In the customized Information form, named "InfoArea", only the variables *vTitle*, *vSubject* and *vComments* are editable. Here is the method attached to this form:

```
    Case of
        : (Form event=On Load)
            WR GET DOCUMENT INFO (WArea;vTitle;vSubject;vAuthor;vCy;vComments;
                                          DCreat;HCreat;DModif;HModif;Lock)
                `You assign the empty elements if necessary
            If (vCy="")
                vCy:="A.C.I."
                vAuthor:=Current user
                vCreation:=String(DCreat)+" at "+Time string(HCreat)
                vModification:=String(DModif)+" at "+Time string(HModif)
            End if
        : (Form event=On Unload)    `When the form is closed
⇒           WR SET DOCUMENT INFO(WArea;vTitle;vSubject;vAuthor;vCy;vComments)
    End case
```

See also

WR GET DOCUMENT INFO.

7

WR Picture Control

Picture Control, Introduction

WR Picture Control

version 6.5 (Modified)

The 4D Write commands of the "WR Picture Control" theme allow you to manage pictures in 4D Write areas. Using these commands, you can insert, position and delete any picture in your 4D Write areas.

WR DELETE PICTURE IN PAGE

version 6.5

WR DELETE PICTURE IN PAGE (area; pictureNumber)

Parameter	Type		Description
area	Longint	→	4D Write area
pictureNumber	Longint	→	Picture number

Description

The WR DELETE PICTURE IN PAGE command deletes the picture whose number is passed in pictureNumber from the 4D Write area referenced by area. For the WR DELETE PICTURE IN PAGE command to operate properly, the picture must be located in the page, rather than in the text stream. To delete a picture in the text stream, select it and call WR DELETE SELECTION.

You can retrieve a type number of pictures in an area by using, WR Count(area;13). When deleting a picture, 1 is substracted from each of the following picture numbers. You can also retrieve the picture number using the WR Get selected picture command.

Example

The following example deletes all the pictures located in the page for the specified area.

```
        $NbOccurrence:=WR Count(area;13)
        For ($i;1;$NbOccurrence)
                `It is always the first picture that is deleted
⇒           WR DELETE PICTURE IN PAGE (area;1)
        End for
```

See also

WR GET PICTURE IN PAGE INFO, WR INSERT PICTURE.

WR GET PICTURE IN PAGE INFO

version 6.5

WR GET PICTURE IN PAGE INFO (area; pictureNumber; page; behind; firstPage; horizPos; verticalPos; width; height; origWidth; origHeight)

Parameter	Type		Description
area	Longint	→	4D Write area
pictureNumber	Longint	→	Picture number
page	Longint	←	Picture location
behind	Integer	←	0=Picture is in front of the text, 1=Picture is behind the text
firstPage	Integer	←	1=Picture is not in the first page, 0=Picture appears in all the pages
horizPos	Number	←	Horizontal position in the page
verticalPos	Number	←	Vertical position in the page
width	Number	←	Current width of the picture
height	Number	←	Current height of the picture
origWidth	Number	←	Original width of the picture
origHeight	Number	←	Original height of the picture

Description

The WR GET PICTURE IN PAGE INFO returns information about the picture whose number was passed in pictureNumber, as it currently appears in the 4D Write area referenced by area.

Warning: this command should not be used with pictures that are part of the text flow.

• page allows you to know in which page the picture is displayed.
If page is greater than -1, the picture is displayed in the page whose number was returned. This value takes into account the page numbering as it is currently defined.
If page is equal to -1, the picture is displayed in all the pages.
If page is equal to -2, the picture is displayed in all the right pages.
If page is equal to -3, the picture is displayed in all the left pages.

• behind
If behind is equal to 0, the picture is in front of the text.
If behind is equal to 1, the picture is behind the text.

• firstPage
If firstPage is equal to 0, the picture is displayed on all pages.
If firstPage is equal to 1, the picture is displayed on all pages except the first page.

horizPos and vertPos return the coordinates of the picture's upper left corner in relation to the upper left corner of the page. Those values are expressed in the current default units for the document.

width and height return the current dimensions of the picture.
origWidth and origHeight return the original dimensions of the picture before any modification. If the picture was not resized, origWidth and origHeight return the same values as width and height. Those values are expressed in the current default units for the document.

Note: It may be convenient to change the current unit to pixels for some computations.

Example

See the example for the WR SET PICTURE IN PAGE INFO command.

See also

WR DELETE PICTURE IN PAGE, WR GET PICTURE SIZE, WR SET PICTURE IN PAGE INFO.

version 6.5

WR GET PICTURE SIZE (area; width; height; origWidth; origHeight)

Parameter	Type		Description
area	Longint	→	4D Write area
width	Number	←	Current width of the picture
height	Number	←	Current height of the picture
origWidth	Number	←	Width of the original picture
origHeight	Number	←	Height of the original picture

Description

The WR GET PICTURE SIZE command allows you to retrieve information about the size of a selected picture. That picture must be located in the text flow. To get size information about a picture embedded in a page, use the WR GET PICTURE IN PAGE INFO command. For the WR GET PICTURE SIZE command to operate properly, the picture has to be the only element of the selection.

height is the picture height. It is expressed in the current default units for the document.

width is the picture width. It is expressed in the current default units for the document.

origHeight and origWidth are respectively the original height and width before the picture was resized. If origHeight and origWidth are identical to height and width the picture has not been resized. origHeight and origWidth are expressed in the current document unit.

Note: If you want to select a picture, you can use the WR SELECT command.

Examples

See the examples for the WR INSERT PICTURE and WR GET CURSOR POSITION commands.

See also

WR GET PICTURE IN PAGE INFO, WR SET PICTURE SIZE.

WR Get selected picture

version 6.5

WR Get selected picture (area; status) → Picture

Parameter	Type		Description
area	Longint	→	4D Write area
status	Integer	←	Picture status
Function result	Picture	←	Selected picture

Description

The WR Get selected picture command returns a copy of the picture currently selected in the 4D Write area referenced by area.

The status parameter can return any of the following values:
• If status = -1, no picture is selected.
• If status = 0, the selected picture is in the text flow.
• If status > 0, the selected picture is in the page.

status can help you identify the picture when using WR GET PICTURE IN PAGE INFO, WR SET PICTURE IN PAGE INFO or WR DELETE PICTURE IN PAGE.

Example

See the example for the WR SET PICTURE IN PAGE INFO command.

WR INSERT PICTURE

version 6.5

WR INSERT PICTURE (area; picture; destination; horizPos; verticalPos; behind; firstPage)

Parameter	Type		Description
area	Longint	→	4D Write area
picture	Picture	→	Picture to insert
destination	Longint	→	Location of the insertion
horizPos	Number	→	Horizontal position in the page
verticalPos	Number	→	Vertical position in the page
behind	Integer	→	0=picture above the text 1=picture in background
firstPage	Integer	→	1=Picture is not in first page, 0=picture is displayed on all pages

Description

The WR INSERT PICTURE command inserts a picture in the 4D Write area referenced by area at the location specified by destination, horizPos and verticalPos.

picture can either be a picture field or a picture variable. If the parameter content is not a picture, error number 1065 is returned.

The destination optional parameter allows you to define where the picture will be inserted.

• If you want the picture to be inserted into the text flow, pass 0 in destination or omit the parameter. In this case the other parameters will not be used and the picture will either be inserted at the location of the insertion point or will replace the current selection.

• If you want to insert the picture into the page, use one of the following options:
- If destination is greater than 0, the picture will be displayed in the page whose number is destination. The value of destination must take into account the page numbering as it is defined in the Preferences dialog.
- If destination equals -1, the picture will be under the text and it will be displayed in all the pages.
- If destination equals -2, the picture will be under the text and it will be displayed in all the right pages.
- If destination equals -3, the picture will be under the text and it will be displayed in all the left pages.
- If destination equals -4, the picture will be under the text and it will be visible in the page containing the insertion point.

The horizPos and verticalPos optional parameters are expressed in the current default unit for the document. These two parameters set the coordinates of the picture's upper left corner in relation to the upper left corner of the page.

The behind optional parameter allows you to define whether the picture will be behind or in front of the text.
• If behind equals 1, the picture will be behind the text. In this case it is necessary to pay attention to the text and paragraph background attributes. Selecting "None" will allow you to see the picture behind the text.
• If behind equals 0, the picture will be in front of the text.

The firstPage optional parameter applies only if destination equals -1, -2 or -3.
• If firstPage equals 1 , the picture will not be displayed in the first page.
• If firstPage equals 0 , the picture will also be displayed in the first page.

Examples

(1) The following example is an object method attached to a button. It allows you to insert a 4D picture in the 4D Write area and to downsize it by 50%.

```
⇒      WR INSERT PICTURE(Area;Logo)    `Inserting a picture from the Logo field
       WR SELECT(Area;4;1)    `Selecting the picture
       WR GET PICTURE SIZE(Area;Vert;Horiz;pictPosition)    `Getting the picture size
       WR SET PICTURE SIZE(Area;Vert*1/2;Horiz*1/2)    `Resizing the picture
```

(2) For an example of picture insertion in the page, refer to the WR SET PICTURE IN PAGE INFO command.

See also

WR DELETE PICTURE IN PAGE.

WR SELECT PICTURE IN PAGE

WR SELECT PICTURE IN PAGE (area; pictureNum)

Parameter	Type		Description
area	Longint	→	4D Write area
pictureNum	Longint	→	Picture number

Description

The WR SELECT PICTURE IN PAGE command allows you to select the picture whose number is passed in pictureNum. For the command to operate properly, the picture must be located in the page (not in the text flow). If you want to select a picture located in the text flow, you can use WR SELECT(Area;4;XthPosition). Refer to the documentation for the WR SELECT command.

Example

See the example for the WR SET PICTURE IN PAGE INFO command.

See also

WR GET PICTURE IN PAGE INFO, WR INSERT PICTURE, WR SELECT.

WR SET PICTURE IN PAGE INFO

version 6.5

WR SET PICTURE IN PAGE INFO (area; pictureNumber; page; behind; firstPage; horizontPos; verticalPos; width; height)

Parameter	Type		Description
area	Longint	→	4D Write area
pictureNumber	Longint	→	Picture number
page	Longint	→	Location of the picture
behind	Integer	→	0=picture is under the text 1=picture is over the text
firstPage	Integer	→	1=image is not present in first page otherwise 0
horizontPos	Number	→	Horizontal position in page
verticalPos	Number	→	Vertical position in page
width	Number	→	Current picture width
height	Number	→	Current picture height

Description

The WR SET PICTURE IN PAGE INFO command allows you to modify the properties of the picture whose number was passed in pictureNumber.

Warning : this command is not to be used for pictures that are inserted in the text flow.

page allows you to define what page the picture is to be displayed in. To do so, pass the page number in page. This number should take into account the page numbering as it is set in the Preferences dialog box.
• If page equals -1, the picture will be displayed in all pages.
• If page equals -2, the picture will be displayed in all right pages.
• If page equals -3, the picture will be displayed in all left pages.
• If page equals -4, the previous value is not modified.

• behind
If behind equals 0, the picture will appear above the text.
If behind equals 1, the picture will appear behind the text. The text will then have a transparent background unless a background color was previously selected for it.

• firstPage
If firstPage equals 0, the picture will be displayed on all pages.
If firstPage equals 1, the picture will be displayed on all pages except the first page.

horizontPos and verticalPos allow you to set the horizontal and vertical coordinates of the upper left corner of the picture in relation to the upper left corner of the physical page. The value for horizontPos can be between 0 and the total page width. In this case, the printer margins will not be taken into account and the picture may end up located outside the printable area of the page.

Note: When pasting a picture in the User environment, the printer margins are taken into account.

width and height allow you to set the new dimensions of the picture. Values are expressed in the current default units for the document.

Note: Passing -1 in the following parameters will not modify their initial value: behind, firstPage, horizontPos, verticalPos, width and height.

Example

You want to insert the same picture in the header of each of your documents:

```
        C_REAL($PosHoriz;$PosVert;$PictWidth;$PictHeight;$OrigWidth;$OrigHeight;
                                                        $TxtMgTop;$HeadMgBottom)
        WR SET DOC PROPERTY(Area;wr view mode;0)
        $PosHoriz:=WR Get doc property (Area;wr text left margin)
        $PosVert:=WR Get doc property(Area;wr header top margin)
        ALL RECORDS([Interface])
           `Inserting the picture
⇒       WR INSERT PICTURE(Area;[Interface]Logo;-1;$PosHoriz;$PosVert;1;0)
           `Picture is stored in the Logo field
        WR SELECT PICTURE IN PAGE(Area;1)   `Selecting the picture
           `Getting picture properties
        MyPict:=WR Get selected picture(Area;$NumPict)
⇒       WR SET PICTURE IN PAGE INFO (Area;$NumPict;$Page;$Behind;$PageOne;
                      $PosHoriz;$PosVert;$PictWidth;$PictHeight;$OrigWidth;$OrigHeight)
           `Decreasing picture size of 50 %
        $PictHeight:=$PictHeight*1/2
        $PictWidth:=$PictWidth*1/2
⇒       WR SET PICTURE IN PAGE INFO(Area;$NumPict;$Page;$Behind;$PageOne;$PosHoriz;
                                                      $PosVert;$PictWidth;$PictHeight)
           `Checking that the header "covers" the logo
        $TxtMgTop:=WR Get doc property(Area;wr text top margin)
        $HeadMgBottom:=WR Get doc property(Area;wr header bottom margin)
        WR SET DOC PROPERTY(Area;wr text top margin;$PosVert+$PictHeight+
                                                        $TxtMgTop+$HeadMgBottom)
        WR SET DOC PROPERTY(Area;wr header bottom margin;$PosVert+$PictHeight)
```

See also

WR GET PICTURE IN PAGE INFO.

WR SET PICTURE SIZE

WR SET PICTURE SIZE (area; width; height)

Parameter	Type		Description
area	Longint	→	4D Write area
width	Number	→	New picture width
height	Number	→	New picture height

Description

The WR SET PICTURE SIZE command allows you to modify the size of the selected picture in the 4D Write area referenced by area.

This command has no effect on background pictures. To resize background pictures, use the WR SET PICTURE IN PAGE INFO command.

width and height are expressed in the current default units for the document. The values given must be within the page or within the column, when using multiple columns.

To use pixels as a unit, you can temporarily change the current default unit for the document and set it back after calling WR SET PICTURE SIZE.

Example

See the example for the WR INSERT PICTURE command.

See also

WR GET PICTURE SIZE.

8

WR Printing

Printing, Introduction

WR Printing

version 6.5 (Modified)

The 4D Write commands and functions of the "WR Printing" theme allow you to control the printing of a 4D Write area.

These commands are useful when you want to print a report or a form letter without having the user choose **Print** from the **File** menu.

WR PRINT

WR PRINT (area; mode; nbCopies)

Parameter	Type		Description
area	Longint	→	4D Write area
mode	Integer	→	0=Values 1=References
nbCopies	Integer	→	Number of copies to be printed

Description

The WR PRINT command prints the document contained in area. This command is the procedural equivalent of choosing **Print...** from the **File** menu without the display of the printing dialog boxes.

WR PRINT prints area once. Use WR PRINT MERGE if you want to print area once for each record in a selection.

If mode equals 1, referenced elements appear between left and right double angle brackets (« ») in your 4D Write area. If mode equals 0, the values of the referenced elements will be printed in the 4D Write area.
WR PRINT does not compute references. If you want the references to be updated before printing, execute the statement WR EXECUTE COMMAND (area;wr cmd compute references) before WR PRINT.

The nbCopies parameter controls the number of copies to be printed.

Example

The following example is the method for a button used on the form that contains area. If you click on this button, area will be printed. The document contains references that have to be updated before printing:

```
	WR EXECUTE COMMAND (area;wr cmd compute references)
⇒	WR PRINT (area; 0;1)
```

See Also

WR PRINT MERGE.

WR PRINT MERGE

WR PRINT MERGE (area; table; display)

Parameter	Type		Description
area	Longint	→	4D Write area
table	Integer	→	File number
display	Integer	→	Display/suppress the print settings dialog box

Description

The WR PRINT MERGE command prints the document contained in area once for each record in the selection of table. table is the number of the merging table. If table equals 0, WR PRINT MERGE displays the standard Print Mailing dialog box, allowing you to specify the table and change the selection of records for that table.
If the document contains references, they will be automatically processed before printing.

If display equals 0, the Print Settings dialog box does not appear. If display equals 1, the Print Settings dialog box appears.

Example

The following example prints a letter for each record in the [Clients] table. The letter is stored in a record of the [Letters] table.

```
   ALL RECORDS (Clients])   `Selecting all clients
   QUERY ([Letters];[Letters]Ref="Expedite")   `Looking for Expedite template
   Temp:=WR New offscreen area    `Creating an offscreen area
   WR PICTURE TO AREA(Temp;[Letters]Doc_)   `Placing template in offscreen area
⇒  WR PRINT MERGE (Temp;3)   `Merging the template with the selection in table 3
   WR DELETE OFFSCREEN AREA (Temp)   `Deleting the offscreen area
```

See Also

WR PRINT.

9

WR Tabs

Tabs, Introduction

version 6.5 (Modified)

The commands of the "WR Tabs" theme allow you to control the position and the properties of a tab stop located in a 4D Write area.

You can read or set tab stop properties as well as delete existing tabs, or create new ones.

WR ADD TAB

version 6.5

WR ADD TAB (area; position; justification; fillCharacter)

Parameter	Type		Description
area	Longint	→	4D Write area
position	Number	→	Tab location
justification	Integer	→	Justification value
fillCharacter	Alpha	→	Selected fill character

Description

The WR ADD TAB allows you to add a new tab at the location passed in position, measured from the left margin of the document. It also allows you to to define the fill character and the justification of the new tab stop.

This tab stop will be added to all the paragraphs of the selection. If a tab stop already exist at this location, it will be replaced by the one you just created.

position is the distance from the left margin (expressed in the document's default unit).

The justification optional parameter determines the tab stop type. You can either use the value or one of the following predefined constants.

Value	Constant	Text alignment
1	wr left tab	Left aligned
2	wr centered tab	Centered
3	wr right tab	Right aligned
4	wr decimal tab	Decimal
5	wr vertical separator tab	Vertical separator

If justification is omitted, a left aligned tab is created.

Note: The list of constants is available in Appendix D: 4D Write Constants.

The fillCharacter optional parameter can be any character whose ASCII code is between 33 and 127. This character will be added using the same font as the tab stop.
If fillCharacter is omitted or if you pass an empty string, no fill character will be inserted.

Example

The following example create a left tab stop, 50 units away from the left margin with a dot as fill character.

⇒ ***WR ADD TAB*** (area;50;1;".")
`or

⇒ ***WR ADD TAB*** (area;50;<u>wr left tab</u>;".")

See also

WR ADD STYLESHEET TAB, WR DELETE TAB.

WR DELETE TAB

WR DELETE TAB (area; tabNum)

Parameter	**Type**		**Description**
area	Longint	→	4D Write area
tabNum	Longint	→	Tabulation number

Description

The WR DELETE TAB command eletes the tab whose number (counting left-to-right) is passed in tabNum from the 4D Write area referenced by area. If other tabs are located at the same position, they too will be deleted.

Note: If the selection consists of several paragraphs, the tab numbering applies to the last selected paragraph.

Example

You want to remove all the tab stops from your document:

```
C_LONGINT(Area;$i;$TabNum;$uniform)
    `Inserting the cursor at the beginning of the area
WR SET SELECTION(Area;0;0)
    `Counting the number of paragraphs in the document
NbParag:=WR Count(Area;wr nb paragraphs)
    `Processing each paragraph
For ($i;1;NbParag)
        `Getting the position of the paragraph
    WR GET PARAGRAPHS(Area;Start;Pos)
        `Going inside the paragraph
    WR SET SELECTION(Area;Start+1;Debut+1)
        `Getting the number of tab stops
    $TabNum:=WR Get text property(Area;wr tab;$uniforme)
    While ($TabNum#0)
⇒       WR DELETE TAB(Area;1)
        $TabNum:=$TabNum-1
    End while
        `Repositioning just after the last processed paragraph
    WR GET SELECTION(Area;Pos;Pos)
End for
```

See also

WR ADD TAB, WR DELETE STYLESHEET TAB.

WR GET TAB

WR GET TAB (area; tabNumber; position; alignment; fillCharacter)

Parameter	Type		Description
area	Longint	→	4D Write area
tabNumber	Longint	→	Tab number
position	Number	←	Tab position
alignment	Integer	←	Justification value for the tab
fillCharacter	String	←	Fill character

Description

The WR GET TAB command returns the position, the alignment and the fill character for the tab whose number was passed in tabNumber and in the current ruler of area. The current ruler is the ruler in which the insertion point appears, or the last ruler when several paragraphs are selected.

• tabNumber: To know the number of tabs in the paragraph, you can use WR Get text property(area;45;Uniform), which will return the number of tab stops. You can then loop through the tab numbers to retrieve all the parameters of the current ruler.

• position: position is the distance from the left document margin to the tab stop, expressed in the current default units of the document.

• alignment: alignment is the alignment type of the tab.

Value	Text alignment
1	Left alignment
2	Centered
3	Right alignment
4	Decimal
5	Vertical separator

• fillCharacter can be any character whose ASCII code is contained between 33 and 127. If fillCharacter is an empty string, then there is no fill character in the tab stop setting.

Examples

See the examples for the WR SET TAB and WR DELETE TAB commands.

See also

WR GET STYLESHEET TAB, WR SET TAB.

WR SET TAB

version 6.5

WR SET TAB (area; tabNumber; position; alignment; fillCharacter)

Parameter	Type		Description
area	Longint	→	4D Write area
tabNumber	Longint	→	Tabulation number
position	Number	→	New tabulation position
alignment	Integer	→	New value for the tabulation justification
fillCharacter	String	→	New character selected as fill character

Description

The WR SET TAB command allows you to set the parameters of the tab stop whose number was passed in tabNumber (tabs are counted left to right). The WR SET TAB command will move the tab stop to position and will set the fill character as well as the alignment of the tab stop.

The selected tab stop will be modified for all the paragraphs of the current selection. If a tab stop already exists at the new location it will be replaced by the tab stop you just modified.

position is the distance from the left margin. position is expressed in the current default unit for the document. If you do not want to change the position of the tab stop, pass -1 in the parameter.

alignment specifies the alignment for the tab stop. You can either use the value or the constant.

Value	Constant	Text alignment
-1	-	No change
1	wr left tab	Left alignment
2	wr centered tab	Centered
3	wr right tab	Right alignment
4	wr decimal tab	Decimal
5	wr vertical separator tab	Vertical separator

Note: The list of text properties and their references are available in the Appendix D: 4D Write Constants, in the "WR Tabs" theme.

fillCharacter can be any character whose ASCII code is contained between 33 and 127. This character is displayed in the same font as the modified tab stop.

