










# 4D Write Pro Reference

-  Presentation
-  Defining a 4D Write Pro area
-  Storing 4D Write Pro documents in 4D Object fields
-  Using a 4D Write Pro area
-  Printing 4D Write Pro documents
-  Filter expressions contained in a 4D Write Pro document
-  Importing 4D Write documents
-  4D Write Pro Language
-  **NEW** What's new

# Presentation

## Overview

---

4D Write Pro offers 4D users an advanced word-processing tool, fully integrated with your 4D database. Using 4D Write Pro, you can write pre-formatted emails and/or letters containing images, a scanned signature, formatted text and placeholders for dynamic variables. You can also create invoices or reports dynamically, including formatted text and images.

The key features of the product are:

- 4D Write compatibility: a 4D Write Pro object can open and convert legacy 4D Write documents while supporting most of their specific properties.
- Word processing: a 4D Write Pro object embedded in a form provides standard word-processing features, including text and style manipulation, image insertion, import and export, and much more.
- Database integration:
  - A 4D Write Pro object can display variable parts which will be filled with data from the database, or data computed by 4D.
  - 4D Write Pro documents can be stored within database fields or on disk.

## Installation and activation

---

4D Write Pro is no longer a plug-in but is fully integrated into 4D itself, making it easier to deploy and manage. No additional installation is required; you can add 4D Write Pro areas to your forms and handle 4D Write Pro variables directly in your 4D applications.

However, note that 4D Write Pro uses the same license as 4D Write. You need to have this license installed in your application in order to enable the feature.

**Requirements:** On Windows, 4D Write Pro features rely on Direct2D. With Windows 7 or Windows Server 2008 machines, make sure the *Platform Update for Windows* has been installed so that the required Direct2D version is available.

## About this manual

---

This manual is the *4D Write Pro Reference Guide*. It covers all 4D Write Pro features, including the user interface and language commands.

Note that 4D Write Pro objects can be handled using specific commands ("**4D Write Pro**" theme) as well as commands from other 4D themes ("**Objects (Forms)**" and "**Styled Text**" themes), documented in the *4D Language Reference* manual.



# 4D Write Pro

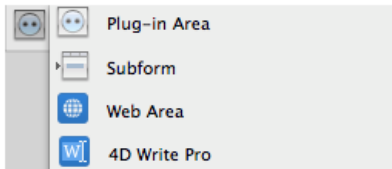
- WP CREATE BOOKMARK New 16.0
- WP DELETE BOOKMARK New 16.0
- WP EXPORT DOCUMENT
- WP EXPORT VARIABLE
- WP GET ATTRIBUTES
- WP Get bookmark range New 16.0
- WP GET BOOKMARKS New 16.0
- WP Get page count New 16.0
- WP Get paragraphs
- WP Get pictures
- WP Get range
- WP Get selection
- WP Import document
- WP INSERT BREAK New 16.0
- WP INSERT DOCUMENT New 16.0
- WP INSERT PICTURE New 16.0
- WP Is font style supported
- WP New Updated 16.0
- WP PRINT Updated 16.0
- WP RESET ATTRIBUTES
- WP SELECT
- WP SET ATTRIBUTES
- WP USE PAGE SETUP

## Defining a 4D Write Pro area

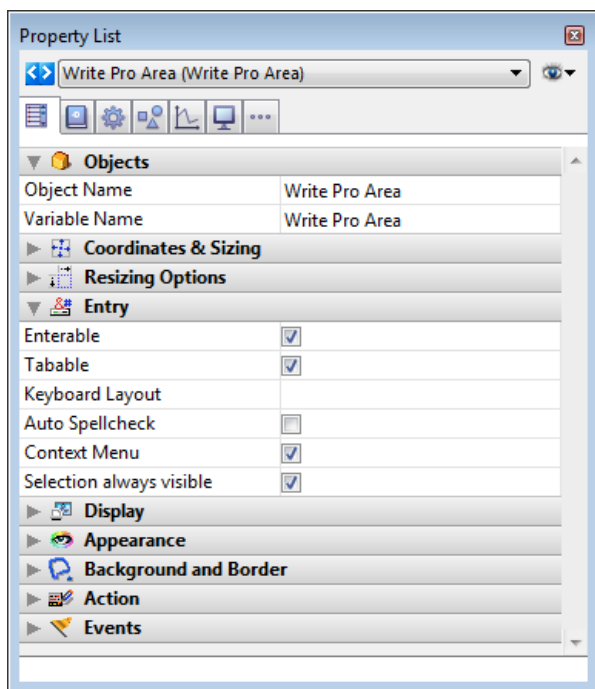
### Creating the area

---

In 4D, 4D Write Pro documents are displayed and edited manually in a 4D form object named **4D Write Pro**. This object is available as part of the last tool (Plug-in Area, Web Area, etc.) found in the object bar:



A 4D Write Pro form area is configured by means of standard properties found in the Property List, such as **Object Name** and **Variable Name**, **Coordinates**, **Entry**, **Display**, **Appearance**, and/or **Events**.



The **Variable Name** property can be used in the language as a reference to the 4D Write Pro area. Note that the variable must be of the *Object* type (for more information, refer to the **C\_OBJECT** command).

"Entry" properties manage basic features for text entry:

- **Enterable**: enables you to lock/unlock the area in order to allow or prevent editing
- **Auto Spellcheck**: available for 4D Write Pro areas
- **Context Menu**: allows you to enable/disable the context menu in Application mode (see the **Using a 4D Write Pro area** section)
- **Selection always visible**: handles text selection as in standard text areas.

### Using the 4D Write Pro Widget of the Object library

You can create a preconfigured 4D Write Pro area using the **4D Write Pro** object found in the Object library ("Entry areas" theme):



This area comes with a control panel for managing all the attributes of the area (font, color, style, etc.):

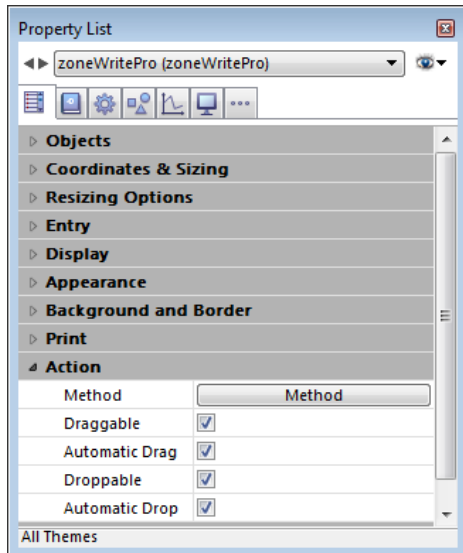


For more information, refer to the **4D Write Pro area** section.

### Configuring Drag and Drop

---

To configure the drag and drop features for your 4D Write Pro areas, you need to select the appropriate options in the "Action" theme of the Property List:



4D Write Pro areas support two drag and drop modes:

- **Custom mode:** only "Draggable" and "Droppable" options checked.  
In this mode, you can select text and start to move it. The object method is then called with the [On Begin Drag Over](#) event, and you can define the drop action using custom code.
- **Automatic mode:** "Draggable", "Droppable", "Automatic Drag" and "Automatic Drop" options checked.  
In this mode, you can automatically move or copy (by pressing the **Alt/Option** key) the selected text. The [On Begin Drag Over](#) event is not triggered.

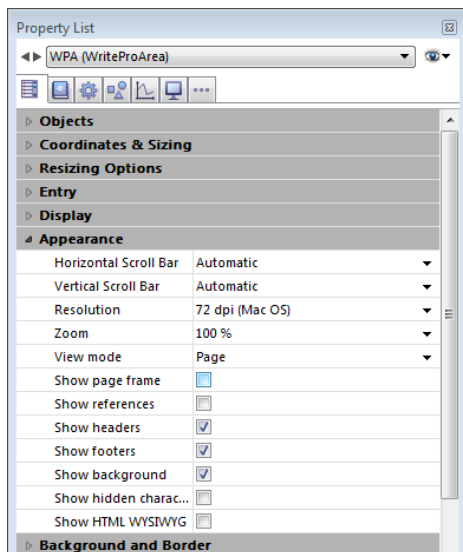
**Note:** Selecting only "Automatic Drag" and "Automatic Drop" options will have no effect in the 4D Write Pro area.

## Configuring View properties

---

Document view properties are directly available in the Property list for 4D Write Pro areas to allow you to define how a 4D Write Pro document will be displayed by default in this area. These properties let you customize, for example, whether 4D Write Pro documents are displayed as they would be printed, or as they would be rendered in a browser. You can set different views of the same 4D Write Pro document in the same form.

Document view settings are handled through specific items in the **Appearance** theme of the Property list for 4D Write Pro form objects:



- **Resolution:** Sets the screen resolution for the 4D Write Pro area contents. By default, it is set to **72 dpi (Mac OS)**, which is the standard resolution for 4D forms on all platforms. Setting this property to **Automatic** means that document rendering will differ between Mac OS and Windows platforms. Setting a specific dpi value will make the document rendering the same on both Mac OS and Windows platforms.

- **Zoom:** Sets the zoom percentage for displaying 4D Write Pro area contents. Default is 100%.
- **View mode:** Sets the mode for displaying the 4D Write Pro document in the form area. Three values are available:
  - **Page** (default): the most complete view mode, which includes page outlines, orientation, margins, page breaks, headers and footers, etc. For more information, please refer to the [Page view features](#) paragraph.
  - **Draft:** draft mode with basic document properties
  - **Embedded:** view mode suitable for embedded areas; it does not display margins, footers, headers, page frames, etc.  
This mode can also be used to produce a Web-like view output (if you also select the 96 dpi resolution and the **Show HTML WYSIWYG** option).

**Note:** The **View mode** property is only used for onscreen rendering. Regarding printing settings, specific rendering rules are automatically used (see [Printing 4D Write Pro documents](#)).

- **Show page frame:** Displays/hides the page frame when Page view mode is set to "Page". Default is hidden.
- **Show references:** Displays all inserted 4D expressions in the document as *references*. When this option is unchecked (default), 4D expressions are displayed as *values*. When you insert a 4D field or expression, 4D Write Pro computes and displays its current value. If you wish to know which field or expression is displayed, check this option. The field or expression references then appear in your document, with a gray background. For example, you have inserted the current date along with a format, the date is displayed:

July 11, 2016

If you check the **Show references** option, the reference is displayed:

String(Current date,Internal date long)

**Note:** 4D expressions can be inserted using the **ST INSERT EXPRESSION** command.

- **Show headers/footers:** Displays/hides the headers and footers when Page view mode is set to "Page" (displayed by default). For more information on headers and footers, please refer to the section.
- **Show background:** Displays/hides both background images and background color (displayed by default).
- **Show hidden characters:** Displays/hides invisible characters (hidden by default).
- **Show HTML WYSIWYG:** Enables/disables the HTML WYSIWYG view, in which any 4D Write Pro advanced attributes which are not compliant with all browsers are removed (disabled by default).

**Compatibility note:** 4D Write Pro documents created with versions up to 4D v15 R5 are displayed using the default values for these properties, with the exception of the Resolution property, which is set to Automatic in this case.

## Storing 4D Write Pro documents in 4D Object fields

You can store your 4D Write Pro documents automatically in the 4D data file. If you created a 4D Write Pro area on a form and created an Object field to store the area's contents, any text entered in the area is saved automatically with each record when the record is validated. You can then use the **QUERY BY ATTRIBUTE** command in order to select records based on the value of their internal attributes. You can also add and query your own attributes with 4D Write Pro areas.

This section describes the following features:

- Binding a 4D Object field to a 4D Write Pro area in a form
- Setting, getting, and querying custom attributes of stored 4D Write Pro documents using the **OB SET**, **OB Get** standard object commands, and **QUERY BY ATTRIBUTE**.

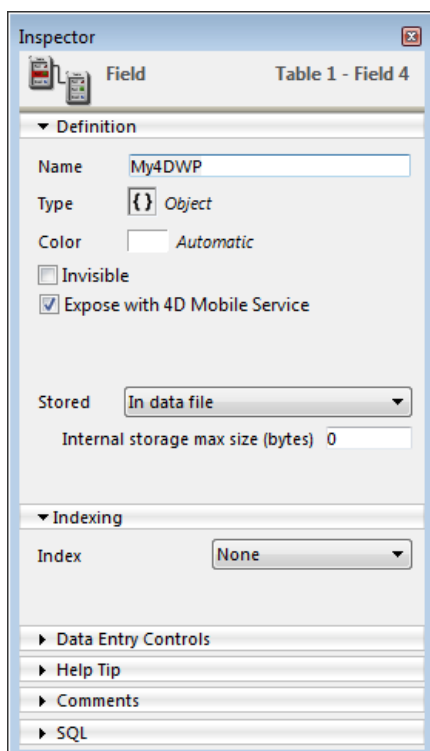
### Assigning a 4D Object field to a 4D Write Pro area

To bind a 4D Write Pro area with a 4D Object field, you just need to reference the field in the Variable Name property of the area.