Example

In the selection, you want to delete the tab stops that are located at 168 points, move tab stops from 252 points to 280 points and assign '$' as fill character:

```
        C_LONGINT(Area;$i;$Nbtab;$Unit;$uniform;$Justif)
        C_REAL($Pos)
        C_STRING(2;$fill)
        $Nbtab:=WR Get text property(Area;wr tab;$uniform)
            `Storing current unit
        $Unit:=WR Get doc property(Area;wr unit)
        If ($Unit#2)
                `Setting unit to points if not already set
            WR SET DOC PROPERTY(Area;wr unit;2)
        End if
        $i:=1
        Repeat
            WR GET TAB(Area;$i;$pos;$Justif;$fill)
            Case of
                : ($Pos=168)
                        `Deleting tab stops located at 168 points
                    WR DELETE TAB(Area;$i)
                    $Nbtab:=$Nbtab-1
                : ($Pos=252)
                        `Moving tab stops located at 252 points to 280 points
⇒                   WR SET TAB(Area;$i;350;$Justif;"$")
                    $i:=$i+1
            End case
        Until ($i=$Nbtab)
            `Going back to original unit
        WR SET DOC PROPERTY (Area;wr unit;$Unit)
```

See also

WR GET TAB, WR SET STYLESHEET TAB.

10

WR Style Sheet

Style Sheet, Introduction

WR Style Sheet

version 6.5 (Modified)

The commands and functions of the "WR Style Sheet" theme allow you to have control over the style sheet used for the text selection.

You can retrieve the current style sheet or apply a different one. This capability enables you to control formatting features like bold, italics, and font size.

You can also delete any existing style sheet.

WR ADD STYLESHEET TAB

WR ADD STYLESHEET TAB (area; styleSheetNumber; location{; justification{; fillCharacter}})

Parameter	Type		Description
area	Longint	→	4D Write area
styleSheetNumber	Longint	→	Stylesheet number
location	Number	→	Tab location
justification	Integer	→	Justification value for the tabulation
fillCharacter	String	→	Selected fill character

Description

The WR ADD STYLESHEET TAB command allows you to add a new tab stop to the list of tab stops that the parameter styleSheetNumber refers to. Using the WR ADD STYLESHEET TAB command, you can set the tab postion, its type and its fill character.

If there is already tab stop at position, it will be replaced by the tab stop you just defined.

Note: Text that uses the style sheet you want to modify will not be updated unless you call the WR UPDATE STYLESHEET command to update text that uses that style sheet.

position is the distance from the left margin (expressed in the document's default units).

The justification optional parameter determines the tab stop type. You can either use the value or one of the following predefined constants.

Value	Constant	Text alignment
1	wr left tab	Left aligned
2	wr centered tab	Centered
3	wr right tab	Right aligned
4	wr decimal tab	Decimal
5	wr vertical separator tab	Vertical separator

If justification is omitted, a left aligned tab is created.

Note: The list of constants is available in Appendix D: 4D Write Constants.

The fillCharacter optional parameter can be any character whose ASCII code is between 33 and 127. This character will be added using the same font as the tab stop. If fillCharacter is omitted or if you pass an empty string, no fill character will be inserted.

Example

See the example for the WR UPDATE STYLESHEET command.

See also

WR ADD TAB, WR DELETE STYLESHEET TAB, WR GET STYLESHEET TAB, WR SET STYLESHEET TAB.

WR APPLY STYLESHEET

WR APPLY STYLESHEET (area; styleSheetNumber)

Parameter	Type		Description
area	Longint	→	4D Write area
styleSheetNumber	Longint	→	Stylesheet number

Description

The WR APPLY STYLESHEET command applies to the current selection in the 4D Write area designated by area the style sheet whose number is passed in styleSheetNumber. The formats of the style sheet will then be applied to the selection and the selection will appear as using that style sheet (when the cursor is in the text, the style sheet will be displayed in the style sheet drop-down list from the Style toolbar).

If styleSheetNumber does not correspond to any style sheet, the error 1078 (unknown style sheet) is be returned.

Example

See the example for the WR Create stylesheet command.

See also

WR Create stylesheet, WR UPDATE STYLESHEET.

WR Create stylesheet (area; name{; applyTo{; shortCut}}) → Longint

Parameter	Type		Description
area	Longint	→	4D Write area
name	String	→	Stylesheet name
applyTo	Longint	→	0=characters 1=paragraphs
shortCut	String	→	One character
Function result	Longint	←	Stylesheet reference number

Description

The WR Create stylesheet creates a new style sheet and returns the number that was assigned to it. The features of the new style sheet are set by the parameters name, applyTo and shortCut. You can modify the style sheet by using the WR SET STYLESHEET TEXT PROP, WR SET STYLESHEET FONT, WR SET STYLESHEET TAB and the style sheet reference number.

• name: the length of a style sheet name is limited to 31 characters.

• applyTo optional parameter
- If applyTo equals 0, the style sheet will be a character stylesheet.
- If applyTo equals 1, the style sheet will be a paragraph stylesheet.
- If applyTo is omitted, the style sheet will be a character style sheet.

• The shortCut optional parameter allows you to assign a keyboard shortcut to the style sheet. It only accepts one character. To use the shortcut you will need to press the key passed in this parameter with the Ctrl key (on Windows) or the Command key (on MacOS). It is recommended that you use a number in order to avoid any conflict with the standard 4D Write keyboard shortcuts.

If shortCut is omitted or if it is an empty character string no shortcut will be assigned to the style sheet.

Example

You want to add to each document your own customized character style sheet and to apply it to the selection. The style sheet is assigned the shorctuts **Command+1** on Mac OS and **Ctrl+1** on Windows. The font used is Comic Sans MS 12 points.

```
⇒    $NumSheet:=WR Create stylesheet (Area;"MyOwnStyle";0;"1")
     WR SET STYLESHEET FONT (Area;$NumSheet;"Comic Sans MS")
     WR SET STYLESHEET TEXT PROP (Area;$NumSheet;wr font size;12;1)
     WR EXECUTE COMMAND(Area;wr cmd select all)
     WR APPLY STYLESHEET(Area;$NumSheet)
```

See also

WR APPLY STYLESHEET, WR DELETE STYLESHEET, WR UPDATE STYLESHEET.

WR DELETE STYLESHEET

version 6.5

WR DELETE STYLESHEET (area; stylesheetNum)

Parameter	Type		Description
area	Longint	→	4D Write area
stylesheetNum	Longint	→	Stylesheet number

Description

The WR DELETE STYLESHEET command deletes the style sheet whose number was passed in styleSheetNum from the 4D Write area referenced by area. .

Warning: System style sheets cannot be deleted. You can use the WR GET STYLESHEET INFO command to determine if the style sheet is protected from deletion.

Example

You want to delete each unprotected style sheets in your document:

```
	C_LONGINT(Area)
	C_INTEGER(NbStyleSheet;$SheetNum)
	   `Counting number of style sheets
	NbStyleSheet:=WR Count(Area;wr nb stylesheets)
	$SheetNum:=1
	For ($i;1;NbStyleSheet)
	   WR GET STYLESHEET INFO(Area;$SheetNum;$Name;$ApplyTo;$Protected;
	                                                            $Shortcut)
	   If ($Protected=0)   `If the style sheet is not protected...
⇒	      WR DELETE STYLESHEET (Area;$SheetNum)
	   Else
	      $SheetNum:=$SheetNum+1
	   End if
	End for
```

See also

WR CREATE STYLESHEET.

WR DELETE STYLESHEET TAB

version 6.5

WR DELETE STYLESHEET TAB (area; stylesheetNumber; tabNumber)

Parameter	Type		Description
area	Longint	→	4D Write area
stylesheetNumber	Longint	→	Stylesheet number
tabNumber	Longint	→	Number of the tabulation to delete

Description

The WR DELETE STYLESHEET TAB command deletes the tab stop whose number was passed in tabNumber from the styleSheetNumber style sheet, in the 4D Write area referenced by area. Style sheets are numbered from top to bottom, as listed in the style sheet dialog box. This command has no effect on the selected text, even if it currently uses the styleSheetNumber style sheet.

To update the text that uses the modified style sheet, you need to use the WR UPDATE STYLESHEET command.

Example

See the example for the WR UPDATE STYLESHEET command.

See also

WR ADD STYLESHEET TAB, WR DELETE TAB.

WR Get stylesheet font

WR Get stylesheet font (area; stylesheetNumber) → String

Parameter	Type		Description
area	Longint	→	4D Write area
stylesheetNumber	Longint	→	Stylesheet number
Function result	String	←	Name of the font, or "" if no font is defined

Description

The WR Get stylesheet font command returns the name of the font that was assigned to the style sheet whose number was passed in styleSheetNumber in the 4D Write area referenced by area. Style sheet are numbered from top to bottom as shown in the style sheet dialog. If no font is defined for that style sheet, an empty string is returned.

Example

You want to remove the "Font" attribute from each style sheet where it is used, whenever the specified font is not installed in the system:

```
    ARRAY STRING(80;FontsArray)
    WR FONTS TO ARRAY(FontsArray)
    $StyleSheetNum:=WR Count(Area;wr nb stylesheets)
    For ($i;1;$StyleSheetNum)
⇒       $Fonts:=WR Get stylesheet font(Area;$i)
        If (($Fonts#"") & (Find in array(Area;$Fonts)=0))
            WR SET STYLESHEET FONT(Area;$i;"")
        End if
    End for
```

See also

WR Get font, WR SET STYLESHEET FONT.

WR GET STYLESHEET INFO

WR GET STYLESHEET INFO (area; stylesheetNumber; name; applyTo; protected; shortcut)

Parameter	Type		Description
area	Longint	→	4D Write area
stylesheetNumber	Longint	→	Stylesheet number
name	String	←	Stylesheet name
applyTo	Integer	←	0=characters, 1=paragraphs
protected	Integer	←	0= non protected, 1= protected
shortcut	String	←	One character or "" if no shortcut

Description

The WR GET STYLESHEET INFO allows you to retrieve information about the style sheet whose number is passed in styleSheetNumber and which is contained in the 4D Write area referenced by area.

• name returns the title of the style sheet.

• applyTo
If applyTo is equal to 0, the style sheet will only apply to characters.
If applyTo is equal to 1, the style sheet will only apply to paragraphs.

• protected
If protected is equal to 0, the style sheet is not protected, thus it is not a system style sheet.
If protected is equal to 1, the style sheet is protected, it is therefore a system style sheet and it cannot be deleted.

shortcut returns the shortcut assigned to the style sheet, if any. It consists of only one character. When using that shortcut you will need to hold down the Ctrl key (on Windows) or the Command key (on MacOS) while pressing the shortcut key.

If shortcut is an empty string, no shortcut is assigned to styleSheetNumber.

Examples

See examples for the WR SET STYLESHEET INFO, WR DELETE STYLESHEET and WR UPDATE STYLESHEET commands.

See also

WR SET STYLESHEET INFO.

WR GET STYLESHEET TAB

WR GET STYLESHEET TAB (area; stylesheetNum; tabNumber; position; justification; fillCharacter)

Parameter	Type		Description
area	Longint	→	4D Write area
stylesheetNum	Longint	→	Stylesheet number
tabNumber	Longint	→	Tab number
position	Number	←	Position of the tab
justification	Integer	←	Alignment value for the tab
fillCharacter	String	←	Selected fill character

Description

The WR GET STYLESHEET TAB command allows you to retrieve the settings of the tab stop whose number was passed in tabNumber and which belongs to the style sheet whose number was passed in styleSheetNumber in the 4D Write area referenced by area.

To know the number of tabs in the style sheet, you can use: WR GET STYLESHEET INFO(area;styleSheetNumber;wr tab;applyTO), which will return the number of tab stops.

position is the distance from the left document margin to the tab stop, expressed in the current default units of the document.

alignment is the alignment type of the tab:

Value	Text alignment
1	Left alignment
2	Centered
3	Right alignment
4	Decimal
5	Vertical separator

fillCharacter can be any character whose ASCII code is between 33 and 127. If fillCharacter is an empty string, then there is no fill character in the tab stop setting.

Example

You want to change the fill characters for each style sheet tab stop, and then update your document.

```
    $StyleSheetNum:=WR Count(Area;wr nb stylesheets)
    For ($i;1;$StyleSheetNum)
        $TabNum:=WR Get stylesheet text prop(Area;$i;wr tab;$Apply)
        If ($TabNum#0)
            For ($j;1;$TabNum)
⇒               WR GET STYLESHEET TAB(Area;$i;$j;$Pos;$Justif;$FillChar)
                If ($FillChar#"")
                    WR SET STYLESHEET TAB(Area;$i;$j;$Pos;$Justif;Char(126))
                End if
            End for
            WR UPDATE STYLESHEET(Area;$i)
        End if
    End for
```

See also

WR ADD STYLESHEET TAB, WR GET TAB, WR SET STYLESHEET TAB.

WR Get stylesheet text prop

WR Style Sheet

version 6.5

WR Get stylesheet text prop (area; stylesheetNumber; property; applyTo) → Real

Parameter	Type		Description
area	Longint	→	4D Write area
stylesheetNumber	Longint	→	Stylesheet number
property	Integer	→	Number of the text property to read
applyTo	Integer	←	0=the property is not applied 1=the property is applied
Function result	Real	←	Depends on the property parameter

Description

The WR Get stylesheet text prop command allows you to know, for area, whether the property passed in property is applied to the selection.

• property

If property = 7 (wr font number Constant), the returned value is an internal number. 4D Write sequentially assigns font numbers to fonts as they are used. This number can only be used by the WR SET STYLESHEET TEXT PROP command. It is recommended that you should use the WR Get stylesheet font and WR SET STYLESHEET FONT whose operation is based on font names.

The property 15 (wr stylesheet number Constant) has not meaning for this function.

If property = 64 (wr tab Constant), WR Get stylesheet text prop returns the number of tab stops set for the style sheet.

For color properties, the returned value will respect the following format (as in 4D and in the former version of 4D Write): 0x00RRVVBB. To separate the RGB values, use the WR COLOR TO RGB command.

If -1 is returned for the properties 11 (wr strikethrough color Constant), 12 (wr underline color Constant), or 13 (wr shadow color Constant), these elements are in the same color as the text.

If -1 is returned for the property 10 (wr text back color Constant), there is no background color selected for the text.

Note: property can be set using constants.

The list of the text properties constants are available in Appendix D: 4D Write Constants.

• If applyTo is equal to 1, the style sheet takes into account the property.
• If applyTo is equal to 0, the style sheet does not take into account the property.

Examples

See the examples for the WR UPDATE STYLESHEET, WR GET STYLESHEET TAB commands.

See also

WR SET STYLESHEET TEXT PROP.

WR SET STYLESHEET FONT

WR SET STYLESHEET FONT (area; stylesheetNumber; font)

Parameter	Type		Description
area	Longint	→	4D Write area
stylesheetNumber	Longint	→	Stylesheet number
font	Alpha	→	Font name

Description

The WR SET STYLESHEET FONT allows you to modify the character font for the style sheet whose number is passed in styleSheetNumber in the 4D Write document referenced by area.

Pass in font the name of the font you want to apply. If you want to apply the style sheet to the selection, pass an empty character string in font.

If font is not installed in the system, the error 1077 (Font not in system) is returned.

Example

See the example for the command WR SET STYLESHEET INFO.

See also

WR Get stylesheet font, WR SET FONT.

WR SET STYLESHEET INFO

WR SET STYLESHEET INFO (area; styleSheetNumber; name; applyTo; shortCut)

Parameter	Type		Description
area	Longint	→	4D Write area
styleSheetNumber	Longint	→	Style sheet number
name	Alpha	→	Name of the style sheet
applyTo	Integer	→	0=characters 1=paragraphs
shortCut	Alpha	→	one character "" if no shortcut

Description

The WR SET STYLESHEET INFO command allows you to modify the properties of the style sheet whose reference number is passed in styleSheetNumber and which is contained in the 4D Write document with the reference number area. The style sheet number corresponds to the order of apperance the style sheet when it is either displayed in the Style sheet drop-down list or in the list in the Style sheets dialog.

• name
If name is an empty string, the original name of the style sheet will not be modified. The name of a style sheet must not exceed 31 characters.

Warning: two style sheets can both have the same name, however they will always have different reference numbers.

• applyTo
If applyTo equals -1, the current value will remain the same.
If applyTo equals 0, the style sheet applies to characters.
If applyTo equals -1, the style sheet applies to paragraphs.

A paragraph style sheet always apply to all the paragraphs of the selection, even if the first or last paragraphs are partially selected. By default a newly created style sheet is a character style sheet.

• shortCut
The shortCut optional parameter allows you to assign a keyboard shortcut to the style sheet. It only accepts one character. To use the shortcut you will need to press the key passed in this parameter with the Ctrl key (on Windows) or the Command key (on Mac OS). It is recommended that you use a number in order to avoid any conflict with the standard 4D Write keyboard shortcuts.
If shortCut is omitted or if it is an empty character string no shortcut will be assigned to the style sheet.

• styleSheetNumber
If you want the style sheet number to remain identical, you need to call the WR GET STYLESHEET INFO command and use the reference number returned by that command .

Example

You want to modify the definition of the “Title” style sheet: its name is changed to “Title 14”, its font should be set to Times 14 with the Bold style attribute selected as well as the blue color.

```
       NbStyles:=WR Count (Area;12)
       For ($i;1;NbStyles)
          WR GET STYLESHEET INFO(Area;$i;$Name;$ApplyTo;$Protected;$Shortcut)
          If ($Name="Title")
⇒            WR SET STYLESHEET INFO(Area;$i;"Title 14";$ApplyTo;$Shortcut)
             WR SET STYLESHEET FONT(Area;$i;"Times")
             WR SET STYLESHEET TEXT PROP(Area;$i;wr font size;14;1)
             WR SET STYLESHEET TEXT PROP(Area;$i;wr bold;1;1)
             WR SET STYLESHEET TEXT PROP(Area;$i;wr text color;212;1)
          End if
       End for
```

See also

WR GET STYLESHEET INFO.

WR SET STYLESHEET TAB

WR SET STYLESHEET TAB (area; stylesheetNumber; tabNumber; position; alignment; fillChar)

Parameter	Type		Description
area	Longint	→	4D Write area
stylesheetNumber	Longint	→	Stylesheet number
tabNumber	Longint	→	Tab number
position	Number	→	New tab position
alignment	Integer	→	New value for the tab alignment
fillChar	String	→	Selected fill character

Description

The WR SET STYLESHEET TAB command allows you to modify the parameters of the tab stop whose number was passed in tabNumber (tabs are counted left to right) belonging to the style sheet whose number was passed in styleSheetNumber (style sheets are counted top to bottom as shown in the style sheets dialog)). The WR SET STYLESHEET TAB command will move the tab stop to position and will set the fill character as well as the alignment of the tab stop.

This command has no effect on the selected text even if it uses the style sheet being modified.

• If you want to update the text that uses that style sheet, call the WR UPDATE STYLESHEET command after modifying the style sheet definition.

• If you want to immediately apply the new tab properties of the style sheet to both the style sheet and the current selection, use the WR APPLY STYLESHEET command.

If a tab stop already exists at the new location in the style sheet, it will be replaced by the tab stop that is the subject of this command.

position is the distance from the left margin to which you want to move the tab stop. position is expressed in the current default unit for the document. If you do not want to change the position of the tab stop, pass -1 in the position parameter.

alignment specifies the type of alignment you want to select for the tab stop. You can either use the value or the constant.

Value	Constant	Text alignment
-1	-	No change
1	wr left tab	Left alignment
2	wr centered tab	Centered
3	wr right tab	Right alignment
4	wr decimal tab	Decimal
5	wr vertical separator tab	Vertical separator

Note: The list of text properties and their references are available in the Appendix D: 4D Write Constants, in the "WR Tabs" theme.

fillCharacter can be any character whose ASCII code is contained between 33 and 127. This character is displayed in the same font as the modified tab stop.

Example

See the example for the WR GET STYLESHEET TAB command.

See also

WR ADD STYLESHEET TAB, WR DELETE STYLESHEET TAB.

WR SET STYLESHEET TEXT PROP

WR SET STYLESHEET TEXT PROP (area; styleSheetNumber; property; value; apply)

Parameter	Type		Description
area	Longint	→	4D Write area
styleSheetNumber	Longint	→	Stylesheet number
property	Longint	→	Number of the property to read
value	Number	→	Value for the property chosen
apply	Integer	→	1 = apply the value to the property 0 = do not apply the value to the property

Description

The WR SET STYLESHEET TEXT PROP command allows you to modify the text attributes of the style sheet whose number is passed in styleSheetNumber.

• If you want all the text that currently uses this style sheet to be updated, call the WR UPDATE STYLESHEET command after modifying teh style sheet definition.

• If you want to immediatly apply with this command the new text properties of the style sheet to both the style sheet and the current selection, use the WR APPLY STYLESHEET command.

•The meaning given to the value parameter depends on the property value used.
If the value for property is constant property wr bold or 0, values for value can either be 1 (True) or 0 (False).
If the value for property is constant property wr font size or 8, values for value can be 9, 10, 12... but it must not exceed 255.

Note: property and value can be set using constants.

Both lists of text properties and text properties values are available in the Appendix D: 4D Write Constants, in the "WR Text properties" and "WR Text properties values" themes.

• Pass 1 in the apply parameter if you want to apply the changes to the property. If you do so, value will define the new value for the property.
• Pass 0 in the apply parameter if you do not want to apply the changes to the property. If you do so, value will have no effect.

Example

See example for command WR SET STYLESHEET INFO.

See also

WR Get stylesheet text prop.

version 6.5

WR UPDATE STYLESHEET (area; stylesheetNumber)

Parameter	Type		Description
area	Longint	→	4D Write area
stylesheetNumber	Longint	→	Stylesheet number

Description

The WR UPDATE STYLESHEET command updates the displayed formatting of all the text using the style sheet referenced by styleSheetNumber in the 4D Write area referenced by area. After this command is executed, all text formatted with the referenced style will be formatted according to the current definition of that style.

Example

You want to replace the tab stops in the "LayoutPar" style sheet and update text areas wherever that style sheet is applied:

```
    `Looking for the style sheet number
$StyleSheetNb:=WR Count(Area;wr nb stylesheets)
For ($i;1;$StyleSheetNb)
    WR GET STYLESHEET INFO(Area;$i;$Name;$ApplyTo;$Prot;$Shortcut)
    If ($Name="LayoutPar")
        SheetNumber:=$i
    End if
End for
    `Getting the number of tab stops in the style sheet
$NbTab:=WR Get stylesheet text prop(Area;SheetNumber;wr tab;Apply)
    `Deleting all tab stops
For ($i;1;$NbTab)
    WR DELETE STYLESHEET TAB(Area;SheetNumber;1)
End for
    `Inserting new tabs
WR ADD STYLESHEET TAB(Area;SheetNumber;10;wr left tab;Char(126))
...
    `Updating each paragraph that the style sheet is applied to
⇒ WR UPDATE STYLESHEET(Area;SheetNumber)
```

See also

WR APPLY STYLESHEET, WR CREATE STYLESHEET.

11

WR Text Manipulation

Text Manipulation, Introduction

The commands and functions of the "WR Text Manipulation" theme allow you to handle text. These commands are useful for placing text into or retrieving text from a 4D Write area.

Standard searching and replacing features are also available in this theme.

WR BACKSPACE

WR BACKSPACE (area)

Parameter	Type		Description
area	Longint	→	4D Write area

Description

The command WR BACKSPACE simulates pressing of the Delete or Backspace key.

If characters are selected in area, they are deleted.

If no characters are selected, WR BACKSPACE acts the same as pressing Delete or Backspace. One character at a time is deleted and the insertion point moves one character to the left. If you do not want this to happen, use the command WR DELETE SELECTION.

See also

WR DELETE PICTURE IN PAGE, WR DELETE SELECTION.

WR DELETE SELECTION

WR DELETE SELECTION (area)

Parameter	Type		Description
area	Longint	→	4D Write area

Description

The WR DELETE SELECTION command allows you to delete the current text selection from the 4D Write area referenced by area.

Using the following statement will have the same effect as using the WR DELETE SELECTION command: WR EXECUTE command (area; wr cmd clear).

Note: The value of the wr cmd clear constant is 6.

If there is no current selection, the command has no effect, unlike the WR BACKSPACE command that would delete the character located before the cursor.

Example

You want to delete all soft hyphens in your document:

```
       `Counting number of occurrences
    HyphenNb:=WR Count(Area;wr nb soft hyphens)
    For($i;1;HyphenNb)
          `Selecting each time the first soft hyphen is found
       WR SELECT(Area;9;1)
          `Deleting it
⇒      WR DELETE SELECTION(Area)
    End for
```

See also

WR BACKSPACE, WR DELETE PICTURE IN PAGE.

version 6.5

WR Direct find (blob; charString; wholeWord; upperCase) → Longint

Parameter	Type		Description
blob	Blob	→	Blob containing a 4D Write area
charString	Alpha	→	Character string to be searched for
wholeWord	Integer	→	0=partial match 1=whole word
upperCase	Integer	→	0=ignore uppercase 1=takes uppercase into account
Function result	Longint	←	Search status

Description

The WR Direct find command allows you to directly search for a character string in a BLOB that contains a 4D Write area. Using this command does not require the BLOB to be opened in a 4D Write area first. This means that this command executes very quickly.

If the character string is found, WR Direct find returns the position of the character string in the text.
If the search was unsuccessful, WR Direct find returns -1.
If blob does not represent the contents of a 4D Write area, WR Direct find returns -2.

wholeWord and upperCase allow you to choose some options for the search:
• If wholeWord equals 1, only the whole word will be searched for. For a string to be found using this option, it must occur between punctuation characters (space, comma and so on). If wholeWord does not equal 1, the character string can either be a whole word or part of a longer word.
• If upperCase equals 1, the search will look for a character string whose case matches the case of the original string.

Example

This example proposes a keyword-based search method that searches in a selection of records. Your database manages cooking recipes. The 4D Write areas are saved in BLOB fields. You want to be able to find all recipes that use a specific ingredient. Here is the corresponding method, which is very fast:

```
    ToFind:=Request("Enter the ingredient(s) to find:")
        `Creating an empty set in which all found records will be placed
    CREATE EMPTY SET([MyRecipes];"FoundRecords")
    ALL RECORDS([MyRecipes])   `Browsing all the table selection
    While (Not(End selection([MyRecipes])))
⇒       If (WR Direct find ([MyRecipes]BlobRecipe_;ToFind;1;1)>0)
                `If the ingredient is found, the record is added to the set
            ADD TO SET([MyRecipes];"FoundRecords")
        End if
        NEXT RECORD([MyRecipes])
    End while
    USE SET("FoundRecords")
    OUTPUT FORM([MyRecipes];"Output")
    MODIFY SELECTION([MyRecipes];*)
```

See also

WR Find.

version 6.5

WR Find (area; charString; wholeWord; upperCase; wrap) → Longint

Parameter	Type		Description
area	Longint	→	4D Write area
charString	Alpha	→	String of characters to be searched for
wholeWord	Integer	→	0=partial match 1=whole word
upperCase	Integer	→	0=ignore uppercase 1=takes uppercase into account
wrap	Integer	→	0=search after the insertion point 1=search the whole document
Function result	Longint	←	Search status

Description

The WR Find allows you to search for a character string in a 4D Write area. You can retrieve the position of the words found using the WR GET WORDS command. You can retrieve the position of the selection found using the WR GET SELECTION command. If the character string is found, WR Find returns 1 and select the first occurence.

If the search was unsuccessful, WR Find returns 0 and the current selection is not modified. If area does not exist, WR Find returns -1.

wholeWord and upperCase allow you to define some options of the search:
• If wholeWord equals 1, only the whole word will be searched for. For a string to be found using this option, it must occur between punctuation characters (space, comma and so on). If wholeWord does not equal 1, the character string can either be a whole word or part of a longer word.
• If upperCase equals 1, the search will look for a character string whose case matches the case of the original string .

A search always starts form the position of the insertion point. wrap allows you to define whether the search applies to the entire document. If wrap equals 1, the search will be performed on the entire document, otherwise the search will be stopped at the end of the document.

Examples

(1) You ask users to enter the searched string, then perform the search:

```
        ToFind:=Request("Enter the word(s) to find:")
        If(OK=1)
            WR SET SELECTION(Area;0;0)
⇒           If(WR Find(Area;ToFind;1;1;1)=0)
                ALERT("No occurrence has been found.")
            End if
        End if
```

(2) This example proposes a keyword-based search method that searches in a selection of records. The search is performed in Picture areas.

Important: If you saved your 4D Write areas in BLOB fields, please refer to the example for the WR Find direct command, which is much faster.

Your database manages cooking recipes. The 4D Write areas are saved in Picture fields. You want to be able to find all the recipes that use a specific ingredient. Here is the corresponding method:

```
        ToFind:=Request("Enter the ingredient(s) to find:")
            `Creating an empty set in which all the found records will be placed
        CREATE EMPTY SET([MyRecipes];"FoundRecords")
        ALL RECORDS([MyRecipes])   `Browsing all the table selection
        OffscreenArea:=WR New offscreen area
        While (Not(End selection([MyRecipes])))
            WR PICTURE TO AREA (OffscreenArea;[MyRecipes]PictRecipe_)
⇒           If (WR Find (OffscreenArea;ToFind;1;1;1)=1)
                    `If the ingredient is found, the record is added to the set
                ADD TO SET([MyRecipes];"FoundRecords")
            End if
            NEXT RECORD([MyRecipes])
        End while
        WR DELETE OFFSCREEN AREA (OffscreenArea)
        USE SET("FoundRecords")
        OUTPUT FORM([MyRecipes];"Output")
        MODIFY SELECTION([MyRecipes];*)
```

See also

WR Direct find.

WR Get font

WR Get font (area; sameFont) → String

Parameter	Type		Description
area	Longint	→	4D Write area
sameFont	Longint	←	1 if the font is the same for the entire selection; otherwise 0
Function result	String	←	Name of the font of the last character of the selection

Description

The WR Get font command returns the font name of the font applied to the last character of the selection in the 4D Write area referenced by area.

• If sameFont = 1, the same font is applied to the whole selection.
• If sameFont = 0, other fonts are used in the selection.

Example

You want to retrieve the font of the current selection and apply it to the entire document:

```
⇒      vFont:=WR Get font(Area;vUniform)
       If (vUniform=0)  `If there are several fonts in the current selection
          CONFIRM("There are several fonts in the selection, the font used for the last
                    "+"character is "+vFont+". OK to apply this font to the entire document?")
       Else
          CONFIRM("The font of the selection is "+vFont+". OK to apply this font to the entire
                                                                                  document?)
       End if
       If (OK=1)
          WR EXECUTE COMMAND(Area;wr cmd selec all)   `Selecting the entire document
          WR SET FONT(Area;vFont)   `Applying the new font
             `Moving the insertion point to the beginning of the document
          WR SET SELECTION(Area;0;0)
          WR SCROLL TO SELECTION(Area)  `Displaying the current text selection
       End if
```

See also

WR Get stylesheet font, WR Get text property, WR SET FONT.

WR GET PARAGRAPHS

version 6.5

WR GET PARAGRAPHS (area; beginPara; endPara)

Parameter	Type		Description
area	Longint	→	4D Write area
beginPara	Longint	←	Beginning of the paragraph to return
endPara	Longint	←	End of the paragraph to return

Description

The WR GET PARAGRAPHS command returns the position of the first character of the first paragraph of the selection and the position of the carriage return of the last paragraph of the selection, in the 4D Write area referenced by area.

Example

The following example scans the document and retrieves the position of the first and last character for each paragraph.

```
        `Locating the cursor at the beginning of the area
    WR SET SELECTION (area;0;0)
        `Counting the number of paragraphs in the document
    nbPara:=WR Count(Zone;wr nb paragraphs)
        `Processing paragraphs one by one
    For ($i;1;nbPara)
            `Retrieving the position of the first and last characters
⇒       WR GET PARAGRAPHS(area;begin;Pos)
            `Relocating after the last processed paragraph
        WR SET SELECTION (area;Pos;Pos)
    End for
```

See also

WR Get selected text, WR GET SELECTION, WR Get text.

WR Get selected text

version 6.0

WR Get selected text (area) → Text

Parameter	Type		Description
area	Longint	→	4D Write area
Function result	Text	←	Text selected in area

Description

The command WR Get selected text returns the selected text in area.

The maximum number of characters 4th Dimension can store in a field or variable is 32,000. Therefore, WR Get selected text will return a maximum of 32,000 characters. If more than 32,000 characters are selected, this function returns an empty string and an error will be reported by WR Error number.

Example

The following example places the selected text in area into the variable *vText*.

```
⇒      vText:=WR Get selected text (area)
```

See Also

WR GET PARAGRAPHS, WR GET SELECTION, WR Get text, WR GET WORDS.

WR GET SELECTION

WR GET SELECTION (area; first; last)

Parameter	Type		Description
area	Longint	→	4D Write area
first	Longint	←	Receives first character
last	Longint	←	Receives last character

Description

The WR GET SELECTION command returns, in the first and last variables, the positions of the selected text in Area.

first is always one less than the first character selected. last is always equal to the last character selected. If first and last are equal, no text is selected and the insertion point is positioned after the character described by first.

Example

The following example sets the margins of the whole document and retrieves the original selection:

```
⇒      WR GET SELECTION(area;StartSel;EndSel)  `Re-reading the current selection
       WR EXECUTE COMMAND(area;wr cmd select all)   `Select all
          `Setting margins
       WR SET TEXT PROPERTY(area;wr left margin;49)
       WR SET TEXT PROPERTY(area;wr first indent;49)
       WR SET TEXT PROPERTY(area;wr right margin;504)
       WR SET SELECTION(area;StartSel;EndSel)  `Resetting the selection
```

See Also

WR Get selected text, WR Get text, WR SET SELECTION.

WR Get styled text

version 6.5

WR Get styled text (area) → BLOB

Parameter	Type		Description
area	Longint	→	4D Write area
Function result	BLOB	←	Formatted text

Description

The WR Get styled text command returns the selected text in the 4D Write area referenced by area a BLOB field or variable. The structure of the BLOB returned represents the selected text with both character and paragraph formatting included, although without style sheets.

Text that is returned using the WR Get styled text command can be placed into another 4D Write document using the WR INSERT STYLED TEXT command. The page layout of the 4D Write document into which the styled text is inserted will not be affected by the insertion.

By using the WR Get styled text and the WR INSERT STYLED TEXT commands you can simulate a Copy/Paste operation while using a BLOB as a buffer instead of the clipboard.

Warning: The BLOB returned by WR Get styled text cannot be used with the WR BLOB TO AREA command since it does not include all the elements of a 4D Write area.

Example

See the example for the WR INSERT STYLED TEXT command.

See also

WR INSERT STYLED TEXT.

WR Get text

WR Get text (area; first; last) → Text

Parameter	Type		Description
area	Longint	→	4D Write area
first	Longint	→	First character of text
last	Longint	→	Last character of text
Function result	Text	←	Text between first and last characters

Description

The command WR Get text returns the text in area between the character described by first and the character described by last.

The maximum number of characters 4th Dimension can store in a field or variable is 32,000. Therefore, WR Get text can return a maximum of 32,000 characters. If the difference between last and first is greater than 32,000 characters, and the document has at least last number of characters, WR Get text returns an empty string and an error is reported by WR Error number.

If last is greater than the number of characters in the document and the difference between last and first is less than 32,000, WR Get text returns all of the characters to the end of the document. If last is less than or equal to first, WR Get text returns an empty string.

WR Get text does not change the selected text in area.

Example

The following example places the first 100 characters of area into the variable *vText*.

```
⇒	vText:=WR Get text (area;0;100)
```

See Also

WR GET PARAGRAPHS, WR Get selected text, WR GET SELECTION.

WR Get text property

version 2003 (Modified)

WR Get text property (area; property; sameProperty) → Real

Parameter	Type		Description
area	Longint	→	4D Write area
property	Integer	→	Property number
sameProperty	Integer	←	1 if the whole selection has that property, 0 if part or all of the selection does not have the property
Function result	Real	←	Depends on the property

Description

The WR Get text property command allows you to determine whether the property passed in property is used in the current selection of the 4D Write area referenced by area.

• If sameProperty is equal to 1, the property is applied to the whole selection.
• If sameProperty is equal to 0, the property is not applied to the whole selection.
The returned value then corresponds to the status of the last character of the selection.

The property parameter lets you set the property to be examined. For more information, refer to the description of the WR SET TEXT PROPERTY command.

If you pass an invalid property number, the error 1075 is returned.

Examples

(1) You want to make sure that margin sizes do not exceed a fixed value:

```
⇒	Left:=WR Get text property(Area;wr left margin;$Uniform)
	If(Left<3)   `Setting the left margin to 3
	   WR SET TEXT PROPERTY(Area;wr left margin;3)
	End if
⇒	Right:=WR Get text property(Area;wr right margin;$Uniform)
	If(Right>43)   `Setting the right margin to 43
	   WR SET TEXT PROPERTY(Area;wr right margin;43)
	End if
```

(2) You want users to be able to set the line spacing and alignment, but you do not want them to have access to menus and rulers. The input form contains a button labeled **Info** and two variables, *LineSpacing* and *Alignment*, each of them attached to a method.

- The following is the object method for the **Info** button, it retrieves information about the current cursor position:

```
⇒   LineSpacing:=WR Get text property(Area;wr line spacing;$Uniform)
    If($Uniform=0)
       ALERT("The selection contains several types of line spacings.")
       $Assign:=True
    Else
       $Assign:=False
    End if
⇒   Alignment:=WR Get text property(Area;wr justification;$Uniform)
    If($Uniform=0)
       ALERT("The selection contains several types of alignments.")
    End if
```

- *LineSpacing* object method sets the user's choice for line spacing:

```
    WR SET TEXT PROPERTY(Area;LineSpacing)
```

- *Alignment* object method sets the user's choice for alignment:

```
    WR SET TEXT PROPERTY(Area;Alignment)
```

- In the On load form event, you hide menus and rulers:

```
    If(Form event=On load)
       WR SET DOC PROPERTY(Area;wr view menubar;0)
       WR SET DOC PROPERTY(Area;wr view rulers;0)
    End if
```

See also

WR SET TEXT PROPERTY.

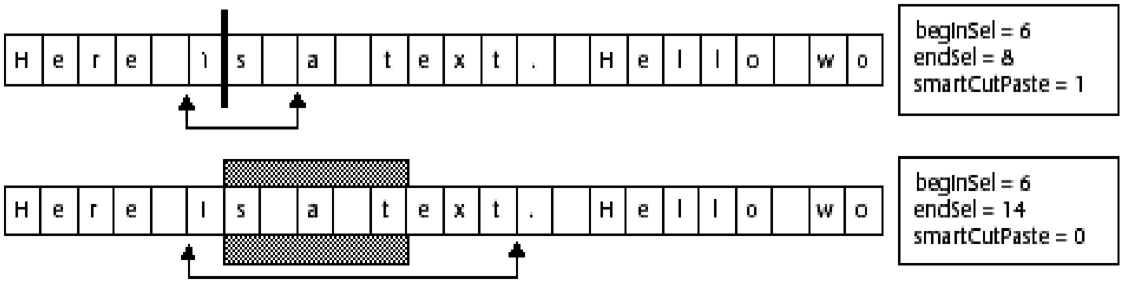
version 6.5

WR GET WORDS (area; beginSel; endSel; smartCutPaste)

Parameter	Type		Description
area	Longint	→	4D Write area
beginSel	Longint	←	Beginning of the word to return
endSel	Longint	←	End of the word to return
smartCutPaste	Integer	←	1 if the last character is a space, otherwise 0

Description

The WR GET WORDS returns the position of the first character of the first word of the selection and the position of the last character of the last word of the selection. It also specifies if the last character of the selection is a space. If no text is selected, beginSel and endSel returns the first and last character of the word the cursor is in.
This command has no effect on the current selection.



If the selection begins in the middle of a word (or between the last character of a word and the next following space), beginSel will return the position of the first character of that word.

If the selection ends in the middle of a word, there are two possible cases:
• If the word is followed by a space, endSel will include the space and smartCutPaste will return 1.
• If the word is not followed by a space, endSel will include the last character of the word and smartCutPaste will return 0.

Example :
The following example scans the document and retrieves the position of the first and last characters for each word.

```
       `Placing the cursor at the beginning of the area
    WR SET SELECTION (area;0;0)
       `Counting the number of words in the document
    nbWords:=WR Count(area;wr nb words)
       `Processing the words one by one
    For ($i;1;nbWords)
          `Retrieving the position of the first and last character of the word
⇒      WR GET WORDS(area;beginning;pos)
          `Relocating after the last processed word
       WR SET SELECTION (area;Pos;Pos)
    End for
```

See also
WR Get selected text, WR Get text.

WR INSERT STYLED TEXT

version 6.5

WR INSERT STYLED TEXT (area; blob)

Parameter	Type		Description
area	Longint	→	4D Write area
blob	BLOB	→	Variable or field

Description

The WR INSERT STYLED TEXT command inserts into the 4D Write area referenced by area the contents of blob. The insertion will either take place at the cursor location or it will replace the current selection. blob can either be a BLOB field or a BLOB variable. It is, however, mandatory that blob was initially created using the WR Get styled text command.

The internal format used to represent the styled text in blob is platform independent. It can be created using a Mac OS computer and be inserted later into a Windows document, or vice versa.

blob contains a selection of 4D Write text with all its text attributes (color, style...) except for style sheets, as well as its paragraph attributes (margins, tab stops, formats...).

Example

You want to store in the table [Letters] the most frequently used templates of your business letters, while still saving hard disk space. To do this, you create in the table a BLOB field called 'Templates'. In the input form for that table, you insert a 4D Write area called 'Area'. Finally, you attach the following method to the form:

```
    Case of
       : (Form event=On Load)
          If (Record number([Letters])#-3)
⇒            WR INSERT STYLED TEXT(Area;[Letters]Templates)
          End if
       : (Form event=On Data Change)
          WR EXECUTE COMMAND(Area;wr cmd select all)
          [Letters]Templates:=WR Get styled text(Area)
    End case
```

See also

WR Get styled text, WR INSERT TEXT.

WR INSERT TEXT

WR INSERT TEXT (area; text)

Parameter	Type		Description
area	Longint	→	4D Write area
text	String	→	First character of text

Description

The command WR INSERT TEXT inserts text into area, replacing any selected characters. If no characters are selected, text is placed at the insertion point. This command can be used in place of WR INSERT EXPRESSION or WR INSERT FIELD when you do not need automatic referencing.

Example

The following example places the text in the variable *vText* into area.

⇒ ***WR INSERT TEXT*** (Area;vText)

See Also

WR INSERT EXPRESSION, WR INSERT FIELD, WR INSERT STYLED TEXT.

WR Mouse to selection

version 6.5 (Modified)

WR Mouse to selection (area; posHoriz; posVert; beginSel; endSel) → Integer

Parameter	Type		Description
area	Longint	→	4D Write area
posHoriz	Integer	→	Horizontal position of mouse in area
posVert	Integer	→	Vertical position of mouse in area
beginSel	Longint	←	Returns beginning of selection
endSel	Longint	←	Returns end of selection
Function result	Integer	←	Selection matching the position of the cursor

Description

The WR Mouse to selection command returns the selection matching the position of the cursor. The command returns 0 if the cursor points to text and returns 1 if it points to a picture.

WR Mouse to selection is used in conjunction with the Drag and Drop manager to find the location of the cursor when the mouse was released and an object was pasted.

beginSel and endSel return different values when you release the mouse button on a reference. Warning: In 4D Write version 6.0 or earlier, the endSel - beginSel expression returns the number of characters contained in the reference after computing. Effective version 6.5, endSel = beginSel +1. In other words, a reference = 1 character regardless of the number of characters contained in the reference, after computing.

The posHoriz and posVert parameters return 0000 by default. In order for them to return a value, you must use the 4th Dimension GET MOUSE command or the AP PICT DRAGGER 4D_Pack routine beforehand. For more information, please refer to these products' documentation.

Example

Consider a 4D Write area that contains a button with a background picture. The button's object method allows you to simulate the drag and drop of the background picture to a location "B" (knowing that the cursor is placed at a location "A").

```
	AP PICT DRAGGER ([Letters]Picture;PosH;PosV)
⇒	NPictureNot:=WR Mouse to selection(Field6;PosH;PosV;StartSel;EndSel)
	WR SET SELECTION(Field6;StartSel;FiEndSelnSel)
	im:=[Letters]Picture*0,5
	WR INSERT PICTURE(Field6;Im)
```

See Also

AP PICT DRAGGER, GET MOUSE.

WR Replace

version 6.5

WR Replace (area; searchedFor; replaceWith; wholeWord; upperCase; replaceAll; wrap) → Longint

Parameter	Type		Description
area	Longint	→	4D Write area
searchedFor	String	→	Character string to search for
replaceWith	String	→	Replacement character string
wholeWord	Integer	→	Search for whole word
upperCase	Integer	→	Take uppercase characters into account
replaceAll	Integer	→	0=replace next 1=replace all
wrap	Integer	→	0=search from the selection 1=search the whole document
Function result	Longint	←	Number of occurrences replaced

Description

The WR Replace command allows you to emulate the Replace command menu of the Edit menu.

If wholeWord equals 1, only whole words will be searched for. For a string to be found using this option, it must occur between punctuation characters (space, comma and so on).

If upperCase equals 1, the search will look for a character string whose case matches the case of the original string .

If replaceAll equals 1, each occurrence of the character string will be replaced. If replaceAll does not equal 1, only the first occurrence will be replaced.

wrap allows you to define whether the search applies to the entire document. If wrap equals 1, the search will be performed on the entire document, otherwise the search will be performed from the position of the selection to the end of the document.

WR Replace returns the number of occurrences replaced.

Example

You want to remove all unnecessary double spaces in your document:

```
            `Assigning a variable that contains double space characters
        ToFind:="  "
            `While occurrences are found
        While(WR Find(Area;ToFind;0;0;1)=1)
                `Replacing double space by a single one
⇒           $n:=WR Replace(Area;ToFind;" ";0;0;1;0)
        End while
```

See also

WR SELECT.

WR SELECT

version 6.7 (Modified)

WR SELECT (area; type; begin{; end})

Parameter	Type		Description
area	Longint	→	4D Write area
type	Integer	→	Type to select
begin	Longint	→	First character
end	Longint	→	Last character. Optional for certain values of type

Description

The WR SELECT command selects text defined by type, begin, and end.