### Creating the Object field in the Structure

In your database structure, any 4D Object field can be used to store 4D Write Pro documents. As with any Object field, you just have to define its standard properties, according to your needs:

- the field name,
- its attributes, such as "Expose with 4D Mobile Service," as well as its index,
- its storage option (for more on this, see [External data storage](#)).

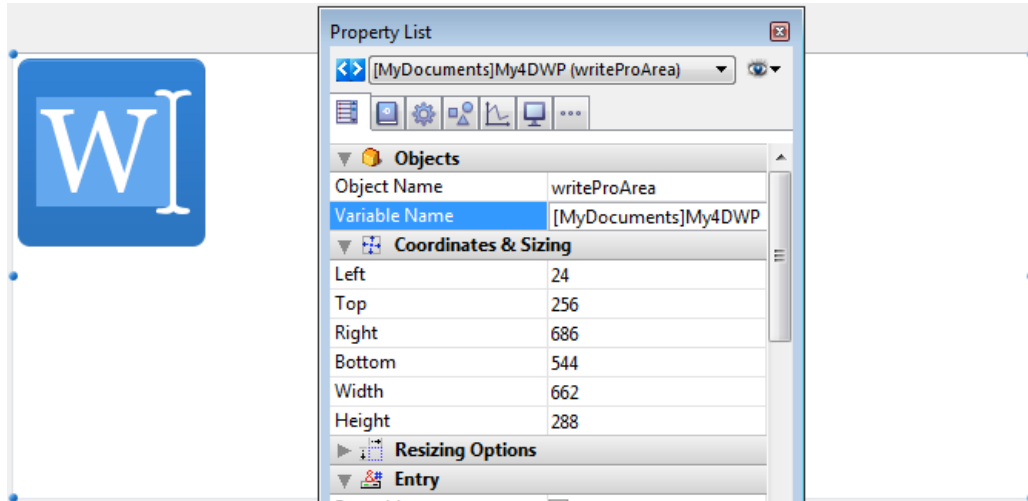


### Assigning the Object field to the 4D Write Pro area

Once you have defined an Object field to store your 4D Write Pro document, you just need to reference it in the

form containing the area. You can use any table or a project form.

In the Form editor, enter the field name using the standard "[Table]Field" notation in the **Variable Name** area of the Property list for the 4D Write Pro area:



Your 4D Write Pro area is then associated with the field, ensuring that its contents will be saved automatically with each record. Note that if you do not use the 4D automatic action buttons, you will have to save the area manually using 4D commands.

## Using custom attributes

When 4D Write Pro areas are stored in Object fields, you can save and read any custom attributes with the 4D Write Pro document, such as, for example, the writer's name, the document category, or any additional information you may find useful. You can then query your custom attributes to select records matching the criteria. Custom attributes will be exported with the **WP EXPORT DOCUMENT** or **WP EXPORT VARIABLE** commands. They will be exported as well when converting a 4D Write Pro Object field to JSON using the **JSON Stringify** command (along with the 4D Write Pro main document attributes).

To set or get custom attributes, you just need to use the standard **OB Get** and **OB SET** commands.

For example, in the form method, you can write:

```
If(Form event=On Validate)
  OB SET([MyDocuments]My4DWP;"myatt_Last edition by";Current user)
  OB SET([MyDocuments]My4DWP;"myatt_Category";"Memo")
End if
```

You can also read custom attributes of the documents:

```
vAttrib:=OB Get([MyDocuments]My4DWP;"myatt_Last edition by")
```

If you have saved custom 4D Write Pro attributes in your data file, you can query these attributes to create a selection of records containing the appropriate attribute value. In the following example, you query the table containing the Object field to select records:

```
QUERY BY ATTRIBUTE ([MyDocuments]; [MyDocuments]My4DWP;"myatt_Category";="Memo")
//selects all records in MyDocuments whose "myatt_Category" custom attribute has the value
"Memo"
//in the My4DWP Object field (bound to a 4D Write Pro area)
```

**Note about custom attribute names:** Since custom attributes share the same naming space as 4D Write Pro internal attributes, we strongly recommend that you use prefixes when defining your own attribute names in order to avoid any conflicts between internal and custom attributes. Non-prefixed names are reserved for 4D Write Pro internal attributes. You can use any custom prefix (for instance, we used "myatt\_" as a prefix in the above example).

**Note:** Starting with 4D v15 R4, 4D Write Pro internal attributes can also be accessed through programming using the standard **QUERY BY ATTRIBUTE**, **OB Get** and **OB SET** commands, but also using **WP SET ATTRIBUTES**, **WP**



**GET ATTRIBUTES** and **WP RESET ATTRIBUTES**. For more information, please refer to the **4D Write Pro Attributes** section.

## QUERY BY ATTRIBUTE

```
QUERY BY ATTRIBUTE ( {aTable}{;}{conjOp ;} objectField ; attributePath ; queryOp ; value {; *} )
```

Parameter	Type	Description
aTable	Table	→ Table for which to return a selection of records, or Default table if omitted
conjOp	Operator	→ Conjunction operator to use to join multiple queries (if any)
objectField	Field	→ Object field to query attributes
attributePath	String	→ Name or path of attribute
queryOp	Operator, String	→ Query operator (comparator)
value	Text, Number, Date, Time	→ Value to compare
*	Operator	→ Continue query flag

### Description

**QUERY BY ATTRIBUTE** looks for records matching the query string defined using the *objectField*, *attributePath*, *queryOp* and *value* parameters, and returns a selection of records for *aTable*.

**Note:** For more information on Object fields (new in 4D v15), please refer to the section of the *Design Reference* manual.

**QUERY BY ATTRIBUTE** changes the current selection of *aTable* for the current process and makes the first record of the new selection the current record. If the *aTable* parameter is omitted, the command applies to the default table. If no default table has been set, an error occurs.

The optional *conjOp* parameter is used to join **QUERY BY ATTRIBUTE** calls when defining multiple queries. The conjunction operators available are the same as the ones for the **QUERY** command:

Conjunction	Symbol to use with QUERY BY ATTRIBUTE
AND	&
OR	
Except	#command_5

The *conjOp* parameter is not used for the first **QUERY BY ATTRIBUTE** call of a multiple query, or if the query is a simple query. If you omit it within a multiple query, the AND (&) operator is used by default.

In *objectField*, pass the Object field whose attribute(s) you want to query. If it belongs to a One table related to *aTable* with an automatic or manual relation, the *objectField* may belong to another table.

**QUERY BY ATTRIBUTE** supports 4D Write Pro custom attributes when documents are stored in Object fields. For more information about this point, please refer to the [Storing 4D Write Pro documents in 4D Object fields](#) section.

In *attributePath*, pass the path of the attribute whose values you want to compare for each record, for example "children.girls.age". If you pass a single name, for example "place", you designate the corresponding attribute found at the first level of the object field.

If an attribute "x" is an array, **QUERY BY ATTRIBUTE** will search records which contain an attribute "x" in which at least one element matches the criteria. To search in array attributes, it is necessary to indicate to the **QUERY BY ATTRIBUTE** command that attribute "x" is an array by appending "[" to its name in *attributePath* (see example 3).

#### Notes:

- Keep in mind that attribute names are case-sensitive: you can have different "MyAtt" and "myAtt" attribute names in the same record.
- Attribute names are trimmed to eliminate extra spaces. For example, " my first attribute .my second attribute " is interpreted as "my first attribute.my second attribute".

The *queryOp* parameter is the comparison operator that is applied between *objectField* and *value*. You can pass one of the symbols shown here:

Comparison	Symbol to use with QUERY BY ATTRIBUTE
Equal to	=
Not equal to	#command_5
Less than	<
Greater than	>
Less than or equal to	<=
Greater than or equal to	>=

**Note:** It is also possible to specify the comparison operator as a text expression instead of a symbol. See the **QUERY** command description for more information.

*value* is the data against which *attributePath* will be compared. The value can be any expression that evaluates to the same data type as *attributePath*. The value is evaluated once, at the beginning of the query. The value is not evaluated for each record. To query for a string contained within a string (a "contains" query), use the wildcard symbol (@) in *value* to isolate the string to be searched for as shown in this example: "@Smith@". Note that in this case, the search only partially benefits from the index (compactness of data storage).