Type	Selection	Comments
0	Characters	Selects the characters located between begin and end. In this case, this is the same as using WR SET SELECTION.
1	4D Expression	Selects the reference whose rank in the document is defined by begin. end must be omitted.
2	Paragraphs	Selects the paragraphs located between begin and end.
3	Ruler (paragraph attributes)	Selects the paragraphs that use the Xth ruler (whose rank in the document starts at the beginning of the text). end must be omitted.
4	Picture	Selects the picture whose rank in the document is defined by begin. end must be omitted.
5	Style (character attributes)	Selects the words that use the Xth style (whose rank in the document starts at the beginning of the text). end must be omitted.
6	Word	Selects the word in which the insertion point is located.
7	Page break	Selects the page breaks whose rank in the document is defined by begin. end must be omitted.
8	Column break	Selects the column breaks whose rank in the document is defined by begin. end must be omitted.
9	Hyphen	Selects the hyphen whose rank in the document is defined by begin. end must be omitted.
10	Page number	Selects the page number whose rank in the document is defined by begin. end must be omitted. The selection only carries over to page numbers inserted into the body of text.
11	Date and time	Selects the date and time variable whose rank in the document is defined by begin. end must be omitted. The selection only carries over to the dates or times automatically updated and inserted into the body of text.
12	Hyperlink	Selects the hyperlink whose rank in the document is defined by begin. end must be omitted.

13	HTML Expression	Selects the HTML expression whose rank in the document is defined by begin. end must be omitted.
14	RTF Expression	Selects the RTF expression whose rank in the document is defined by begin. end must be omitted.

Examples

(1) The following example executes different functions based on the presence or the absence of a Page break:

```
      `Setting the selection
   WR SET SELECTION (area;0;0)
      `Try to select the first page break
⇒  WR SELECT (area;7;1)
      `Retrieving the limits of the new selection
   WR GET SELECTION (area;$vlbegin;$vlend)
   If (($vlbegin=0) & ($vlend=0))
         `There is no page break
   Else
         `Do something with the page break
   End if
```

(2) The following example selects the references in the 4D Write area referenced by area and applies to them a style that makes them easy to spot:

```
   NbObjects:=WR Count(area;4)
      `Counting the number of references
   For (i;1;NbObjects)
⇒     WR SELECT(area;1;i)
         `Selecting each reference
      WR GET REFERENCE(area;TableNo;FieldNo;vName;vType)
      WR SET TEXT PROPERTY(area;wr bold;1)
      WR SET TEXT PROPERTY(LaZone;wr text color;wr blue)
         `Applying Blue and Bold to the selection
   End for
```

See Also

WR Count, WR Replace, WR SELECT PICTURE IN PAGE.

WR SET FONT

WR SET FONT (area; font)

Parameter	Type		Description
area	Longint	→	4D Write area
font	String	→	Font name

Description

The WR SET FONT allows you to set the font for the current selection in the 4D Write area referenced by area.

Pass in font the name of the font you want to apply. If font is not installed in the system, the error 1077 is returned.

Example

See the example for the command WR Get font.

See also

WR FONTS TO ARRAY, WR Get font, WR SET STYLESHEET FONT.

WR SET SELECTION

WR SET SELECTION (area; first; last)

Parameter	Type		Description
area	Longint	→	4D Write area
first	Longint	→	First character
last	Longint	→	Last character

Description

The command WR SET SELECTION selects the text in area described by the numbers first and last. Text is selected from first + 1 characters to last.

If first and last are equal, WR SET SELECTION places the insertion point after the character described by first. If last is greater than the length of the text in Area, WR SET SELECTION selects the text to the end of the document. If last is less than first, WR SET SELECTION does nothing.

Example

The following example selects the first ten characters in area:

```
⇒	WR SET SELECTION (area;0;10)
```

See Also

WR GET SELECTION.

WR SET TEXT PROPERTY

version 2003 (Modified)

WR SET TEXT PROPERTY (area; property; value)

Parameter	Type		Description
area	Longint	→	4D Write area
property	Integer	→	Number of the text property to set
value	Number	→	Value for the selected property

Description

The WR SET TEXT PROPERTY command allows you to modify the text properties of the current selection in the 4D Write area referenced by area.

property and value are to be used jointly.

Tip: We advise you to use the WR SET FONT command instead of WR SET TEXT PROPERTY (Area;wr font number;Value), because font numbers are managed dynamically and may be different between operating systems.

If you pass an illegal property number, the error 1075 will be generated.
If you pass an illegal value for the selected property, the error 1076 will be generated.

Notes:
• property and value can be set using constants. A list of text properties and a list of values for text properties values are available in Appendix D: 4D Write Constants, in the "WR Text properties" and "WR Text properties values" themes. You can either pass the value or the constant.
•The list of error codes is available in Appendix C: Error Codes.

The following constants and values can be used with the WR SET TEXT PROPERTY and WR Get text property commands:

property (WR Text properties)	used to set or get (value or WR Text properties values) :
wr bold (0)	the bold style on the text (false=0, true=1)
wr italic (1)	the italic style on the text (false=0, true=1)
wr shadow (2)	the shadow style on the text (false=0, true=1)
wr strikethrough (3)	the strikethrough style on the text (false=0, true=1)
wr underline (4)	the underline style on the text: no underline=0, wr single underline (1), wr word underline (2), wr double underline (3), wr hatched underline (4)
wr superscript or subscript (5)	text in superscript or subscript: none=0, wr superscript (1), wr subscript (2)

wr capital case (6)	text in small capitals, capitals or lower case: lower case=0, wr capitals (1), wr small capitals (2)
wr font number (7)	the value passed is an internal number. 4D Write assigns font numbers gradually as they are used. It is generally advisable to use the WR Get font and WR SET FONT commands that work with font names.
wr font size (8)	the size of the text (value between 9 and 255)
wr text color (9) wr text back color (10) wr strikethrough color (11) wr underline color (12) wr shadow color (13)	the value must be passed in the form 0x00RRGGBB as in 4th Dimension (or in the previous version of 4D Write). You can use the constants of the WR Standard colors theme
wr links appearance (14)	the appearance of the links: wr no links appearance (0), wr unvisited links appearance (1), wr visited links appearance (2)
wr stylesheet number (15)	pass the index of the stylesheet in the list. Keep in mind that if you pass a stylesheet index, the text will be assigned a stylesheet, but the properties of this stylesheet will not be applied to it. The WR APPLY STYLESHEET command both sets the property and applies the properties of the stylesheet.
wr user property (16)	its value can be set freely. You can set and get any customized value for this property. For example, if you want to keep a hierarchical list in parallel with a text, you can use this property to store an element reference for the hierarchical list. Each time you click on the text, you get the property and select the corresponding element in the hierarchical list.
wr justification (32)	text justification: wr left justified (0), wr centered (1), wr right justified (2), wr full justified (3)
wr line spacing (33)	the line spacing, the value varies from 1 to 10 in steps of 0.5: 1=single spacing, 1.5=1.5 spacing, 2=double spacing
wr bullet (34)	the bullet style: wr black square bullet (110), wr white square bullet (111), wr black circle bullet (108), wr white circle bullet (109), wr diamonds bullet (117), wr clubs bullet (118), wr no bullet (0)
wr left margin (35)	the distance with respect to the left dead margin. The value is expressed in the current unit of the document.
wr first indent (36)	the distance with respect to the right margin. <0 = to the left of the right margin, >0 = to the right of the right margin. The value is expressed in the current unit of the document.
wr right margin (37)	the distance with respect to the right dead margin. The value is expressed in the current unit of the document.

wr border back color (38) wr border line color (39)	the value must be passed in the form 0x00RRGGBB as in 4th Dimension (or in the previous version of 4D Write). You can use the constants of the WR Standard colors theme.
wr border line style (40)	the style and size of the border line: 0=1-pt line, 1=2-pt line, 2=3-pt line, 3=dotted line, 4=double dotted line, 5= triple dotted line, 6=double 1-pt line, 7=double inside 2-pt line, 8=triple center 2-pt line, 9=double outside 2-pt line, (2003) 10=1/2-pt line, (2003) 11=1/4-pt line. Setting the border line style directly affects the borders of the selection, or lets you set the type of border before putting it in place. It is better to set the type of border first and then to place them. That way, you avoid having to redraw. Keep in mind that the border style is the same for the all the sides (left/right and top/bottom) of a selection.
wr left border (41)	setting of the border (false=0, true=1)
wr right border (42)	setting of the border (false=0, true=1)
wr top border (43)	setting of the border (false=0, true=1)
wr bottom border (44)	setting of the border (false=0, true=1)
wr border spacing (45)	the distance between the border and text. The value is expressed in the current unit of the document.
wr tab (64)	the number of tabs in the last paragraph of the selection. Property not valid with WR SET TEXT PROPERTY — to be used only with WR Get text property.

Examples

(1) You want to apply to the current selection the following properties: Times font, 12 points, Violet color, no italic, bold.

```
    Violet:=WR RGB to color(61952;2048;33792)
    WR SET FONT(Area;"Times")
⇒   WR SET TEXT PROPERTY(Area;wr font size;12)
⇒   WR SET TEXT PROPERTY(Area;wr text color;wr violet)
⇒   WR SET TEXT PROPERTY(Area;wr bold;1)
⇒   WR SET TEXT PROPERTY(Area;wr italic;0)
```

(2) You want to set the margins to a predefined value:

```
    WR GET SELECTION(Area;StartSel;EndSel)   `Storing the current text selection
    WR UPDATE MODE(Area;0)   `Disabling screen updating
    WR EXECUTE COMMAND(Area;wr cmd select all)   `Selecting all
       `Setting the document unit to centimeters
    WR SET DOC PROPERTY(Area;wr unit;0)
```

```
	`Setting the document margins in centimeters
⇒	WR SET TEXT PROPERTY(Area;wr right margin;1,8)
⇒	WR SET TEXT PROPERTY(Area;wr left margin;1,3)
	WR SET SELECTION(Area;StartSel;EndSel)   `Setting back the selection
	WR UPDATE MODE(Area;1)   `Enables screen updating
```

See also

WR Get text property.

12

WR Utilities

Utilities, Introduction

version 6.5 (Modified)

The commands and functions of the "WR Utilities" theme provide utilities for activities such as handling errors and events, allowing you to control your 4D Write areas.

The WR Count function allows you to get basic information on the contents of your 4D Write area. The WR FONTS TO ARRAY command lists the fonts currently installed in your Operating System.

Also, the color management commands enable you to manage the display of colors in your 4D Write areas.

WR COLOR TO RGB

version 6.0

WR COLOR TO RGB (color; red; green; blue)

Parameter	Type		Description
color	Longint	→	Color
red	Longint	←	Receives red value (0 to 65535)
green	Longint	←	Receives green value (0 to 65535)
blue	Longint	←	Receives blue value (0 to 65535)

Description

The command WR COLOR TO RGB maps the color defined by color into its three components: red, green, and blue. These values range from 0 to 65535.
color is an internal number used by 4D Write and can be obtained with the WR RGB to color function.

Example

The following example calculates the closest grey for a given color:

```
⇒      WR COLOR TO RGB (Color;Red;Green;Blue)
       Blue:=(Blue+Green+Red)/3
       Grey:=WR RGB To color (Blue;Blue;Blue)
```

See also

WR RGB to color.

WR Count

version 6.7 (Modified)

WR Count (area; objectNumber) → Longint

Parameter	Type		Description
area	Longint	→	4D Write area
objectNumber	Integer	→	Object number
Function result	Longint	←	Number of objects

Description

The WR Count command allows you to count the number of occurrences of a specific object in a specific area.

Objects that can be counted are:

Object	Constants	ObjectNumber
Characters	wr nb characters	0
Words	wr nb words	1
Paragraphs	wr nb paragraphs	2
Picture in text flow	wr nb pictures in text flow	3
References	wr nb objects	4
Hyphens	wr nb soft hyphens	5
Page breaks	wr nb page breaks	6
Column breaks	wr nb column breaks	7
Time objects	wr nb insertions date time	8
Page numbers	wr nb insertions page number	9
Lines	wr nb lines	10
Pages	wr nb pages	11
Style sheets	wr nb stylesheets	12
Images in pages (background)	wr nb pictures in page	13
Hyperlinks	wr nb hyperlinks	14 (6.7)
RTF Expressions	wr nb RTF expressions	15 (6.7)
HTML Expressions	wr nb HTML expressions	16 (6.7)

• If objectNumber equals 3, background pictures will be ignored (if you want background pictures to be counted, objectNumber must equal 13).

• If objectNumber equals 12, WR Count returns the number of style sheets, including the standard style sheets (default style sheet).

• If objectNumber equals 13 and if an image is repeated in several pages (as selected in the picture properties dialog), the image counts as one.

If you pass a wrong area reference to the command, the error 1022 will be returned.

Examples

See examples for the following commands: WR SELECT, WR INSERT PAGE NUMBER, WR DELETE PICTURE IN PAGE, WR GET WORDS, WR GET PARAGRAPHS and WR UPDATE STYLESHEET.

See also

WR Replace, WR SELECT.

WR Error number

WR Error number (area) → Integer

Parameter	Type		Description
area	Longint	→	4D Write area
Function result	Integer	←	Status of the last operation performed in Area by 4D Write

Description

The command WR Error number returns a number that represents the status of the last operation performed in Area by 4D Write. If WR Error number equals 0, the last operation did not cause an error. If WR Error number does not equal 0, then an error occurred during the last operation in area.

Use WR Error text to get a text explanation of the error. If the Debug window is open and an error occurs, you will also receive the error number in the Debug window.

Example

See example for command WR Error text.

See Also

Appendix C: Error Codes, WR Error text, WR ON ERROR.

WR Error text

WR Error text (error) → String

Parameter	Type		Description
error	Integer	→	Number of error
Function result	String	←	Text description of the error specified by Error

Description

The command WR Error text returns a text description of the error specified by error. You can use this function to receive a description of the error returned by WR Error number.

Example

The following example tests for an error and then displays a different error message depending upon whether or not the user is the Designer:

```
⇒   $Error:=WR Error number (Area)
    If ($Error#0)
       If (Current user="Designer")
⇒         ALERT (WR Error text ($Error))
       Else
          ALERT ("A problem has occurred. Please notify your manager.")
       End if
    End if
```

See Also

Appendix C: Error Codes, WR Error number, WR ON ERROR.

WR FONTS TO ARRAY

version 6.0

WR FONTS TO ARRAY (fonts)

Parameter	Type		Description
fonts	String array	←	Receives array of available fonts

Description

The command WR FONTS TO ARRAY returns the list of available fonts in the fonts array. This list corresponds to the font drop-down list located in the Style palette.

fonts should be declared as a String or Text type array.

Example

You want to check if the fonts required for your templates are installed in the current system. The [Fonts] table stores the list of required fonts. In the On Startup Database Method, you can write:

```
    ARRAY TEXT (aFonts;0)
⇒   WR FONTS TO ARRAY (aFonts)
    ALL RECORDS([Fonts])
    While(Not(End selection([Fonts])))
        If (Find in array(aFonts;[Fonts]Name)=-1)
            ALERT("The font "+[Fonts]Name+" is required, please install it.")
        End if
        NEXT RECORD([Fonts])
    End while
```

See Also

WR SET FONT.

WR ON ERROR

WR ON ERROR (method)

Parameter	Type		Description
method	String	→	Name of method

Description

The WR ON ERROR command installs an interruption method defined and specified by method. This interruption method will be executed every time an error occurs during calls to 4D Write commands. This will allow monitoring of possible execution errors from within your application.

The called method will receive the 3 following parameters:
• $1 represents the area,
• $2 represents the error number,
• $3 represents the error text.

Note: Due to database compilation, $1 and $2 must be declared as Long integers and $3 as Text.

Once method execution is finished, 4th Dimension will return to the interrupted formula.If method is an empty string, WR ON ERROR uninstalls the previously installed error method.

Example

You want to install an error management method for 4D Write.

```
		` Call method
⇒	WR ON ERROR("WriteArea")

		` The WriteArea method displays the number and the error description that
provoked the call
	ALERT ("Error number "+String($2)+Char(13)+$3)
```

See Also

Appendix C: Error Codes, WR Error number, WR Get on error method, WR ON EVENT.

WR Get on error method

version 6.7.2

WR Get on error method → String

Parameter	Type		Description
This command does not require any parameters			
Function result	String	←	Name of on error method

Description

The WR Get on error method command returns the on error method installed by WR ON ERROR.

If no on error method has been installed, an empty string ("") is returned.

See also

WR ON ERROR.

WR ON EVENT

version 6.0

WR ON EVENT (area; event; method)

Parameter	Type		Description
area	Longint	→	4D Write area
event	Longint	→	Event code
method	String	→	Method to execute

Description

The WR ON EVENT command installs method as the method to be called whenever the event described by event occurs in area. Events are passed directly to method before being handled by 4D Write.

If area equals 0, method becomes the default event method for all 4D Write areas until the database is closed. If an area has a specific event method installed, that method is called instead of the default.

The following table lists the possible values for event:

Event	Value
Event method is activated for all events	-1
Key down (including arrow keys, returns, tabs, etc.)	0
Double-click	1
Mouse-click	2
Unused (for compatibility with 4D Calc)	3
Unused (for compatibility with 4D Calc)	4
4D Write area activated or deactivated	5
Unused (for compatibility with 4D Calc)	6
Printing document	7
Ruler modification	8
Dynamic reference modified	9
4D Write area closing	10

When called, method receives six parameters that describe the state of area at the time of the event. You must explicitly type these parameters using compiler directives. The following table describes the parameters received by method:

Variable	Type	Description
$1	Long integer	4D Write area
$2	Integer	Shift key
$3	Integer	Alt (Windows); Option (Mac OS)
$4	Integer	Ctrl (Windows); Command key (⌘) (Mac OS)
$5	Integer	Event type
$6	Integer	Changes depending on event type

$1 returns the long integer that is the area ID where the event took place. $2, $3, and $4 describe whether a specific modifier key was depressed at the time of the event. If the value equals 0, the key was not pressed. If the value equals 1, the key was pressed. $5 returns the event type. $6 varies depending on the type of event.

Method Variables and the Event Parameter ($6)

If event equals 0, $6 returns the ASCII code of the key calling the event.

If event equals 1 or 2, $6 indicates whether you single- or double-clicked a reference. If $6 equals 0, no reference was selected. If $6 equals 1, a reference was selected. method can be called if you perform one of the following actions:

• Single- or double-click a reference

• Right-click (on Windows) or Control-click (on Mac OS).

On Mac OS, pressing the Control key while clicking typically displays a pop-up menu. On Windows, right-clicking typically displays a drop-down menu.

If event equals 5, $6 describes whether or not the area is activated. If $6 equals 0, the 4D Write area is deactivated. If $6 equals 1, the 4D Write area is activated.

If event equals 7 and the print job is a mail merge, $6 indicates the table number for the table used. If the print job is not a mail merge, $6 equals 0.

If event equals 9, $6 indicates where margins have been reset in the document. If $6 equals 0, the margins have been reset in the body. If $6 equals 1, the margins have been reset in the header. If $6 equals 2, the margins have been reset in the footer.

To filter characters, you must use method as a function that returns 0 or 1. This enables you to specify characters in the document that 4D Write will ignore.

Initialize $0 to 1 to make the method trap a particular event. Initialize $0 to 0 if you do not want to trap a particular event. For example, if you do not want the character “@” to appear in your document, filter all characters appearing in the document. If the $6 variable is equal to the ASCII code of the “@” character, you initialize $0 to 1 and ignore it.

Note: If you filter all characters, operations may slow down considerably since the method will be called for each keystroke.

Example

In the following examples, some actions are executed depending on the type of event:

```
      `Form method:
   If (Form event=On load)
⇒     WR ON EVENT (Area;0;"ProcName")
         `Call for all key strokes
⇒     WR ON EVENT (Area;5;"ProcName")
         `Check for area status
      DISABLE MENU ITEM(2;1)
         `Disable menu item "Change font"
   End if


      `ProcName method:
   Case of
      : ($5=0)
            `Intercepts the key strokes
         If ($6=199) | ($6=200)
               `ASCII codes corresponding
            BEEP
            $0:=1
         Else
               `Leave the event to 4D Write
            $0:=0
         End if
      : ($5=5)
            `Intercept change in status of area
         If ($6=0)
               `If the area is inactive
            DISABLE MENU ITEM(2;1)
         Else
            ENABLE MENU ITEM(2;1)
         End if
   End case
```

See Also

WR Get on event method, WR ON ERROR.

WR Get on event method

WR Get on event method (area; event) → String

Parameter	Type		Description
area	Longint	→	4D Write area
event	Longint	→	Event code
Function result	String	←	Name of the installed on event method

Description

The WR Get on event method command allows knowing the name of the on event method installed by WR ON EVENT for the event defined by the event parameter in the specified 4D Write area.

If no on event method has been installed, an empty string ("") is returned.

The events that can be passed in the event parameter are as follows:

Event	Code
Character entry (including arrows, carriage returns, tabs...)	0
Double-click	1
Mouse click	2
Activated or deactivated 4D Write area	5
Print document	7
Ruler modification	8
Dynamic reference modified	9
Close a 4D Write window or area	10

See also

WR ON EVENT.

WR RGB to color

WR Utilities

version 6.0

WR RGB to color (red; green; blue) → Longint

Parameter	Type		Description
red	Longint	→	Red component (0 to 65535)
green	Longint	→	Green component (0 to 65535)
blue	Longint	→	Blue component (0 to 65535)
Function result	Longint	←	Color

Description

The command WR RGB to color returns a compact number that is used by 4D Write to manage colors. This number represents the three component colors: red, green, and blue. The red, green, and blue parameters are the same values used in your system's color picker. These values range from 0 to 65535.

The following table shows the values for red, green, and blue in commonly used colors:

Color	Red	Green	Blue
Red	56576	2048	1536
Green	0	32768	4352
Blue	0	0	54272
Cyan	512	43776	59904
Magenta	64512	62208	1280
Yellow	61952	2048	33792

Example

The following example returns a color between two colors:

```
        WR COLOR TO RGB (c1;r1;g1;b1)
        WR COLOR TO RGB (c2;r2;g2;b2)
⇒       c3:=WR RGB To color ((r1+r2)/2;(g1+g2)/2;(b1+b2)/2)
```

See Also

WR COLOR TO RGB.

13

WR Obsolete Commands

Obsolete Commands, Introduction

version 6.5 (Modified)

The commands and functions in the "WR Obsolete Commands" theme come from previous versions of 4D Write. Starting with version 6.5, each of them can be replaced by 4D Write 6.5 new commands or functions, which allow you to make the best possible use of 4D Write's new features.

To ensure compatibility between 4D Write 6.5 and existing applications, the "Obsolete Commands" are still maintained. They work in the same way as the original ones did. However, in 4D Write 6.5, all these commands are prefixed with the letter "O".

In each command description of this theme, you are provided with the alternative new command proposed by 4D Write. If you want your application to be fully compatible with future versions of 4D Write, we strongly recommend you use 4D Write 6.5 new commands.

Note: In addition, several commands and functions that were in previous versions of 4D Write have been removed from 4D Write 6.5. For more information, please refer to Appendix E: Removed V6.0.x Commands.

WR O Area to picture (area) → Picture

Parameter	Type		Description
area	Longint	→	4D Write area
Function result	Picture	←	Picture of the document in area

Note 6.5: This command was only maintained for compatibility purposes. We recommend using the WR Area to picture command.

Description

The command WR O Area to picture returns a 4th Dimension picture that contains the document in area. The resulting picture is equivalent to the value that 4D Write automatically stores in a picture field. Use this command to save area to a field or to retrieve a 4D Write document from an offscreen area.

Note: The picture returned by this command is a Picture type. The result of this function must be put in either a 4D picture variable or field.

See Also

WR Area to picture.

WR O AUTO SAVE

WR O AUTO SAVE (area)

Parameter	Type		Description
area	Longint	→	4D Write area

Note 6.5: This command was only maintained for compatibility purposes. We recommend using the WR Area to picture command.

Description

The command WR O AUTO SAVE causes the document in area to be saved in a picture field with the same name, if it exists. This command automatically saves a 4D Write area to a picture field. Use WR O AUTO SAVE whenever a 4D Write area is changed procedurally and when the changes will be saved in the corresponding picture field.

See Also

WR Area to picture.

version 6.0

WR O CHANGE STYLE (area; delete; add)

Parameter	**Type**		**Description**
area	Longint	→	4D Write area
delete	Integer	→	Style to be removed
add	Integer	→	Style to be set

Note 6.5: This command was only maintained for compatibility purposes. We recommend using the WR SET TEXT PROPERTY command.

Description

The command WR O CHANGE STYLE sets the style of the text you select.

delete and add are numbers obtained by adding different style numbers together. add is the style parameter that you want the selected text to have. delete is the style parameter that you do not want in the selected text. Style numbers are presented in the following list.

Style	**Value**
Plain	0
Bold	1
Italic	2
Underline	4
Shadow	16
Superscript	32
Subscript	64

Note to Version 6.0 Users: The Outline style does not exist in 4D Write 6.5. Passing its value (8) has no effect.

See Also

WR SET TEXT PROPERTY.

WR O COMPUTE NOW

version 6.0

WR O COMPUTE NOW (area)

Parameter	Type		Description
area	Longint	→	4D Write area

Note 6.5: This command was only maintained for compatibility purposes. We recommend using the WR EXECUTE COMMAND command.

Description

The command WR O COMPUTE NOW recalculates the variable elements (inserted expressions, references, etc.) of the 4D Write document's area parameter.

See Also

WR EXECUTE COMMAND.

WR O Count Stylesheet

version 6.0

WR O Count Stylesheet (area) → Integer

Parameter	Type		Description
area	Longint	→	4D Write area
Function result	Integer	←	Number of style sheets available in area

Note 6.5: This command was only maintained for compatibility purposes. We recommend using the WR Count command.

Description

The command WR O Count Stylesheet returns the number of style sheets available in area.

See Also

WR Count.

WR O CREATE STYLESHEET

WR O CREATE STYLESHEET (area; styleNum; name; font; size; style; color)

Parameter	Type		Description
area	Longint	→	4D Write area
styleNum	Integer	→	Creation order number of the style sheet
name	String	→	Style sheet title
font	Integer	→	Font chosen
size	Integer	→	Size of chosen font
style	Integer	→	Style used
color	Longint	→	Color used

Note 6.5: This command was only maintained for compatibility purposes. We recommend using the WR Create Stylesheet command.

Description

The command WR O CREATE STYLESHEET creates a new style and inserts the list at the position specified by styleNum. Define the new style sheet by assigning values to the name, font, size, style, and color parameters.

name is the name of the style sheet and can have a maximum of 31 characters.

font is the number of the font. Use the WR O Font name function to obtain the integer corresponding to a font.

size is the point size of the font and must be between 1 and 127. If you specify a font size large than 127, it will be ignored.

style is expressed as the sum of the styles used. See the WR O SET STYLESHEET command for a list of styles and their corresponding codes.

color is a long integer that represents the color of the text. Use the WR RGB to color function to obtain the long integer corresponding to a color. See the WR O GET STYLESHEET command for a list of commonly used colors and their values.

See Also

WR CREATE STYLESHEET.

WR O DELETE STYLESHEET

WR O DELETE STYLESHEET (area; styleNum)

Parameter	Type		Description
area	Longint	→	4D Write area
styleNum	Integer	→	Style number in the style sheet

Note 6.5: This command was only maintained for compatibility purposes. We recommend using the WR DELETE STYLESHEET command.

Description

The command WR O DELETE STYLESHEET deletes the style sheet specified by styleNum.

See Also

WR DELETE STYLESHEET.

WR O DISPLAY MENUBAR (area; mode)

Parameter	Type		Description
area	Longint	→	4D Write area
mode	Integer	→	1=Hide 0=Show

Note 6.5: This command was only maintained for compatibility purposes. We recommend using the WR SET DOC PROPERTY command.

Description

The command WR O DISPLAY MENUBAR displays or hides the menu bar in area. If mode equals 1, the menu bar is hidden. If mode equals 0, the menu bar is visible.

See Also

WR SET DOC PROPERTY.

WR O DISPLAY RULER (area; mode)

Parameter	Type		Description
area	Longint	→	4D Write area
mode	Integer	→	1=Hide 0=Show

Note 6.5: This command was only maintained for compatibility purposes. We recommend using the WR SET DOC PROPERTY command.

Description

The command WR O DISPLAY RULER displays or hides the ruler in area. If mode equals 1, the ruler is hidden. If mode equals 0, the ruler is displayed.

See Also

WR SET DOC PROPERTY.

WR O DISPLAY SCROLLBARS

version 6.0

WR O DISPLAY SCROLLBARS (area; displayed)

Parameter	Type		Description
area	Longint	→	4D Write area
displayed	Integer	→	0=Scroll bars hidden 1=Scroll bars displayed

Note 6.5: This command was only maintained for compatibility purposes. We recommend using the WR SET DOC PROPERTY command.

Description

The command WR O DISPLAY SCROLLBARS allows you to display or hide the 4D Write document's scroll bars.

If the value of displayed parameter equals 1, the scroll bars are shown; if it equals 0, the scroll bars are hidden.

See Also

WR SET DOC PROPERTY.

WR O DO COMMAND (area; command{; modifiers})

Parameter	Type		Description
area	Longint	→	4D Write area
command	Integer	→	Command number
modifiers	Longint	→	Modifier key(s) pressed

Notes 6.5:
• This command was only maintained for compatibility purposes. We recommend using the WR EXECUTE COMMAND command.
• Also, the WR O DO COMMAND command does not work with the following commands:
405 (outline style)
702 (subscribe hotlink)
708 (publish hotlink)

Description

The command WR O DO COMMAND executes the menu item specified by command. The menu item will be executed as if you had chosen it from a 4D Write menu.

Note: The command numbers used by WR O DO COMMAND are not documented in the current version of the manual. Please refer to the 4D Write 6.0.x documentation or use the WR EXECUTE COMMAND command.

The following modifier keys can be used, expressed as the sum of key numbers:

Key	Value
Command	256
Shift	512
Caps Lock	1024
Option	2048
Control	4096

See Also

WR EXECUTE COMMAND.

WR O EXPERT COMMAND (area; command; status)

Parameter	**Type**		**Description**
area	Longint	→	4D Write area
command	Integer	→	Command number
status	Integer	→	0=Enabled 1=Disabled

Note 6.5: This command was only maintained for compatibility purposes. We recommend using the WR LOCK COMMAND command.

Description

The command WR O EXPERT COMMAND enables or disables the menu items. If status equals 0, the menu item is enabled. If status equals 1, the menu item is disabled. The WR O EXPERT COMMAND command cannot enable commands disabled by 4D Write.

Note: The command numbers used by the WR O EXPERT COMMAND command are not documented in the present manual. Please refer to the 4D Write 6.0.x documentation or use the WR LOCK COMMAND command.

To disable a menu, pass the menu number followed by 00. For example, to disable the File menu, pass 100 for command.

See Also

WR LOCK COMMAND.

WR O Find

WR O Find (area; criteria{; where{; method}}) → Integer

Parameter	Type		Description
area	Longint	→	4D Write area
criteria	Text	→	String for which to search
where	Integer	→	0=Partial word 1=Whole word
method	Integer	→	0=Not case sensitive 1=Case sensitive
Function result	Integer	←	0 if criteria is not found, 1 if criteria is found

Note 6.5: This command was only maintained for compatibility purposes. We recommend using the WR Find command.

Description

The command WR O Find searches for criteria in area, and if found, selects it. If criteria is not found, WR O Find returns 0. If criteria is found, WR O Find returns 1. WR O Find always begins searching from the position of the last character in the selected text. Because WR O Find selects criteria, its position can be obtained from the WR GET SELECTION command.

The optional where parameter determines whether criteria can match part of a word. If where equals 0, WR O Find selects criteria even if criteria is part of another word. If where equals 1, WR O Find will select criteria only if criteria is found as a whole word, criteria is surrounded by separator characters such as spaces or punctuation marks.

The optional method parameter determines whether the search will be case sensitive. If method equals 0, WR O Find selects criteria regardless of case. If method equals 1, WR O Find will select criteria only if the cases match.

See Also

WR Find.

WR O Font name (fontNumber) → Text

Parameter	Type		Description
fontNumber	Integer	→	Font number
Function result	Text	←	Name of the font whose ID is fontNumber

Note 6.5: This command was only maintained for compatibility purposes. We recommend using the 4D commands.

Description

The command WR O Font name returns the name of the font whose ID is fontNumber. The ID is the same value returned by WR O Font number. If fontNumber does not exist, WR O Font name returns an empty string.

WR O Font number

WR O Font number (fontName) → Integer

Parameter	Type		Description
fontName	Integer	→	Name of the font
Function result	Integer	←	ID for the font named fontName

Note 6.5: This command was only maintained for compatibility purposes. We recommend using the 4D built-in commands.

Description

The command WR O Font number returns the integer ID for the font named fontName. This ID can be used in the WR O SET ATTRIBUTES command. If fontName does not exist, WR O Font number returns 0.

WR O GET ATTRIBUTES (area; font; size; style; color)

Parameter	Type		Description
area	Longint	→	4D write area
font	Integer	←	Receives font number
size	Integer	←	Receives font size
style	Integer	←	Receives font style
color	Longint	←	Receives font color

Note 6.5: This command was only maintained for compatibility purposes. We recommend using the WR Get text property command to manage the sizes, styles and colors and the
WR Get font command to manage fonts.

Description

The command WR O GET ATTRIBUTES returns into the variables font, size, style, and color the attributes of the selected text in Area. If the selected text in area contains more than one font, size, style, or color, WR O GET ATTRIBUTES returns -1 for that attribute.

font is the ID of the font in your system. This is the same value that is returned by WR O Font number. You can use the WR Font name function to determine the name of the font.

size is the size in points of the selected text.

style is a composite number that results from the addition of several style numbers. Style numbers are shown in the following list:

Style	Value
Plain	0
Bold	1
Italic	2
Underline	4
Shadow	16
Superscript	32
Subscript	64

Note to Version 6.0 Users: The Outline style does not exist in 4D Write 6.5. Pass its value (8), will have no effect.

color is a long integer that represents the color of the text. This number can be used in the WR O SET ATTRIBUTES command to set other text to the same color.

See Also

WR Get font, WR Get text property.

WR O GET MARGINS

version 6.0

WR O GET MARGINS (area; left; indent; right)

Parameter	Type		Description
area	Longint	→	4D Write area
left	Integer	←	Receives left margin in points
indent	Integer	←	Receives indent in points
right	Integer	←	Receives right margin in points

Note 6.5: This command was only maintained for compatibility purposes. We recommend using the WR Get text property command.

Description

The command WR O GET MARGINS returns in the left, indent, and right parameters the margin settings of the currently selected paragraph in area. If more than one paragraph is selected, this command returns values for the paragraph where the selection begins.

See Also

WR Get text property.

WR O Get pack options (selector) → Longint

Parameter	Type		Description
selector	Integer	→	Options for templates, menus, or rulers 1=Template saving location 2=Template loading location 3=Menus 4=Rulers 5=Frame 6=Page View 7=reformat Message Suppression
Function result	Longint	←	State of an option

Note 6.5: This command was only maintained for compatibility purposes. We recommend using the WR GET AREA PROPERTY command for options 1 and 2, otherwise the WR Get doc property command.

Description

The command WR O Get pack options returns the state of an option set using WR O SET PACK OPTIONS.

If selector equals 1, WR O Get pack options returns the location to which your templates are saved. If the value returned equals 1, the templates are saved on the server. If the value returned equals 0, the templates are saved on the client.

If selector equals 2, WR O Get pack options returns the location from which your templates are loaded. If the value returned equals 1, the templates are loaded from the server. If the value returned equals 0, the templates are loaded from the client.

If selector equals 3, WR O Get pack options returns whether menus are displayed or hidden. If the value returned equals 1, menus are displayed. If the value returned equals 0, menus are hidden.

If selector equals 4, WR O Get pack options returns whether rulers are displayed or hidden. If the value returned equals 1, rulers are displayed. If the value returned equals 0, rulers are hidden.

If selector equals 5, you can specify whether the frame around the 4D Write area should be displayed or hidden. If the value returned equals 1, the frame is displayed. If the value returned equals 0, the frame is hidden. By default, the frame is displayed.

If selector equals 6, you can specify whether the 4D Write area should be displayed using the Page View mode. If the value returned equals 1, the Page View mode is used. If the value returned equals 0, the No Page View mode is used. By default, the Page View mode is used.

If selector equals 7, you can specify whether 4D Write will display a warning message before reformatting a document when the page setup has changed. If the value returned equals 1, the warning is displayed. If the value returned equals 0, no warning is displayed. By default, the warning is displayed.

See Also

WR GET AREA PROPERTY, WR Get doc property.

WR O Get page

version 6.0

WR O Get page (area; position) → Integer

Parameter	Type		Description
area	Longint	→	4D Write area
position	Longint	→	Position of a character in area
Function result	Integer	←	Page of the character position passed in position

Note 6.5: This command was only maintained for compatibility purposes. We recommend using the WR GET CURSOR POSITION command.

Description

The command WR O Get page returns the page of the character position passed in the position parameter. For example, if the 100th character is on page 3, the value in Result in the following line of code will be 3:

```
⇒     Result:=WR O Get page (Area;100)
```

Note: This function is useful only if you set the document to View Page mode.

See Also

WR GET CURSOR POSITION.

WR O GET PICTURE

WR O GET PICTURE (area; height; width; topLeft)

Parameter	Type		Description
area	Longint	→	4D Write area
height	Integer	←	Receives height of the picture
width	Integer	←	Receives width of the picture
topLeft	Integer	←	Receives position of the picture

Note 6.5: This command was only maintained for compatibility purposes. We recommend using the WR GET PICTURE SIZE command.

Description

The command WR O GET PICTURE returns information for a selected picture in the height, width, and topLeft variables.

height is the height of the picture measured in points. width is the width of the picture also measured in points. topLeft is the horizontal coordinate of the top-left corner of the picture. Distance is calculated from the ruler origin and is expressed in points.

See Also

WR GET PICTURE SIZE.

version 6.0

WR O GET PREFERENCES (area; mode; titlePage; units)

Parameter	Type		Description
area	Longint	→	4D Write area
mode	Integer	←	Receives page mode code
titlePage	Integer	←	Receives title page code
units	Integer	←	Receives unit code

Note 6.5: This command was only maintained for compatibility purposes. We recommend using the WR Get doc property command.

Description

The command WR O GET PREFERENCES returns in area the current settings of the document.

The mode, titlePage, and units parameters can have the following values:

Parameter	Value	Description
Mode	0	Document in No Page mode
	1	Document in Page View mode without frame
	2	Document in Page View mode with frame
TitlePage	0	Header and footer on all pages
	1	Header and footer on first page only
	2	Header and footer on all pages except first page
Units	0	Ruler defined in inches
	1	Ruler defined in centimeters
	2	Ruler defined in points

See Also

WR Get doc property.

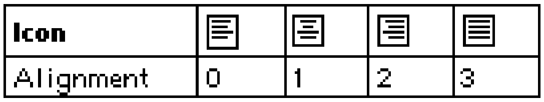
WR O GET RULER (area; leading; alignment)

Parameter	**Type**		**Description**
area	Longint	→	4D Write area
leading	Integer	←	Receives code for spacing between lines
alignment	Integer	←	Receives alignment code

Note 6.5: This command was only maintained for compatibility purposes. We recommend using the WR Get text property command.

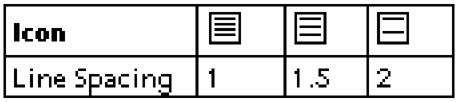
Description

The command WR O GET RULER places into the leading and alignment variables the values you chose for line spacing and alignment for a selection.
alignment can have one of four values.



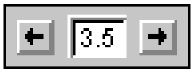
Icon	≡	≡	≡	≡
Alignment	0	1	2	3

By clicking on one of the following icons, you can set the leading to single, one and one-half, or double spacing.



Icon	≡	≡	☰
Line Spacing	1	1.5	2

You can specify leading greater than double line spacing by clicking on the left and right line spacing arrows.



leading is an integer value between 0 and 17 that corresponds to one of the 18 possible line spacing positions. For example, a value of 0 results in leading of a single line while a value of 9 yields leading of five and one-half line spaces.

Value	Line Spacing
0	1.0
1	1.5
2	2.0
3	2.5
4	3.0
5	3.5
6	4.0
7	4.5
8	5.0
9	5.5
10	6.0
11	6.5
12	7.0
13	7.5
14	8.0
15	8.5
16	9.0
17	9.5

See Also

WR Get text property.

WR O Get ScrollBars

WR O Get ScrollBars (area) → Longint

Parameter	Type		Description
area	Longint	→	4D write area
Function result	Longint	←	Scroll bar status

Note 6.5: This command was only maintained for compatibility purposes. We recommend using the WR Get doc property command.

Description

The command WR O Get ScrollBars returns a Long integer that describes the status of the 4D Write document's scroll bars.

- If the function returns 1, the scroll bars are displayed.
- If the function returns 0, the scroll bars are hidden.

See Also

WR Get doc property.

version 6.0

WR O GET STYLESHEET (area; styleNum; name; font; size; style; color)

Parameter	**Type**		**Description**
area	Longint	→	4D Write area
styleNum	Integer	→	Style sheet number
name	String	←	Receives style sheet title
font	Integer	←	Receives font chosen
size	Integer	←	Receives size of chosen font
style	Integer	←	Receives styles used
color	Longint	←	Receives color used

Note 6.5: This command was only maintained for compatibility purposes. We recommend using the WR GET STYLESHEET INFO command.

Description

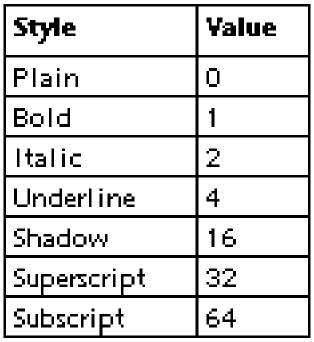
The command WR O GET STYLESHEET returns in the name, font, size, style, and color parameters the values corresponding to the style sheet specified by StyleNum.

name contains the title of the style sheet and can be a maximum of 31 characters.

font returns the number of the font used in the specified style sheet. Use the WR O Font name function to obtain the name of the font with an ID of font.

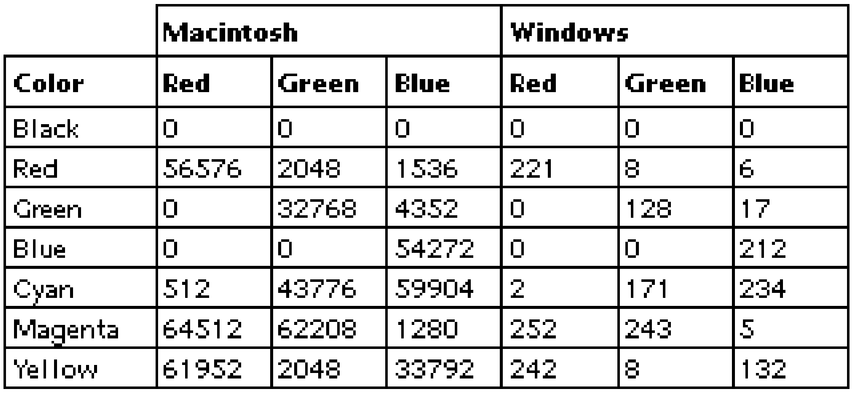
size returns the size of the font used in the specified style sheet.

style returns the sum of the styles used. The list of styles and their corresponding codes is presented in the following table.



Style	**Value**
Plain	0
Bold	1
Italic	2
Underline	4
Shadow	16
Superscript	32
Subscript	64

color returns a long integer that represents the color of the text. Use the WR RGB to color function to obtain the long integer corresponding to a color. The following is a list of commonly used colors and their corresponding values.



	Macintosh			Windows		
Color	Red	Green	Blue	Red	Green	Blue
Black	0	0	0	0	0	0
Red	56576	2048	1536	221	8	6
Green	0	32768	4352	0	128	17
Blue	0	0	54272	0	0	212
Cyan	512	43776	59904	2	171	234
Magenta	64512	62208	1280	252	243	5
Yellow	61952	2048	33792	242	8	132

See Also

WR GET STYLESHEET INFO.

WR O GET TABS

WR O GET TABS (area; tabs)

Parameter	Type		Description
area	Longint	→	4D Write area
tabs	2D Integer array	←	Receives array of tabs

Note 6.5: This command was only maintained for compatibility purposes. We recommend using the WR GET TAB command.

Description

The command WR O GET TABS returns in tabs the tab stops of the selected paragraph in area. If more than one paragraph is selected, WR O GET TABS returns values for the paragraph where the selection begins.

tabs must be a 3 X 10, two-dimensional integer array. WR O GET TABS will not resize tabs. After you issue WR O GET TABS, the three arrays in tabs will contain the following:
• Location of the tab stops in points
• Types of justification
• Leader (fill) characters

The number of tab stops will be returned in the zero element of tabs{1}.

tabs{1} contains the distance in points from the left side of the page to each tab.

tabs{2} contains a number that describes the justification of each tab. Possible values for elements in tabs{2} are as follows.

Justification	Value
Left	0
Right	1
Decimal	2
Centered	3

tabs{3} contains a number that describes the leader character of each tab. Possible values for elements in tabs{3} are as follows.

Leader	Value
None	0
Dots	1
Dashes	2
Underlines	3

Note: The array used in WR O GET TABS must be a 3 X 10, two-dimensional integer array and cannot be resized by the command. You can check the number of tabs in the paragraph by testing the element tabs{1}{0}.

See Also

WR GET TAB.

WR O INSERT HYPHEN (area)

Parameter	Type		Description
area	Longint	→	4D Write area

Note 6.5: This command was only maintained for compatibility purposes. We recommend using the WR EXECUTE COMMAND command.

Description

The command WR O INSERT HYPHEN enables you to insert a hyphen at the insertion point. The insertion point can be located before or after the hyphen. The inserted hyphen is then visible only if the word is at the end of a line.

If a hyphen or punctuation (separator) character is already present at the insertion point, WR O INSERT HYPHEN does nothing and returns an error.

See Also

WR EXECUTE COMMAND.