Here is the structure of a query by attribute:

```
QUERY BY ATTRIBUTE ([Table] ; [Table]ObjectField ; "attribute1.attribute2"; =; value)
```

**Note:** An implicit criteria for all operators (except #command\_5) is that the Object field contains an attribute. However, for the #command\_5 operator, it can be undefined (see below).

### Using the #command\_5 operator (support for Null values)

Queries by attribute using the "#command\_5" operator can have different results depending on whether or not the property is checked for the object field:

- **Map NULL values to blank values** property checked (default option, recommended in most cases). In this case, the "#command\_5" operator should be seen as selecting records where "no attribute" of the field contains the value searched for. In this context, 4D considers in a similar manner:
  - fields where the value of the attribute is different from the value searched for,
  - fields where the attribute is not present (or contains a Null value).

For example, the following query returns records for people who have a dog whose name is not Rex, as well as records for people who do not have a dog, or who have a dog with no name:

```
QUERY BY ATTRIBUTE ([People] ; [People]Animals ; "dog.name" ; #command_5 ; "Rex")
```

Another example: this query will return all records for which *[Table]ObjectField* contains an object which contains an *attribute1* attribute which is itself an object containing an *attribute2* attribute whose value is not *value*, as well as records where the object field does not contain *attribute1* or *attribute2*):

```
QUERY BY ATTRIBUTE ([Table] ; [Table]ObjectField ; "attribute1.attribute2" ; #command_5 ; value)
```

This principle also applies to array attributes. For example:

```
QUERY BY ATTRIBUTE ([People] ; [People]OB_Field ; "locations[].city" ; #command_5 ; "paris")
```

This query will return records for people who do not have any address in Paris.

To specifically obtain records where the attribute is undefined, you can use an empty object (see example 2). Note however that searching for NULL values in array elements is not supported.

- **Map NULL values to blank values** property unchecked ("SQL" mode). In this case, undefined attributes (attributes not present in the field or whose value is Null) are not considered as equivalent to blank values by default. As a result, queries of the type "attribute A is different from attribute B" will not return records where attribute A is undefined. To use the same example as above, when the **Map NULL values to blank values** option is not checked for the *[People]Animals* field, the following query will only return records for people who have a dog whose

"name" attribute does not contain "Rex". Records for people who do not have a dog, or who have a dog with no name will not be returned in this case.

```
QUERY BY ATTRIBUTE ([People]; [People]Animals; "dog.name"; #command_5; "Rex")
```

This operation, closer to the SQL logic, is reserved for specific needs.

## Building multiple queries

Here are the rules for building multiple queries by attribute:

- The first query argument must not contain a conjunction.
- Each successive query argument can begin with a conjunction. If you omit it, the AND (&) operator is used by default.
- The first query and every other query, except the last, must use the \* parameter.
- **QUERY BY ATTRIBUTE** can be mixed with **QUERY** commands (see example).
- To perform the query, do not specify the \* parameter in the last **QUERY BY ATTRIBUTE** command. Alternatively, you can execute the **QUERY** command without any parameters other than the table.

**Note:** Each table maintains its own currently-built query. This means that you can create multiple queries simultaneously, one for each table.

No matter which way a query has been defined:

- If the actual query operation is going to take some time to be performed, 4D automatically displays a message containing a progress thermometer. These messages can be turned on and off by using the **MESSAGES ON** and **MESSAGES OFF** commands. If a progress thermometer is displayed, the user can click on the Stop button to interrupt the query. If the query is completed, OK is set to 1. Otherwise, if the query is interrupted, OK is set to 0 (zero).
- If any indexed object fields are specified, the query is optimized every time that it is possible (indexed fields are searched first) resulting in a query that takes the least amount of time possible.