WR O INSERT PICTURE (area; picture)

Parameter	Type		Description
area	Longint	→	4D Write area
picture	Picture	→	Picture to insert

Note 6.5: This command was only maintained for compatibility purposes. We recommend using the WR INSERT PICTURE command.

Description

The command WR O INSERT PICTURE inserts a picture at the point of selection.

See Also

WR INSERT PICTURE.

WR O Is Hyphen (area) → Longint

Parameter	Type		Description
area	Longint	→	4D Write area
Function result	Longint	←	1, there is a hyphen at the insertion point 0, there is no hyphen at the insertion point

Note 6.5: This command was only maintained for compatibility purposes. We recommend using the WR SELECT command.

Description

The command WR O Is Hyphen returns a Long integer that signifies the presence or absence of hyphens at the insertion point. The insertion point can be located before ar after the hyphen.

• If the function returns 1, there is a hyphen at the insertion point.
• If the function returns 0, there is no hyphen at the insertion point.

See Also

WR SELECT.

WR O LINE SPACING

version 6.0

WR O LINE SPACING (area; leading)

Parameter	Type		Description
area	Longint	→	4D Write area
leading	Integer	→	Spacing between lines [0…17]

Note 6.5: This command was only maintained for compatibility purposes. We recommend using the WR SET TEXT PROPERTY command.

Description

The command WR O LINE SPACING sets the line spacing of the selected paragraphs in area to the value described by leading. leading is an integer value between 0 and 17.

See Also

WR SET TEXT PROPERTY.

WR O MENU STATUS

version 6.0

WR O MENU STATUS (area; command; checked; status{; name})

Parameter	Type		Description
area	Longint	→	4D Write area
command	Integer	→	Command number
checked	Integer	←	Receives checked status 0 = Not checked 1 = Checked with "⇒" 2 = Checked with "-"
status	Integer	←	Receives active status 0 = Inactive 1 = Active
name	Text or String	←	Receives name of menu item

Note 6.5: This command was only maintained for compatibility purposes. We recommend using the WR GET COMMAND INFO command.

Description

The command WR O MENU STATUS determines if the menu item represented by command is checked or active.

Note: The command numbers used by WR O MENU STATUS are not documented in the present manual. Please refer to the 4D Write 6.0.x documentation or use the WR GET COMMAND INFO command.

After a call to WR O MENU STATUS, the checked and active variables contain values that describe the state of command in area.
• If active equals 1, the menu item is enabled. If active equals 0, the item is disabled. If checked equals 0, the item is not checked.
• If checked equals 0, the item is unchecked. If checked equals 1, the item is checked using the standard ⇒ check mark character. If checked equals 2, the item is checked using the "-" dash character.

The optional name parameter must be a text or string variable and will return the text of the menu item.

See Also

WR GET COMMAND INFO.

version 6.0

WR O MOVE PICTURE (area; topLeft)

Parameter	Type		Description
area	Longint	→	4D Write area
topLeft	Integer	→	New picture position

Note 6.5: This command was only maintained for compatibility purposes. We recommend using the alignment settings.

Description

The command WR O MOVE PICTURE moves a selected picture laterally. The selection should not consist of anything except the picture.

topLeft is the horizontal coordinate of the top-left corner of the picture. Distance is calculated from the ruler origin where this value is expressed in points.

WR O ON MENU

WR O ON MENU (area; method)

Parameter	**Type**		**Description**
area	Longint	→	4D Write area
method	String	→	Name of method

Note 6.5: This command was only maintained for compatibility purposes. We recommend using the WR ON COMMAND command.

Description

The command WR O ON MENU installs method as the method for managing 4D Write menus. If method is an empty string, no method will be called. After a menu method is installed, all menu actions must be managed by method.

If area equals 0, WR O ON MENU is applied to all 4D Write areas until the database is closed. method receives the following local variables when the method executes:

Variable	**Description**
$1	A long integer that represents the 4D Write area where the menu was selected
$2	A long integer that contains the command number for the selected menu item
$3	A long integer that describes the modifier keys depressed at the time the menu item was selected.

Explicitly type the $1, $2, and $3 variables using compiler directives.
The following are the modifier keys used, expressed as the sum of key numbers:

Key	**Value**
Command	256
Shift	512
Caps Lock	1024
Option	2048
Control	4096

Note: The command numbers used by WR O ON MENU are not documented in the present manual. Please refer to the 4D Write 6.0.x documentation or use the WR ON COMMAND command.

See Also

WR ON COMMAND.

WR O OPTIONS (area; emptyRefString; saving; printMode; format; undo)

Parameter	Type		Description
area	Longint	→	4D Write area
emptyRefString	String	→	Replacement string for empty references (unused with 4D Write 6.5)
saving	Integer	→	0=No confirmation 1=Confirm -1=No change
printMode	Integer	→	0=Variable length 1=Fixed size -1=No changes
format	Integer	→	0=4D Write and first page as PICT 1=4D Write -1=No change
undo	Integer	→	1=Disabled 0=Enabled -1=No change

Note 6.5: This command was only maintained for compatibility purposes. We recommend using the WR SET AREA PROPERTY command.

Description

The command WR O OPTIONS sets several functional parameters for area.

Pass an empty string in emptyRefString. This parameter is not used in 4D Write 6.5. If the value of a reference is empty and if the **View>Reference** menu command is unchecked, no replacement string is displayed.

The message "This area has been modified. Do you want to save it?" appears when any command has affected the area since its creation. You can disable the display of this message using the WR O OPTIONS command.

confirm controls whether or not 4D Write will display confirmation dialog boxes when a 4D Write area is closed but not saved. If confirm equals 0, no confirmation dialog box appears. If confirm equals 1, 4D Write displays the confirmation dialog box. If confirm equals -1, the current setting is not changed.

printMode controls how area is treated when a form that contains area is printed. If printMode equals 0, the text in area is truncated to the size of area. This is similar to a fixed frame included form. If printMode equals 1, area expands to print all of the text, even across pages. This is similar to a variable frame included form. If printMode equals -1, the current setting is not changed.
format specifies the format in which the document will be saved. If format equals 0, area is saved in the 4D Write format and its first page is saved in PICT format. If format equals 1, area can be saved only in the 4D Write format. If format equals -1, the current setting is not changed.

undo enables or disables the Undo menu item in the Edit menu. Use this parameter in offscreen areas; it saves memory and increases 4D Write speed. If undo equals 0, the Undo menu item is enabled. If undo equals 1, the Undo menu item is disabled. If undo equals -1, the setting is not changed.

See Also

WR SET AREA PROPERTY.

version 6.0

WR O Page number {(area)} → Integer

Parameter	Type		Description
area	Longint	→	4D Write area
Function result	Integer	←	Number of the page currently being printed

Note 6.5: This command was only maintained for compatibility purposes. We recommend using the WR INSERT PAGE NUMBER command.

Description

The command WR O Page number returns the number of the page that is currently being printed. This function can only be used in the header or footer area of a 4D Write document or in a method called from within the header or footer of a 4D Write document. If WR O Page number is used directly in a 4D Write area, the area name may be omitted.

See Also

WR INSERT PAGE NUMBER.

version 6.0

WR O PICTURE TO AREA (area; picture)

Parameter	Type		Description
area	Longint	→	4D Write area
picture	Picture	→	Picture to open

Note 6.5: This command was only maintained for compatibility purposes. We recommend using the WR PICTURE TO AREA command.

Description

The command WR O PICTURE TO AREA assigns to area the 4D Write document in picture. Use this command to manually load Picture from a field or to place a 4D Write document in an offscreen area.

See Also

WR PICTURE TO AREA.

WR O Picture to offscreen area

WR O Picture to offscreen area (picture) → Longint

Parameter	Type		Description
picture	Picture	→	Picture to open
Function result	Longint	←	Area reference number

Note 6.5: This command was only maintained for compatibility purposes. We recommend using the WR PICTURE TO AREA and WR New offscreen area commands.

Description

The command WR O Picture to offscreen area places the document contained in picture into a 4D Write area that is invisible to the user (an offscreen area) and returns a value that can be used to access the new area.
The value returned by WR O Picture to offscreen area can be used in any 4D Write command that requires a 4D Write area. This command is similar to combining WR New offscreen area and WR PICTURE TO AREA.

Note: You should always use WR DELETE OFFSCREEN AREA when you have completed your operation on the offscreen area.

See Also

WR New offscreen area, WR PICTURE TO AREA.

WR O REMOVE HYPHEN

WR O REMOVE HYPHEN (area)

Parameter	Type		Description
area	Longint	→	4D Write area

Note 6.5: This command was only maintained for compatibility purposes. We recommend using the WR SELECT and WR DELETE SELECTION commands.

Description

The command WR O REMOVE HYPHEN enables you to remove the hyphen at the insertion point. The insertion point can be located before or after the hyphen.

If a hyphen is not present at the insertion point, WR O REMOVE HYPHEN does nothing and returns an error.

See Also

WR DELETE SELECTION, WR SELECT.

WR O Replace

version 6.0

WR O Replace (area; old; new; where; method; mode) → Longint

Parameter	Type		Description
area	Longint	→	4D Write area
old	Text	→	String to search for
new	Text	→	String to replace it with
where	Integer	→	0=Partial word 1=Whole word
method	Integer	→	0=Not case sensitive 1=Case sensitive
mode	Integer	→	0=Replace next 1=Replace all
Function result	Longint	←	Number of replacements

Note 6.5: This command was only maintained for compatibility purposes. We recommend using the WR Replace command.

Description

The command WR O Replace searches for occurrences of old in area and replaces them with new. This function also returns the number of replacements. WR O Replace begins replacing from the position of the last character in the selected text.

Where determines whether old can match only whole words. If where equals 0, WR O Replace replaces old if old is part of a word. If where equals 1, WR O Replace replaces old only if old is a whole word. A whole word is a word surrounded by separator characters such as spaces or punctuation marks.

method determines whether or not the replacement will be case sensitive. If method equals 0, WR O Replace replaces old regardless of case. If method equals 1, WR O Replace replaces old only if the cases match.

mode determines how many replacements will be made.

If mode equals 0, only the next occurrence of old is replaced. If mode equals 1, all occurrences of old are replaced with new, regardless of the position of the selected text.

See Also

WR Replace.

WR O RESIZE PICTURE (area; newHeight; newWidth)

Parameter	Type		Description
area	Longint	→	4D Write area
newHeight	Integer	→	New height of the picture
newWidth	Integer	→	New width of the picture

Note 6.5: This command was only maintained for compatibility purposes. We recommend using the WR SET PICTURE SIZE command.

Description

The command WR O RESIZE PICTURE modifies the size of a selected picture in the 4D Write area. The selection should consist only of the selected picture. If no picture is selected, then error 1034 is generated.

newHeight is the new height of the picture in points. newWidth is the new width of the picture also in points.

There are 72 dots per inch (dpi) on a typical Macintosh monitor. If your monitor has a resolution of 82 dpi or more, use a case statement to determine the number of dots per inch to use.

If newHeight or newWidth are less than 0, the size of the picture is not modified. If newHeight or newWidth are equal to 0, the picture is removed.

See Also

WR SET PICTURE SIZE.

WR O Save to picture

version 6.0

WR O Save to picture (area) → Picture

Parameter	Type		Description
area	Longint	→	4D Write area
Function result	Picture	←	4th Dimension picture

Note 6.5: This command was only maintained for compatibility purposes. We recommend using the WR Area to picture command.

Description

The command WR O Save to picture returns a 4th Dimension picture that contains the document in area. The resulting picture is equivalent to the value that is automatically stored in a picture field by 4D Write. This command is useful when area is saved manually to a field or on disk. You can also use this command to retrieve a 4D Write document from an offscreen area.
The difference between WR O Area to picture and WR O Save to picture is subtle. WR O Save to picture sets the document-saved flag for area to True while WR O Area to picture does not. Consequently, closing the window or form that contains area after a call to WR O Save to picture will not cause 4D Write to prompt you to save the document.

Note: The result of this function must be put in either a 4th Dimension picture field or a variable.

See Also

WR Area to picture.

WR O SET ATTRIBUTES

version 6.0

WR O SET ATTRIBUTES (area; font; size; style; color)

Parameter	Type		Description
area	Longint	→	4D Write area
font	Integer	→	Font number
size	Integer	→	Font size
style	Integer	→	Font style
color	Longint	→	Font color

Note 6.5: This command was only maintained for compatibility purposes. We recommend using the WR SET TEXT PROPERTY command to manage font sizes, styles and colors and the
WR SET FONT command to manage fonts.

Description

The command WR O SET ATTRIBUTES sets the attributes of the selected text in area to the values described by font, size, style, and color. If you want to leave one of the attributes unchanged, pass -1 for that attribute.

font is the ID of the font in your system. This value can be obtained by the WR O Font number function.

size is the size of the font expressed in points.

style is the style of the font expressed as a sum of style numbers. The following is a list of style numbers:

Style	Value
Plain	0
Bold	1
Italic	2
Underline	4
Shadow	16
Superscript	32
Subscript	64

When styles are set for a selection of text, each style is applied separately and the selection is affected by the following conditions:

• A style is in effect for the entire selection, such as bold or italic. WR O SET ATTRIBUTES applies that style number and that style is deselected throughout the selection.

• A style is not in effect anywhere in the selection. WR O SET ATTRIBUTES applies that style number throughout the selection.

• A style is in effect for only part of the selection. WR O SET ATTRIBUTES applies that style throughout the selection.

Note to Version 6.0 Users: The Outline style does not exist in 4D Write 6.5. Passing its value (8) has no effect.

color is a long integer that represents the color of the text. The following is a list of commonly used colors and their values:

	Macintosh			**Windows**		
Color	**Red**	**Green**	**Blue**	**Red**	**Green**	**Blue**
Black	0	0	0	0	0	0
Red	56576	2048	1536	221	8	6
Green	0	32768	4352	0	128	17
Blue	0	0	54272	0	0	212
Cyan	512	43776	59904	2	171	234
Magenta	64512	62208	1280	252	243	5
Yellow	61952	2048	33792	242	8	132

See Also

WR SET FONT, WR SET TEXT PROPERTY.

WR O SET MARGINS (area; left; indent; right)

Parameter	Type		Description
area	Longint	→	4D Write area
left	Integer	→	Left margin in points
indent	Integer	→	Indent in points
right	Integer	→	Right margin in points

Note 6.5: This command was only maintained for compatibility purposes. We recommend using the WR SET TEXT PROPERTY command.

Description

The command WR O SET MARGINS sets the margins of the selected paragraphs in area to the values given by left, indent, and right. The values of left, indent, and right are expressed in points from the left side of the document.

To leave one of the margins unchanged, use -1 for that parameter.

See Also

WR SET TEXT PROPERTY.

WR O SET PACK OPTIONS

WR O SET PACK OPTIONS (selector; value)

Parameter	Type		Description
selector	Integer	→	Options for templates, menus, or rulers 1=Template saving location 2=Template loading location 3=Menus, 4=Rulers, 5=Frame, 6=Page View 7=Reformat Message Suppression
value	Longint	→	For templates: 1=Server, 0=Client For menus and rulers: 1=displayed, 0=hidden

Note 6.5: This command was only maintained for compatibility purposes. We recommend using the WR SET AREA PROPERTY command for options 1 and 2, otherwise the WR SET DOC PROPERTY command.

Description

The command WR O SET PACK OPTIONS is used to set options for templates, menus, or rulers. All options set by WR O SET PACK OPTIONS are temporary. To ensure that the options are always in effect, set the options in the Startup method.

If selector equals 1, you can specify where your templates will be saved. If value equals 1, the templates will be saved on the server. If value equals 0, the templates will be saved on the client. By default, templates are saved on the server (value equals 1).

If selector equals 2, you can specify the location from which your templates will be loaded. If value equals 1, the templates will be loaded from the server. If value equals 0, the templates will be loaded from the client. By default, templates are loaded from the server (value equals 1).

If selector equals 3, you can specify whether the menus should be displayed or hidden. If value equals 1, the menus are displayed.
If value equals 0, the menus are hidden. By default, menus are displayed (value equals 1).

If selector equals 4, you can specify whether the rulers should be displayed or hidden. If value equals 1, the rulers are displayed. If value equals 0, the rulers are hidden. By default, the rulers are displayed (value equals 1).

If selector equals 5, you can specify whether the frame around the 4D Write area should be displayed or hidden. If value equals 1, the frame is displayed. If value equals 0, the frame is hidden. By default, the frame is displayed (value equals 1).

If selector equals 6, you can specify whether the 4D Write area is displayed using the Page View mode. If value equals 1, the Page View mode is used. If value equals 0, the No Page View mode is used. By default, the Page View mode is used (value equals 1).

If selector equals 7, you can specify whether 4D Write displays a warning message before reformatting a document when the page setup has changed. If value equals 1, the warning is displayed. If value equals 0, no warning is displayed. By default, the warning is displayed (value equals 1).

WR O SET PACK OPTIONS is especially useful for hiding menus or rulers in an external window. By using the command before creating an external window with the Open external window function, you can hide the menus or rulers before the window is displayed. If you use the WR O DISPLAY RULER or WR O DISPLAY MENUBAR commands instead, 4th Dimension hides the menus or rulers after the window is displayed, causing an awkward redrawing of the window.

See Also

WR SET AREA PROPERTY, WR SET DOC PROPERTY.

version 6.0

WR O SET PREFERENCES (area; mode; firstPage; units)

Parameter	Type		Description
area	Longint	→	4D Write area
mode	Integer	→	0=No page 1=Page view 2=Page view with frame -1=No changes
firstPage	Integer	→	0=Normal 1=Header and footer on all but first page 2=Header and footer on first page only -1=No changes
units	Integer	→	0=Inches 1=Centimeters 2=Points -1=No changes

Note 6.5: This command was only maintained for compatibility purposes. We recommend using the WR SET DOC PROPERTY command.

Description

The command WR O SET PREFERENCES defines the options you can use to display the document contained in area.

To leave an attribute unchanged for any one of the parameters, pass -1 for that attribute.

See Also

WR SET DOC PROPERTY.

WR O SET STYLESHEET (area; styleNum; name; font; size; style; color)

Parameter	Type		Description
area	Longint	→	4D Write area
styleNum	Integer	→	Style sheet number
name	String	→	Style sheet title
font	Integer	→	Font chosen
size	Integer	→	Size of chosen font, 0 = No change
style	Integer	→	Style sheet used, 0 = No change
color	Longint	→	Color used

Note 6.5: This command was only maintained for compatibility purposes. We recommend using the WR SET STYLESHEET INFO command.

Description

The command WR O SET STYLESHEET modifies the parameters name, font, size, style, and color of the style sheet specified by styleNum.

name is the title of the style sheet and has a maximum of 31 characters.

font is the number of the font used in the specified style sheet. Use the WR Font number function to obtain the number of the font in question.

size is the point size of the font used in the specified style sheet. The value must be between 1 and 127. If you specify a font size large than 127, it will be ignored.

style is expressed as a sum of the styles used. The following is a list of styles and their corresponding codes.

Style	Value
Plain	0
Bold	1
Italic	2
Underline	4
Shadow	16
Superscript	32
Subscript	64

To leave a font attribute unchanged, pass -1 for that attribute.

Note to Version 6.0 Users: The Outline style does not exist in 4D Write 6.5. If you pass its value (8), nothing will happen.

color is a long integer that represents the color of the text. Use the WR RGB to color function to obtain the long integer corresponding to a color. See the WR O GET STYLESHEET command for a list of commonly used colors and their values.

See Also

WR SET STYLESHEET INFO.

WR O SET TABS (area; old; new; leader; justification)

Parameter	Type		Description
area	Longint	→	4D Write area
old	Integer	→	Old tab in points
new	Integer	→	New tab in points
leader	Integer	→	Leader character [0…3]
justification	Integer	→	Justification [0…3]

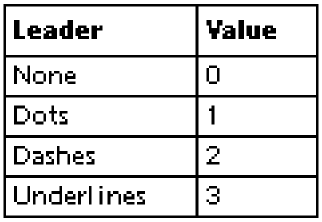
Note 6.5: This command was only maintained for compatibility purposes. We recommend using the WR SET TAB command.

Description

The command WR O SET TABS adds, modifies, or deletes tab stops for the selected paragraphs in area. This command moves the tab stop described by old to the tab stop described by new.

If old equals -1, or if there is no tab stop at old, WR O SET TABS creates a new tab at the position specified by new. If new equals -1, WR O SET TABS deletes the tab at the position specified by old. If a tab is created or moved, it is modified by the leader and justification parameters.

leader specifies the fill character to use for the tab. The following are the possible values for leader:



Leader	Value
None	0
Dots	1
Dashes	2
Underlines	3

justification specifies the type of justification the tab will have. The following are the possible values for justification:

Justification	Value
Left	0
Right	1
Decimal	2
Centered	3

See Also

WR SET TAB.

WR O STATISTICS

WR Obsolete Commands

version 6.0

WR O STATISTICS (area; characters; paragraphs; objects; hotlinks; modified; pages)

Parameter	Type		Description
area	Longint	→	4D Write area
characters	Longint	←	Receives character count
paragraphs	Integer	←	Receives paragraph count
objects	Integer	←	Receives object count
hotlinks	Integer	←	Receives hot links count
modified	Integer	←	Receives modified status 0=Document not modified 1=Document modified
pages	Integer	←	Receives number of pages

Note 6.5: This command was only maintained for compatibility purposes. We recommend using the WR Count command.

Description

The command WR O STATISTICS returns the number of characters, paragraphs, and hot links in area along with 4th Dimension objects. This command also returns the number of pages in area.
After WR O STATISTICS is called, each parameter will contain a value as described in the following table:

Parameter	**Value Returned**
Characters	Total number of characters in *Area*
Paragraphs	Total number of paragraphs in *Area*
Objects	Total number of 4th Dimension objects (fields, expressions, and hot links) in *Area*.
Hotlinks	Total number of hot links subscribed to in *Area*.
Modified	0 = Document not modified 1 = Document modified
Pages	Total number of pages in the document

See Also

WR Count.

WR O STRUCTURE ACCESS

version 6.0

WR O STRUCTURE ACCESS (area; mode)

Parameter	Type		Description
area	Longint	→	4D Write area
mode	Integer	→	0=Allow access 1=Restrict access

Note 6.5: This command was only maintained for compatibility purposes. We recommend using the WR LOCK COMMAND command.

Description

The command WR O STRUCTURE ACCESS controls access to 4th Dimension fields.
If mode equals 0, 4D Write enables the 4D Expression... menu item in the Insert menu. If mode equals 1, 4D Write disables the 4D Expression... menu item in the Insert menu and cancels the keyboard shortcut for accessing the table-field pop-up menu. This command enables the Designer to restrict access of specific fields in the database.

To make the table-field pop-up menu available:
• On Windows, click with the right mouse button.
• On Mac OS, hold down the Ctrl key while pressing the mouse button.

See Also

WR LOCK COMMAND.

WR O TEXT ALIGNMENT

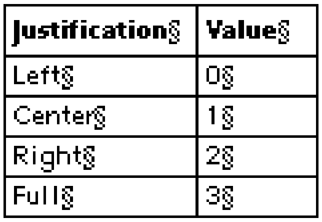
WR O TEXT ALIGNMENT (area; alignment)

Parameter	Type		Description
area	Longint	→	4D Write area
alignment	Integer	→	0=Left 1=Center 2=Right 3=Full

Note 6.5: This command was only maintained for compatibility purposes. We recommend using the WR SET TEXT PROPERTY command.

Description

The command WR O TEXT ALIGNMENT sets the alignment of the selected paragraphs in area to the value described by alignment. The following are the possible values for alignment:



Justification§	Value§
Left§	0§
Center§	1§
Right§	2§
Full§	3§

See Also

WR SET TEXT PROPERTY.

14

Appendixes

version 6.7 (Modified)

Special Keys
In addition to scrolling, 4D Write allows you to use the following key combinations.

Key	**Explanation**
Home	Moves the insertion point to the beginning of the line
End	Moves the insertion point to the end of the line
Ctrl (or Command) + Home	Moves the insertion point to the beginning of the document
Ctrl (or Command) + End	Moves the insertion point to the end of the document
Page Up	Scrolls one page up (does not modify the current selection)
Page Down	Scrolls one page down (does not modify the current selection)
Enter	Inserts a column break or a page break (depending on the current mode)
Ctrl (or Command) + left arrow	Moves the insertion point to the beginning of the current word or to the beginning of the previous word if the insertion point was already at the beginning of the current word.
Ctrl (or Command) + right arrow	Moves the insertion point to the end of the current word or to the end of the following word if the insertion point was already at the end of the current word
Ctrl (or Command) + up arrow	Moves the insertion point to the beginning of the current paragraph
Ctrl (or Command) + down arrow	Moves the insertion point to the end of the current paragraph
Ctrl (or Command) + Delete	Deletes the next word or the letters located on the right of the cursor.
Ctrl (or Command) + Backspace	Deletes the next word or the letters located on the left of the cursor
Shift (in combination with any of the above keys to move the insertion point or view)	Extends or reduces the current selection

Click Combinations
4D Write allows you to use the following mouse click combinations:

Combination	Explanation
Single click	Moves the insertion point, deselecting any text that was selected
Double-click	Selects the word that was double-clicked and the following space (if any)
Triple-click	Selects the paragraph
Click in left margin	Selects the line next to the click
Double-click in left margin	Selects the paragraphs next to the click
Shift+Click	Extends the current selection to the location of the click
Ctrl+Click (Command +Click on MacOS)	Selects text under a picture pasted in a page
Right-Click (Windows) Control+Click (Mac)	Displays a pop-up menu allowing you to insert a field at the insertion point

version 6.7 (Modified)

The following table lists the command value for each menu item. These numbers will remain the same, even if menu items are modified or moved in future versions of 4D Write. For more information, refer to the description of the WR EXECUTE COMMAND command. The following codes can also be used by the WR ON COMMAND, WR LOCK COMMAND and WR GET COMMAND INFO commands.

When using these commands you can either pass the menu item number or the constant. Constants are also listed in Appendix D: 4D Write Constants, theme “WR Commands”.

Menu	But.	Command	#	Constant
File	No	(Menu itself)	100	wr cmd file menu
	Yes	New	101	wr cmd new
	Yes	Open	102	wr cmd open
	Yes	Save	103	wr cmd save
	No	Save as...	104	wr cmd save as
	No	Save as Template	110	wr cmd save as template
	No	Preferences...	105	wr cmd preferences
	No	Page SetUp...	106	wr cmd page setup
	Yes	Print Preview	107	wr cmd print preview
	Yes	Print...	108	wr cmd print
	No	Print Merge...	109	wr cmd print merge
	No	Goto Full Window/Return to Form	20	wr cmd goto full windows
Edit	No	(Menu itself)	200	wr cmd edit menu
	Yes	Undo Fonction (vary)	1	wr cmd undo
	Yes	Redo Fonction (vary)	2	wr cmd redo
	Yes	Cut	3	wr cmd cut
	Yes	Copy	4	wr cmd copy
	Yes	Paste	5	wr cmd paste
	No	Clear	6	wr cmd clear
	No	Select All	7	wr cmd select all
	Yes	Find...	208	wr cmd find
	No	Find Next	209	wr cmd find next
	No	Replace...	210	wr cmd replace
	No	Replace next	211	wr cmd replace next
	No	Change Case	220	wr cmd change case submenu
	No	/ lower case	221	wr cmd lower case
	No	/ UPPER CASE	222	wr cmd upper case
	No	/ Title Case	223	wr cmd title case
	No	/ tOGGLE cASE	224	wr cmd toggle case
	No	Show Selection	309	wr cmd show selection
	No	Goto Page...	807	wr cmd goto page

View	No	(Menu itself)	300	wr cmd view menu
	No	Normal	302	wr cmd view normal
	No	Page	303	wr cmd view page
	No	Toolbars	330	wr cmd toolbars submenu
	No	/ View Standard Toolbar	331	wr cmd view standard toolbar
	No	/ View Format Toolbar	332	wr cmd view format toolbar
	No	/ View Style Toolbar	333	wr cmd view style toolbar
	No	/ View Borders Toolbar	334	wr cmd view borders toolbar
	No	View Ruler	311	wr cmd view ruler
	No	View Header	312	wr cmd view header
	No	View Footer	313	wr cmd view footer
	Yes	View References	314	wr cmd view references
	No	View Pictures	315	wr cmd view pictures
	Yes	View Invisibles	316	wr cmd view invisibles
	No	View Frames	317	wr cmd view frames
	No	View Horizontal Scrollbar	318	wr cmd view HScrollbar
	No	View Vertical Scrollbar	319	wr cmd view VScrollbar
	No	View MenuBar	310	wr cmd view menubar
	No	View Status Bar	320	wr cmd status bar
Insert	No	(Menu itself)	400	wr cmd insert menu
	No	Date and Time...	401	wr cmd insert date and time
	Yes	Current Hour	411	wr cmd insert current hour
	Yes	Current Date	412	wr cmd insert current date
	No	Page Number...	402	wr cmd insert page number
	No	Special Character...	409	wr cmd insert special char
	No	Soft Hyphen	404	wr cmd insert soft hyphen
	No	Non Breaking Space	405	wr cmd insert No break space
	No	Column Break	410	wr cmd insert column break
	No	Page Break	406	wr cmd insert page break
	No	HTML Expression...	414	wr cmd insert HTML expression
	No	Hyperlink...	413	wr cmd insert hyperlink
	No	4D Expression...	407	wr cmd insert 4D expression
Style	No	(Menu itself)	500	wr cmd style menu
	No	Plain	501	wr cmd plain
	Yes	Bold	502	wr cmd bold
	Yes	Italic	503	wr cmd italic
	No	Shadow	504	wr cmd shadow
	No	StrikeThrough	505	wr cmd strikethrough
	No	Underline	520	
	No	/ No Underline	521	wr cmd no underline
	No	/ Single Underline	522	wr cmd continuous underline
	No	/ Word Underline	523	wr cmd word underline
	No	/ Double Underline	524	wr cmd double underline
	No	/ Hatched Underline	525	wr cmd hatched unde
	Yes	Button Underline	530	wr cmd underline button
	No	Superscript	506	wr cmd superscript

	No	Subscript	507	wr cmd subscript
	No	Capitals	508	wr cmd capitals
	No	Small Capitals	509	wr cmd small capitals
Colors	No	(Menu itself)	600	wr cmd colors menu
		Text	601	
		/ Black Text	602	wr cmd black text
		/ Red Text	603	wr cmd red text
		/ Orange Text	604	wr cmd orange text
		/ Yellow Text	605	wr cmd yellow text
		/ Green Text	606	wr cmd green text
		/ Blue Text	607	wr cmd blue text
		/ Violet Text	608	wr cmd violet text
		/ White	609	wr cmd white text
		/ LightGrey Text	610	wr cmd light grey text
		/ MediumGrey Text	611	wr cmd medium grey text
		/ DarkGrey Text	612	wr cmd dark grey
		/ Other Text Color...	613	wr cmd other text color
		Back	615	
		/ No Back Color	628	wr cmd no back color
		/ White Back	616	wr cmd white back
		/ LightRed Back	617	wr cmd light red back
		/ LightOrange Back	618	wr cmd light orange back
		/ LightYellow Back	619	wr cmd light yellow back
		/ LightGreen Back	620	wr cmd light green back
		/ LightBlue Back	621	wr cmd light blue back
		/ LightViolet Back	622	wr cmd light violet back
		/ LightGrey Back	623	wr cmd light grey back
		/ MediumGrey Back	624	wr cmd medium grey back
		/ DarkGrey Back	625	wr cmd dark grey back
		/ Black Back	626	wr cmd black back
		/ Other Back Color...	627	wr cmd other back color
		Strikethrough	631	
		/ Automatic Strikethrough Color	632	wr cmd auto striketh color
		/ Black Strikethrough	633	wr cmd black striketh
		/ Red Strikethrough	634	wr cmd red striketh
		/ Orange Strikethrough	635	wr cmd orange striketh
		/ Yellow Strikethrough	636	wr cmd yellow striketh
		/ Green Strikethrough	637	wr cmd green striketh
		/ Blue Strikethrough	638	wr cmd blue striketh
		/ Violet Strikethrough	639	wr cmd violet striketh
		/ White Strikethrough	640	wr cmd white striketh
		/ LightGrey Strikethrough	641	wr cmd light grey striketh
		/ MediumGrey Strikethrough	642	wr cmd medium grey striketh
		/ DarkGrey Strikethrough	643	wr cmd dark grey striketh
		/ Other Strikethrough Color...	644	wr cmd other striketh color
		Underline	645	
		/ Automatic Underline Color	646	wr cmd auto underline color

		/ Black Underline	647	wr cmd black underline
		/ Red Underline	648	wr cmd red underline
		/ Orange Underline	649	wr cmd orange underline
		/ Yellow Underline	650	wr cmd yellow underline
		/ Green Underline	651	wr cmd green underline
		/ Blue Underline	652	wr cmd blue underline
		/ Violet	653	wr cmd violet underline
		/ White Underline	654	wr cmd white underline
		/ LightGrey Underline	655	wr cmd light grey underline
		/ MediumGrey Underline	656	wr cmd medium grey underline
		/ DarkGrey Underline	657	wr cmd dark grey underline
		/ Other Underline Color...	658	wr cmd other underline color
		Shadow	661	
		/ LightGrey Shadow	662	wr cmd light grey shadow
		/ MediumGrey Shadow	663	wr cmd medium grey shadow
		/ DarkGrey Shadow	664	wr cmd dark grey shadow
		/ Other Shadow Color...	665	wr cmd other shadow color
		Paragraph Back	671	
		/ No Back Color	684	wr cmd no border back color
		/ White Paragraph Back	672	wr cmd white border back
		/ LightRed Paragraph Back	673	wr cmd lgt red border back
		/ LightOrange Paragraph Back	674	wr cmd lgt orange border back
		/ LightYellow Paragraph Back	675	wr cmd lgt yellow border back
		/ LightGreen Paragraph Back	676	wr cmd lgt green border back
		/ LightBlue Paragraph Back	677	wr cmd lgt blue border back
		/ LightViolet Paragraph Back	678	wr cmd lgt violet border back
		/ LightGrey Paragraph Back	679	wr cmd lgt grey border back
		/ MediumGrey Paragraph Back	680	wr cmd med grey border back
		/ DarkGrey Paragraph Back	681	wr cmd dark grey border back
		/ Black Paragraph Back	682	wr cmd black border back
		/ Other Paragraph Back Color...	683	wr cmd other border back color
		Border	685	
		/ Black Border	686	wr cmd black border
		/ Red Border	687	wr cmd red border
		/ Orange Border	688	wr cmd orange border
		/ Yellow Border	689	wr cmd yellow border
		/ Green Border	690	wr cmd green border
		/ Blue Border	691	wr cmd blue border
		/ Violet Border	692	wr cmd violet border
		/ White Border	693	wr cmd white border
		/ LightGrey Border	694	wr cmd light grey border
		/ MediumGrey Border	695	wr cmd medium grey border
		/ DarkGrey Border	696	wr cmd dark grey border
		/ Other Border Color...	697	wr cmd other border color
Paragraph	No	(Menu itself)	700	wr cmd paragraph menu
	No	Copy Ruler	701	wr cmd copy ruler
	No	Paste Ruler	702	wr cmd paste ruler

	No	Bullet ->	1020	
	No	/ No Bullet	1021	wr cmd no bullet
	No	/ Black Square	1022	wr cmd black square bullet
	No	/ White Square	1023	wr cmd white square bullet
	No	/ Black Circle	1024	wr cmd black circle bullet
	No	/ White Circle	1025	wr cmd white circle bullet
	No	/ Diamonds	1026	wr cmd diamonds bullet
	No	/ Clubs	1027	wr cmd clubs bullet
	No	/ Other Bullet...	1028	wr cmd other bullet
	Yes	Align Left	711	wr cmd align left
	Yes	Align Center	712	wr cmd align center
	Yes	Align Right	713	wr cmd align right
	Yes	Full Justification	714	wr cmd full justification
	Yes	Single Spaced	721	wr cmd single spaced
	Yes	1.5 Line Spaced	722	wr cmd 1.5 line space
	Yes	Double Spaced	723	wr cmd double spaced
	No	Other Line Spacing	724	wr cmd other line spacing
Format	No	(Menu itself)	750	wr cmd format menu
	No	Character...	751	wr cmd character
	No	Paragraph...	752	wr cmd paragraph
	No	Tabs...	753	wr cmd tabs
	No	Borders...	754	wr cmd borders
	Yes	Left border	1005	wr cmd left border
	Yes	Top border	1006	wr cmd top border
	Yes	Right border	1007	wr cmd right border
	Yes	Bottom border	1008	wr cmd bottom border
	Yes	All borders	1009	wr cmd all borders
	Yes	Borders inside	1010	wr cmd borders inside
	Yes	No borders	1011	wr cmd no borders
	No	Style Sheets...	755	wr cmd stylesheets
	No	Columns...	756	wr cmd columns
Tools	No	(Menu itself)	800	wr cmd tools menu
	No	Table Wizard...	408	wr cmd table wizard
	No	Spelling...	805	wr cmd spellcheck
	No	Language...	806	wr cmd language
	No	Document Information...	801	wr cmd doc information
	No	Document Statistics...	802	wr cmd doc statistics
	No	Compute References Now	803	wr cmd compute references
	No	Freeze References	804	wr cmd freeze references

About menus and submenus
Some of these constants refer to menus (for example, wr cmd view menu) or submenus (for example, wr cmd change case submenu).
These constants can only be used with the WR GET COMMAND and WR LOCK COMMAND commands (WR LOCK COMMAND deactivates or reactivates the totality of the menu or submenu).
When these constans are used with the WR EXECUTE COMMAND or WR ON COMMAND commands, these latter have no effect.

See Also

WR EXECUTE COMMAND, WR GET COMMAND INFO, WR LOCK COMMAND, WR ON COMMAND.

version 6.7 (Modified)

The following is a list of error codes returned by 4D Write. These codes are used by the WR Error number, WR Error text and WR ON ERROR commands.

Code	Text Error
1002	Error while printing.
1003	Invalid left margin parameter (too close to the right margin).
1004	Invalid indent parameter (too close to the right margin).
1005	Invalid right margin parameter (too close to the left margin and/or indent).
1006	Invalid tab parameter.
1007	Invalid array parameter: Array is not a valid type or size, or is not an array at all.
1012	The file has not been saved.
1013	Invalid selection (either start < 0 or end < start).
1015	The file has not been read.
1016	Invalid menu or item reference.
1017	This field does not seem to be a 4D Write field.
1022	Invalid area parameter passed to an external command.
1023	Invalid 4D file reference number.
1024	A 4D text variable or field allows a maximum of 32000 characters.
1028	Invalid position passed to WR Select.
1032	This file does not exist.
1034	There is no picture selected.
1035	Invalid size parameter.
1036	Invalid position parameter.
1038	This style does not exist.
1041	Not enough memory to execute this command.
1044	Invalid event type.
1047	Invalid field reference.
1048	Invalid option number.
1051	This path does not exist.
1054	First parameter is invalid.
1055	Second parameter is invalid.
1056	Third parameter is invalid.
1057	Fourth parameter is invalid.
1060	You cannot insert a subfield.
1065	This picture does not seem to be valid.
1066	You cannot create more than 256 tab stops.
1067	Invalid tab position.
1068	Invalid tab justification.
1069	You cannot insert a Blob.
1072	There is no hyphen to remove.
1073	Invalid expression.
1074	Invalid Blob.
1075	Text property out of range.

1076	Text property value out of range.
1077	Font not in system.
1078	Unknown stylesheet.
1079	Document property out of range.
1080	Document property value out of range.
1081	Selection has changed during printing.
1082	Invalid destination number.
1083	Invalid picture in page number.
1084	Invalid tab number.
1085	Page number format out of range.
1086	Invalid page number.
1087	Invalid column number.
1088	Invalid line number.
1089	Invalid option number.
1090	Invalid statistic number.
1091	Invalid frame reference.
1092	Invalid command number.
1093	Cannot print. Document is already printing.
1094	Reserved StyleSheet.
1095	Cannot Open File.
1096	Cannot open fast saved Word file.
1097	The document was damaged and has been repaired.

See Also

WR Error number, WR Error text, WR ON ERROR.

Appendix D: 4D Write Constants

version 6.7.1 (Modified)

Numerous 4D Write commands accept constants as parameters. When you call these commands in your methods, you can pass either the name of the constant or its value. 4D Write constants appear underlined in the Method Editor.

To insert a constant in your code, type its name in the Method Editor or drag it from the Explorer's Constants page and drop it in your method.

WR Text properties

wr bold	0
wr italic	1
wr shadow	2
wr strikethrough	3
wr underline	4
wr superscript or subscript	5
wr capital case	6
wr font number	7
wr font size	8
wr text color	9
wr text back color	10
wr strikethrough color	11
wr underline color	12
wr shadow color	13
wr links appearance	14
wr stylesheet number	15
wr user property	16
wr justification	32
wr line spacing	33
wr bullet	34
wr left margin	35
wr first indent	36
wr right margin	37
wr border back color	38
wr border line color	39
wr border line style	40
wr left border	41
wr right border	42
wr top border	43
wr bottom border	44
wr border spacing	45
wr tab	64

WR Text properties values

wr single underline	1
wr word underline	2
wr double underline	3
wr hatched underline	4
wr superscript	1
wr subscript	2
wr capitals	1
wr small capitals	2
wr left justified	0
wr centered	1
wr right justified	2
wr full justified	3
wr no links appearance	0
wr unvisited links appearance	1
wr visited links appearance	2
wr black square bullet	110
wr white square bullet	111
wr black circle bullet	108
wr white circle bullet	109
wr diamonds bullet	117
wr clubs bullet	118
wr no bullet	0

WR Standard colors

wr automatic	-1
wr white	16777215
wr light grey	13421772
wr medium grey	10066329
wr dark grey	6710886
wr black	0
wr red	16711680
wr orange	16750848
wr yellow	16770560
wr green	52249
wr blue	3381759
wr violet	13369599
wr light red	16757683
wr light orange	16767398
wr light yellow	16777164
wr light green	11796403
wr light blue	11790079
wr light violet	16761087

WR Document properties

wr first page	0
wr view mode	1
wr view rulers	2
wr view frames	3
wr view headers	4
wr view footers	5
wr view pictures	6
wr view Hscrollbar	7
wr view Vscrollbar	8
wr view statusbar	9
wr view menubar	10
wr view standard palette	11
wr view format palette	12
wr view style palette	13
wr view borders palette	14
wr view invisible chars	15
wr view references	16
wr view column separators	17
wr different on first page	18
wr different left right pages	19
wr widow orphan	20
wr unit	21
wr default tab	22
wr language	23
wr number of columns	24
wr columns spacing	25
wr binding	26
wr opposite pages	27
wr right first page	28
wr text inside margin	29
wr text outside margin	30
wr text left margin	29
wr text right margin	30
wr text top margin	31
wr text bottom margin	32
wr header top margin	33
wr header bottom margin	34
wr footer top margin	35
wr footer bottom margin	36
wr paper width	37
wr paper height	38
wr dead left margin	39
wr dead top margin	40
wr printable width	41
wr printable height	42
wr data size	43

wr undo buffer size	44
wr horizontal splitter	45
wr vertical splitter	46
wr links color	47
wr visited links color	48
wr view frame area	49
wr view first page header	50
wr view first page footer	51
wr first page top margin	52
wr first page bottom margin	53
wr header 1st page top margin	54
wr header 1st page bottom mg	55
wr footer 1st page top margin	56
wr footer 1st page bottom mg	57
wr draft mode	58
wr column width	59

WR Area properties

wr confirm dialog	0
wr save preview	1
wr allow undo	2
wr modified	3
wr fixed print size	4
wr convert dialog	5
wr minimized button title	6
wr window title	7
wr minimum width	8
wr minimum height	9
wr save template on server	10
wr load template on server	11
wr convert by token	12

WR Events

wr on key	0
wr on double click	1
wr on single click	2
wr on triple click	3
wr on activate	5
wr on printing	7
wr on ruler	8
wr on compute references	9
wr on close	10

WR Frames

wr text frame	0
wr right header	1

wr right footer	2
wr left header	3
wr left footer	4
wr first header	5
wr first footer	6