## Date values in the object

Dates are stored in objects according to database settings; by default, the time zone is taken into account (see the [JSON use local time](#) selector in the **SET DATABASE PARAMETER** command).

```
!1973-05-22! -> "1973-05-21T23:00:00.000Z"
```

This setting is also taken into account during queries, so you do not have to worry about it if you always use your database at the same place and if settings are the same on all machines that access the data. In this case, the following query will correctly return records whose Birthday attribute equals !1973-05-22! (saved as "1973-05-21T23:00:00.00Z"):

```
QUERY BY ATTRIBUTE ([Persons]; [Persons]OB_Info; "Birthday"; =; !1973-05-22!)
```

If you do not want to use the GMT settings, you can modify these settings using the following instruction:

```
SET DATABASE PARAMETER (JSON use local time; 0)
```

Keep in mind that the scope of this setting is the process only. If you execute this instruction, then October 1st, 1965 will be stored "1965-10-01T00:00:00.000Z" but you will need to set the same parameter before launching your queries:

```
SET DATABASE PARAMETER (JSON use local time; 0)
QUERY BY ATTRIBUTE ([Persons]; [Persons]OB_Info; "Birthday"; =; !1976-11-27!)
```

## Using the virtual length property

You can use the virtual "length" property with this command. This property is available automatically for all array type attributes and returns the size of the array, i.e. the number of elements it contains. It can be used in the context of executing the **QUERY BY ATTRIBUTE** command (see example 4).

## Example 1

---

In this example, the "age" attribute is either a string or an integer and we want to find people whose age is between 20 and 29. The first two lines query the attribute as an integer ( $\geq 20$  and  $< 30$ ) and the last ones query the field as a string (starts with "2" but is different from "2").

```
QUERY BY ATTRIBUTE ([Persons]; [Persons]OB_Info;"age";>=;20;*)
QUERY BY ATTRIBUTE ([Persons]; & ; [Persons]OB_Info;"age";<;30;*)
QUERY BY ATTRIBUTE ([Persons]; |; [Persons]OB_Info;"age";=;"2@";*)
QUERY BY ATTRIBUTE ([Persons]; & ; [Persons]OB_Info;"age";#command_5;"2") //no final * to launch
execution
```

## Example 2

---

The **QUERY BY ATTRIBUTE** command can be used to find records where certain attributes are defined (or are not defined). To do this, you have to use an empty object.

```
//Find records where e-mail is defined in the object field
C_OBJECT($undefined)
QUERY BY ATTRIBUTE ([Persons]; [Persons]Info;"e-mail";#command_5;$undefined)
```

```
//Find records where zip code is NOT defined in the object field
C_OBJECT($undefined)
QUERY BY ATTRIBUTE ([Persons]; [Persons]Info;"zip code";=;$undefined)
```

**Note:** This specific syntax is not supported with array type attributes. Searching for NULL values in array elements will give invalid results.

## Example 3

---

You want to search a field containing array attributes. With the following two records:

*Record1:*

```
[People]name: "martin"
[People]OB_Field:
  "locations" : [ {
    "kind": "office",
    "city": "paris"
  } ]
```

*Record2:*

```
[People]name: "smith"
[People]OB_Field:
  "locations" : [ {
    "kind": "home",
    "city": "lyon"
  } , {
    "kind": "office",
    "city": "paris"
  } ]
```

... **QUERY BY ATTRIBUTE** will find people with a location in "paris" using this statement:

```
//flag the array attribute with "[]" syntax
QUERY BY ATTRIBUTE ([People]; [People]OB_Field;"locations[].city";=;"paris")
//selects "martin" and "smith"
```

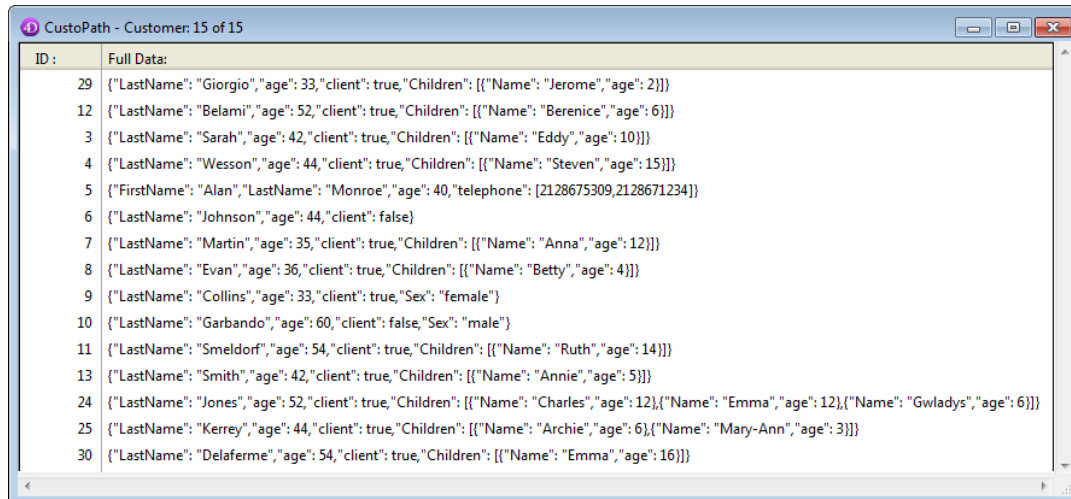
**Note:** If you defined several criteria on the same array attribute, the matched criteria will not necessarily apply to