WR Commands

wr cmd undo	1
wr cmd redo	2
wr cmd cut	3
wr cmd copy	4
wr cmd paste	5
wr cmd clear	6
wr cmd select all	7
wr cmd about	10
wr cmd help	11
wr cmd goto full window	20
wr cmd file menu	100
wr cmd new	101
wr cmd open	102
wr cmd save	103
wr cmd save as	104
wr cmd save as template	110
wr cmd preferences	105
wr cmd page setup	106
wr cmd print preview	107
wr cmd print	108
wr cmd print merge	109
wr cmd edit menu	200
wr cmd find	208
wr cmd find next	209
wr cmd replace	210
wr cmd replace next	211
wr cmd replace all	212
wr cmd change case submenu	220
wr cmd lower case	221
wr cmd upper case	222
wr cmd title case	223
wr cmd toggle case	224
wr cmd view menu	300
wr cmd view normal	302
wr cmd view page	303
wr cmd show selection	309
wr cmd view menubar	310
wr cmd view ruler	311
wr cmd view header	312
wr cmd view footer	313

wr cmd view references	314
wr cmd view pictures	315
wr cmd view invisibles	316
wr cmd view frames	317
wr cmd view HScrollbar	318
wr cmd view VScrollbar	319
wr cmd status bar	320
wr cmd toolbars submenu	330
wr cmd view standard toolbar	331
wr cmd view format toolbar	332
wr cmd view style toolbar	333
wr cmd view borders toolbar	334
wr cmd insert menu	400
wr cmd insert date and time	401
wr cmd insert page number	402
wr cmd insert soft hyphen	404
wr cmd insert non break space	405
wr cmd insert page break	406
wr cmd insert 4D expression	407
wr cmd table wizard	408
wr cmd insert special char	409
wr cmd insert column break	410
wr cmd insert current hour	411
wr cmd insert current date	412
wr cmd insert hyperlink	413
wr cmd insert HTML expression	414
wr cmd style menu	500
wr cmd plain	501
wr cmd bold	502
wr cmd italic	503
wr cmd shadow	504
wr cmd strikethrough	505
wr cmd superscript	506
wr cmd subscript	507
wr cmd capitals	508
wr cmd small capitals	509
wr cmd underline button	530
wr cmd no underline	521
wr cmd single underline	522
wr cmd word underline	523
wr cmd double underline	524
wr cmd hatched underline	525
wr cmd colors menu	600
wr cmd black text	602
wr cmd red text	603
wr cmd orange text	604
wr cmd yellow text	605
wr cmd green text	606

wr cmd blue text	607
wr cmd violet text	608
wr cmd white text	609
wr cmd light grey text	610
wr cmd medium grey text	611
wr cmd dark grey text	612
wr cmd other text color	613
wr cmd white back	616
wr cmd light red back	617
wr cmd light orange back	618
wr cmd light yellow back	619
wr cmd light green back	620
wr cmd light blue back	621
wr cmd light violet back	622
wr cmd light grey back	623
wr cmd medium grey back	624
wr cmd dark grey back	625
wr cmd black back	626
wr cmd other back color	627
wr cmd no back color	628
wr cmd auto striketh color	632
wr cmd black striketh	633
wr cmd red striketh	634
wr cmd orange striketh	635
wr cmd yellow striketh	636
wr cmd green striketh	637
wr cmd blue striketh	638
wr cmd violet striketh	639
wr cmd white striketh	640
wr cmd light grey striketh	641
wr cmd medium grey striketh	642
wr cmd dark grey striketh	643
wr cmd other striketh color	644
wr cmd auto underline color	646
wr cmd black underline	647
wr cmd red underline	648
wr cmd orange underline	649
wr cmd yellow underline	650
wr cmd green underline	651
wr cmd blue underline	652
wr cmd violet underline	653
wr cmd white underline	654
wr cmd light grey underline	655
wr cmd medium grey underline	656
wr cmd dark grey underline	657
wr cmd other underline color	658
wr cmd light grey shadow	662
wr cmd medium grey shadow	663

wr cmd dark grey shadow	664
wr cmd other shadow color	665
wr cmd white border back	672
wr cmd lgt red border back	673
wr cmd lgt orange border back	674
wr cmd lgt yellow border back	675
wr cmd lgt green border back	676
wr cmd lgt blue border back	677
wr cmd lgt violet border back	678
wr cmd lgt grey border back	679
wr cmd med grey border back	680
wr cmd dark grey border back	681
wr cmd black border back	682
wr cmd other border back color	683
wr cmd no border back color	684
wr cmd black border	686
wr cmd red border	687
wr cmd orange border	688
wr cmd yellow border	689
wr cmd green border	690
wr cmd blue border	691
wr cmd violet border	692
wr cmd white border	693
wr cmd light grey border	694
wr cmd medium grey border	695
wr cmd dark grey border	696
wr cmd other border color	697
wr cmd paragraph menu	700
wr cmd copy ruler	701
wr cmd paste ruler	702
wr cmd align left	711
wr cmd align center	712
wr cmd align right	713
wr cmd full justification	714
wr cmd single spaced	721
wr cmd 1.5 line space	722
wr cmd double spaced	723
wr cmd other line spacing	724
wr cmd format menu	750
wr cmd character	751
wr cmd paragraph	752
wr cmd tabs	753
wr cmd borders	754
wr cmd stylesheets	755
wr cmd columns	756
wr cmd tools menu	800
wr cmd doc information	801
wr cmd doc statistics	802

wr cmd compute references	803
wr cmd freeze references	804
wr cmd spellcheck	805
wr cmd language	806
wr cmd goto page	807
wr cmd stylesheet dropdown	1000
wr cmd size dropdown	1001
wr cmd font dropdown	1002
wr cmd left border	1005
wr cmd top border	1006
wr cmd right border	1007
wr cmd bottom border	1008
wr cmd all borders	1009
wr cmd borders inside	1010
wr cmd no borders	1011
wr cmd standard bullet	1012
wr cmd left tab	1031
wr cmd centered tab	1032
wr cmd right tab	1033
wr cmd decimal tab	1034
wr cmd vertical separator	1035
wr cmd no bullet	1021
wr cmd black square bullet	1022
wr cmd white square bullet	1023
wr cmd black circle bullet	1024
wr cmd white circle bullet	1025
wr cmd diamonds bullet	1026
wr cmd clubs bullet	1027
wr cmd other bullet	1028

WR Tabs

wr left tab	1
wr centered tab	2
wr right tab	3
wr decimal tab	4
wr vertical separator tab	5

WR Count

wr nb characters	0
wr nb words	1
wr nb paragraphs	2
wr nb pictures in text flow	3
wr nb objects	4
wr nb soft hyphens	5
wr nb page breaks	6
wr nb column breaks	7

wr nb insertions date time	8
wr nb insertions page number	9
wr nb lines	10
wr nb pages	11
wr nb stylesheets	12
wr nb pictures in page	13
wr nb hyperlinks	14
wr nb RTF expressions	15
wr nb HTML expressions	16

Appendix E: Removed V6.0.x Commands

The following 4D Write version 6.0.x commands have been removed from the version 6.5. These commands will appear prefixed with the letter "R" in version 6.5 methods and will have no effect.

WR R Append break
WR R Append document
WR R Close document
WR R Create document
WR R EXPORT TRANSLATORS
WR R IMPORT TRANSLATORS
WR R INSTALL DEBUG WINDOW
WR R ModuleInfo
WR R REMOVE DEBUG WINDOW
WR R SET GLOBAL OPTIONS
WR R SUBSCRIBE

Constants

WR Area properties

Related command(s): WR GET AREA PROPERTY, WR SET AREA PROPERTY.

Constant	Type	Value
wr allow undo	Long Integer	2
wr confirm dialog	Long Integer	0
wr convert by token	Long Integer	12
wr convert dialog	Long Integer	5
wr fixed print size	Long Integer	4
wr load template on server	Long Integer	11
wr minimized button title	Long Integer	6
wr minimum height	Long Integer	9
wr minimum width	Long Integer	8
wr modified	Long Integer	3
wr save preview	Long Integer	1
wr save template on server	Long Integer	10
wr window title	Long Integer	7

WR Commands

Related command(s): WR EXECUTE COMMAND, WR GET COMMAND INFO, WR LOCK COMMAND, WR ON COMMAND.

Constant	**Type**	**Value**
wr cmd 1.5 line space	Long Integer	722
wr cmd about	Long Integer	10
wr cmd align center	Long Integer	712
wr cmd align left	Long Integer	711
wr cmd align right	Long Integer	713
wr cmd all borders	Long Integer	1009
wr cmd auto striketh color	Long Integer	632
wr cmd auto underline color	Long Integer	646
wr cmd black back	Long Integer	626
wr cmd black border	Long Integer	686
wr cmd black border back	Long Integer	682
wr cmd black circle bullet	Long Integer	1024
wr cmd black square bullet	Long Integer	1022
wr cmd black striketh	Long Integer	633
wr cmd black text	Long Integer	602
wr cmd black underline	Long Integer	647
wr cmd blue border	Long Integer	691
wr cmd blue striketh	Long Integer	638
wr cmd blue text	Long Integer	607
wr cmd blue underline	Long Integer	652
wr cmd bold	Long Integer	502
wr cmd borders	Long Integer	754
wr cmd borders inside	Long Integer	1010
wr cmd bottom border	Long Integer	1008
wr cmd capitals	Long Integer	508
wr cmd centered tab	Long Integer	1032
wr cmd change case submenu	Long Integer	220
wr cmd character	Long Integer	751
wr cmd clear	Long Integer	6
wr cmd clubs bullet	Long Integer	1027
wr cmd colors menu	Long Integer	600
wr cmd columns	Long Integer	756
wr cmd compute references	Long Integer	803
wr cmd copy	Long Integer	4
wr cmd copy ruler	Long Integer	701
wr cmd cut	Long Integer	3

WR Commands (continued)

Constant	Type	Value
wr cmd dark grey back	Long Integer	625
wr cmd dark grey border	Long Integer	696
wr cmd dark grey border back	Long Integer	681
wr cmd dark grey shadow	Long Integer	664
wr cmd dark grey striketh	Long Integer	643
wr cmd dark grey text	Long Integer	612
wr cmd dark grey underline	Long Integer	657
wr cmd decimal tab	Long Integer	1034
wr cmd diamonds bullet	Long Integer	1026
wr cmd doc information	Long Integer	801
wr cmd doc statistics	Long Integer	802
wr cmd double spaced	Long Integer	723
wr cmd double underline	Long Integer	525
wr cmd edit menu	Long Integer	200
wr cmd file menu	Long Integer	100
wr cmd find	Long Integer	208
wr cmd find next	Long Integer	209
wr cmd font dropdown	Long Integer	1002
wr cmd format menu	Long Integer	750
wr cmd freeze references	Long Integer	804
wr cmd full justification	Long Integer	714
wr cmd goto full window	Long Integer	20
wr cmd goto page	Long Integer	807
wr cmd green border	Long Integer	690
wr cmd green striketh	Long Integer	637
wr cmd green text	Long Integer	606
wr cmd green underline	Long Integer	651
wr cmd hatched underline	Long Integer	530
wr cmd help	Long Integer	11
wr cmd insert 4D expression	Long Integer	407
wr cmd insert column break	Long Integer	410
wr cmd insert current date	Long Integer	412
wr cmd insert current hour	Long Integer	411
wr cmd insert date and time	Long Integer	401
wr cmd insert HTML expression	Long Integer	414
wr cmd insert hyperlink	Long Integer	413
wr cmd insert menu	Long Integer	400
wr cmd insert non break space	Long Integer	405
wr cmd insert page break	Long Integer	406

WR Commands (continued)

Constant	Type	Value
wr cmd insert page number	Long Integer	402
wr cmd insert soft hyphen	Long Integer	404
wr cmd insert special char	Long Integer	409
wr cmd italic	Long Integer	503
wr cmd language	Long Integer	806
wr cmd left border	Long Integer	1005
wr cmd left tab	Long Integer	1031
wr cmd lgt blue border back	Long Integer	677
wr cmd lgt green border back	Long Integer	676
wr cmd lgt grey border back	Long Integer	679
wr cmd lgt orange border back	Long Integer	674
wr cmd lgt red border back	Long Integer	673
wr cmd lgt violet border back	Long Integer	678
wr cmd lgt yellow border back	Long Integer	675
wr cmd light blue back	Long Integer	621
wr cmd light green back	Long Integer	620
wr cmd light grey back	Long Integer	623
wr cmd light grey border	Long Integer	694
wr cmd light grey shadow	Long Integer	662
wr cmd light grey striketh	Long Integer	641
wr cmd light grey text	Long Integer	610
wr cmd light grey underline	Long Integer	655
wr cmd light orange back	Long Integer	618
wr cmd light red back	Long Integer	617
wr cmd light violet back	Long Integer	622
wr cmd light yellow back	Long Integer	619
wr cmd lower case	Long Integer	221
wr cmd med grey border back	Long Integer	680
wr cmd medium grey back	Long Integer	624
wr cmd medium grey border	Long Integer	695
wr cmd medium grey shadow	Long Integer	663
wr cmd medium grey striketh	Long Integer	642
wr cmd medium grey text	Long Integer	611
wr cmd medium grey underline	Long Integer	656
wr cmd new	Long Integer	101
wr cmd no back color	Long Integer	628
wr cmd no border back color	Long Integer	684
wr cmd no borders	Long Integer	1011
wr cmd no bullet	Long Integer	1021

WR Commands (continued)

Constant	Type	Value
wr cmd no underline	Long Integer	522
wr cmd open	Long Integer	102
wr cmd orange border	Long Integer	688
wr cmd orange striketh	Long Integer	635
wr cmd orange text	Long Integer	604
wr cmd orange underline	Long Integer	649
wr cmd other back color	Long Integer	627
wr cmd other border back color	Long Integer	683
wr cmd other border color	Long Integer	697
wr cmd other bullet	Long Integer	1028
wr cmd other line spacing	Long Integer	724
wr cmd other shadow color	Long Integer	665
wr cmd other striketh color	Long Integer	644
wr cmd other text color	Long Integer	613
wr cmd other underline color	Long Integer	658
wr cmd page setup	Long Integer	106
wr cmd paragraph	Long Integer	752
wr cmd paragraph menu	Long Integer	700
wr cmd paste	Long Integer	5
wr cmd paste ruler	Long Integer	702
wr cmd plain	Long Integer	501
wr cmd preferences	Long Integer	105
wr cmd print	Long Integer	108
wr cmd print merge	Long Integer	109
wr cmd print preview	Long Integer	107
wr cmd red border	Long Integer	687
wr cmd red striketh	Long Integer	634
wr cmd red text	Long Integer	603
wr cmd red underline	Long Integer	648
wr cmd redo	Long Integer	2
wr cmd replace	Long Integer	210
wr cmd replace all	Long Integer	212
wr cmd replace next	Long Integer	211
wr cmd right border	Long Integer	1007
wr cmd right tab	Long Integer	1033
wr cmd save	Long Integer	103
wr cmd save as	Long Integer	104
wr cmd save as template	Long Integer	110
wr cmd select all	Long Integer	7

WR Commands (continued)

Constant	Type	Value
wr cmd shadow	Long Integer	504
wr cmd show selection	Long Integer	309
wr cmd single spaced	Long Integer	721
wr cmd single underline	Long Integer	523
wr cmd size dropdown	Long Integer	1001
wr cmd small capitals	Long Integer	509
wr cmd spellcheck	Long Integer	805
wr cmd standard bullet	Long Integer	1012
wr cmd status bar	Long Integer	320
wr cmd strikethrough	Long Integer	505
wr cmd style menu	Long Integer	500
wr cmd stylesheet dropdown	Long Integer	1000
wr cmd stylesheets	Long Integer	755
wr cmd subscript	Long Integer	507
wr cmd superscript	Long Integer	506
wr cmd table wizard	Long Integer	408
wr cmd tabs	Long Integer	753
wr cmd title case	Long Integer	223
wr cmd toggle case	Long Integer	224
wr cmd tools menu	Long Integer	800
wr cmd top border	Long Integer	1006
wr cmd underline button	Long Integer	521
wr cmd undo	Long Integer	1
wr cmd upper case	Long Integer	222
wr cmd vertical separator	Long Integer	1035
wr cmd view borders toolbar	Long Integer	334
wr cmd view footer	Long Integer	313
wr cmd view format toolbar	Long Integer	332
wr cmd view frames	Long Integer	317
wr cmd view header	Long Integer	312
wr cmd view HScrollbar	Long Integer	318
wr cmd view invisibles	Long Integer	316
wr cmd view menubar	Long Integer	310
wr cmd view normal	Long Integer	302
wr cmd view page	Long Integer	303
wr cmd view pictures	Long Integer	315
wr cmd view references	Long Integer	314
wr cmd view ruler	Long Integer	311
wr cmd view standard toolbar	Long Integer	331

WR Commands (continued)

Constant	Type	Value
wr cmd view style toolbar	Long Integer	333
wr cmd view VScrollbar	Long Integer	319
wr cmd violet border	Long Integer	692
wr cmd violet striketh	Long Integer	639
wr cmd violet text	Long Integer	608
wr cmd violet underline	Long Integer	653
wr cmd white back	Long Integer	616
wr cmd white border	Long Integer	693
wr cmd white border back	Long Integer	672
wr cmd white circle bullet	Long Integer	1025
wr cmd white square bullet	Long Integer	1023
wr cmd white striketh	Long Integer	640
wr cmd white text	Long Integer	609
wr cmd white underline	Long Integer	654
wr cmd word underline	Long Integer	524
wr cmd yellow border	Long Integer	689
wr cmd yellow striketh	Long Integer	636
wr cmd yellow text	Long Integer	605
wr cmd yellow underline	Long Integer	650
wr toolbars submenu	Long Integer	330

WR Count

Related command(s): WR Count.

Constant	Type	Value
wr nb characters	Long Integer	0
wr nb column breaks	Long Integer	7
wr nb HTML expressions	Long Integer	16
wr nb hyperlinks	Long Integer	14
wr nb insertions date time	Long Integer	8
wr nb insertions page number	Long Integer	9
wr nb lines	Long Integer	10
wr nb objects	Long Integer	4
wr nb page breaks	Long Integer	6
wr nb pages	Long Integer	11
wr nb paragraphs	Long Integer	2
wr nb pictures in text flow	Long Integer	3
wr nb RTF expressions	Long Integer	15
wr nb soft hyphens	Long Integer	5
wr nb stylesheets	Long Integer	12
wr nb words	Long Integer	1

WR Document properties

Related command(s): WR Get doc property, WR SET DOC PROPERTY.

Constant	Type	Value
wr binding	Long Integer	26
wr column width	Long Integer	59
wr columns spacing	Long Integer	25
wr data size	Long Integer	43
wr dead left margin	Long Integer	39
wr dead top margin	Long Integer	40
wr default tab	Long Integer	22
wr different left right pages	Long Integer	19
wr different on first page	Long Integer	18
wr draft mode	Long Integer	58
wr first page	Long Integer	0
wr first page bottom margin	Long Integer	53
wr first page top margin	Long Integer	52
wr footer 1st page bottom mg	Long Integer	57
wr footer 1st page top margin	Long Integer	56
wr footer bottom margin	Long Integer	36
wr footer top margin	Long Integer	35
wr header 1st page bottom mg	Long Integer	55
wr header 1st page top margin	Long Integer	54
wr header bottom margin	Long Integer	34
wr header top margin	Long Integer	33
wr horizontal splitter	Long Integer	45
wr language	Long Integer	23
wr links color	Long Integer	47
wr number of columns	Long Integer	24
wr opposite pages	Long Integer	27
wr paper height	Long Integer	38
wr paper width	Long Integer	37
wr printable height	Long Integer	42
wr printable width	Long Integer	41
wr right first page	Long Integer	28
wr text bottom margin	Long Integer	32
wr text inside margin	Long Integer	29
wr text left margin	Long Integer	29
wr text outside margin	Long Integer	30
wr text right margin	Long Integer	30

WR Document properties (continued)

Constant	Type	Value
wr text top margin	Long Integer	31
wr undo buffer size	Long Integer	44
wr unit	Long Integer	21
wr vertical splitter	Long Integer	46
wr view borders palette	Long Integer	14
wr view column separators	Long Integer	17
wr view first page footer	Long Integer	51
wr view first page header	Long Integer	50
wr view footers	Long Integer	5
wr view format palette	Long Integer	12
wr view frame area	Long Integer	49
wr view frames	Long Integer	3
wr view headers	Long Integer	4
wr view Hscrollbar	Long Integer	7
wr view invisible chars	Long Integer	15
wr view menubar	Long Integer	10
wr view mode	Long Integer	1
wr view pictures	Long Integer	6
wr view references	Long Integer	16
wr view rulers	Long Integer	2
wr view standard palette	Long Integer	11
wr view statusbar	Long Integer	9
wr view style palette	Long Integer	13
wr view Vscrollbar	Long Integer	8
wr visited links color	Long Integer	48
wr widow orphan	Long Integer	20

WR Events

Related command(s): WR Get on event method, WR ON EVENT.

Constant	Type	Value
wr on activate	Long Integer	5
wr on close	Long Integer	10
wr on compute references	Long Integer	9
wr on double click	Long Integer	1
wr on key	Long Integer	0
wr on printing	Long Integer	7
wr on ruler	Long Integer	8
wr on single click	Long Integer	2
wr on triple click	Long Integer	3

WR Frames

Related command(s): WR SET FRAME.

Constant	**Type**	**Value**
wr first footer	Long Integer	6
wr first header	Long Integer	5
wr left footer	Long Integer	4
wr left header	Long Integer	3
wr right footer	Long Integer	2
wr right header	Long Integer	1
wr text frame	Long Integer	0

WR Standard colors

Related command(s): WR COLOR TO RGB, WR SET STYLESHEET TEXT PROP, WR SET TEXT PROPERTY.

Constant	Type	Value
wr automatic	Long Integer	-1
wr black	Long Integer	0
wr blue	Long Integer	3381759
wr dark grey	Long Integer	6710886
wr green	Long Integer	52249
wr light blue	Long Integer	11790079
wr light green	Long Integer	11796403
wr light grey	Long Integer	13421772
wr light orange	Long Integer	16767398
wr light red	Long Integer	16757683
wr light violet	Long Integer	16761087
wr light yellow	Long Integer	16777164
wr medium grey	Long Integer	10066329
wr orange	Long Integer	16750848
wr red	Long Integer	16711680
wr violet	Long Integer	13369599
wr white	Long Integer	16777215
wr yellow	Long Integer	16770560

WR Tabs

Related command(s): WR ADD STYLESHEET TAB, WR ADD TAB, WR SET STYLESHEET TAB, WR SET TAB.

Constant	Type	Value
wr centered tab	Long Integer	2
wr decimal tab	Long Integer	4
wr left tab	Long Integer	1
wr right tab	Long Integer	3
wr vertical separator tab	Long Integer	5

WR Text properties

Related command(s): WR Get stylesheet text prop, WR Get text property, WR SET STYLESHEET TEXT PROP, WR SET TEXT PROPERTY.

Constant	Type	Value
wr bold	Long Integer	0
wr border back color	Long Integer	38
wr border line color	Long Integer	39
wr border line style	Long Integer	40
wr border spacing	Long Integer	45
wr bottom border	Long Integer	44
wr bullet	Long Integer	34
wr capital case	Long Integer	6
wr first indent	Long Integer	36
wr font number	Long Integer	7
wr font size	Long Integer	8
wr italic	Long Integer	1
wr justification	Long Integer	32
wr left border	Long Integer	41
wr left margin	Long Integer	35
wr line spacing	Long Integer	33
wr links appearance	Long Integer	14
wr right border	Long Integer	42
wr right margin	Long Integer	37
wr shadow	Long Integer	2
wr shadow color	Long Integer	13
wr strikethrough	Long Integer	3
wr strikethrough color	Long Integer	11
wr stylesheet number	Long Integer	15
wr superscript or subscript	Long Integer	5
wr tab	Long Integer	64
wr text back color	Long Integer	10
wr text color	Long Integer	9
wr top border	Long Integer	43
wr underline	Long Integer	4
wr underline color	Long Integer	12
wr user property	Long Integer	16

WR Text properties values

Related command(s): WR SET STYLESHEET TEXT PROP, WR SET TEXT PROPERTY.

Constant	Type	Value
wr black circle bullet	Long Integer	108
wr black square bullet	Long Integer	110
wr capitals	Long Integer	1
wr centered	Long Integer	1
wr clubs bullet	Long Integer	118
wr diamonds bullet	Long Integer	117
wr double underline	Long Integer	3
wr full justified	Long Integer	3
wr hatched underline	Long Integer	4
wr left justified	Long Integer	0
wr no bullet	Long Integer	0
wr no links appearance	Long Integer	0
wr right justified	Long Integer	2
wr single underline	Long Integer	1
wr small capitals	Long Integer	2
wr subscript	Long Integer	2
wr superscript	Long Integer	1
wr unvisited links appearance	Long Integer	1
wr visited links appearance	Long Integer	2
wr white circle bullet	Long Integer	109
wr white square bullet	Long Integer	111
wr word underline	Long Integer	2

Command Index