the same array element. In the following example, the query returns "smith" because it has a "locations" element whose "kind" is "home" and a "locations" element whose "city" is "paris", even if it's not the same element:

```
QUERY BY ATTRIBUTE ([People]; [People]OB_Field;"locations[].kind";="home";*)
QUERY BY ATTRIBUTE ([People]; & ;[People]OB_Field;"locations[].city";="paris")
//selects "smith"
```

## Example 4

This example illustrates the use of the virtual "length" property. Your database has a [Customer]full\_Data object field with the following data:



ID:	Full Data:
29	{"LastName": "Giorgio", "age": 33, "client": true, "Children": [{"Name": "Jerome", "age": 2}]}
12	{"LastName": "Belami", "age": 52, "client": true, "Children": [{"Name": "Berenice", "age": 6}]}
3	{"LastName": "Sarah", "age": 42, "client": true, "Children": [{"Name": "Eddy", "age": 10}]}
4	{"LastName": "Wesson", "age": 44, "client": true, "Children": [{"Name": "Steven", "age": 15}]}
5	{"FirstName": "Alan", "LastName": "Monroe", "age": 40, "telephone": [2128675309, 2128671234]}
6	{"LastName": "Johnson", "age": 44, "client": false}
7	{"LastName": "Martin", "age": 35, "client": true, "Children": [{"Name": "Anna", "age": 12}]}
8	{"LastName": "Evan", "age": 36, "client": true, "Children": [{"Name": "Betty", "age": 4}]}
9	{"LastName": "Collins", "age": 33, "client": true, "Sex": "female"}
10	{"LastName": "Garbando", "age": 60, "client": false, "Sex": "male"}
11	{"LastName": "Smeldorf", "age": 54, "client": true, "Children": [{"Name": "Ruth", "age": 14}]}
13	{"LastName": "Smith", "age": 42, "client": true, "Children": [{"Name": "Annie", "age": 5}]}
24	{"LastName": "Jones", "age": 52, "client": true, "Children": [{"Name": "Charles", "age": 12}, {"Name": "Emma", "age": 12}, {"Name": "Gwladys", "age": 6}]}
25	{"LastName": "Kerrey", "age": 44, "client": true, "Children": [{"Name": "Archie", "age": 6}, {"Name": "Mary-Ann", "age": 3}]}
30	{"LastName": "Delaferme", "age": 54, "client": true, "Children": [{"Name": "Emma", "age": 16}]}

You want to get the records for any customers who have two or more children. To do this, you can write:

```
QUERY BY ATTRIBUTE ([Customer]; [Customer]full_Data;"Children.length";>=;2)
```

## System variables and sets

If the query is carried out correctly, the OK system variable is set to 1.

The OK variable is set to 0 if:

- the user clicks on the **Cancel/Stop** button,
- in 'query and lock' mode (see the **SET QUERY AND LOCK** command), the query has found at least one locked record. In this case as well, the LockedSet system set is updated.

## WP EXPORT DOCUMENT

WP EXPORT DOCUMENT ( wpDoc ; filePath {; format {; options}} )

Parameter	Type		Description
wpDoc	Object	⇒	4D Write Pro variable
filePath	String	⇒	Path of exported file
format	Longint	⇒	Document output format
options	Longint	⇒	Export options

### Description

---

The **WP EXPORT DOCUMENT** command exports the *wpDoc* 4D Write Pro object to a document on disk defined by the *filePath* parameter as well as any optional parameters.

For more information, refer to the description of this command in the [4D Write Pro Language](#) chapter of the *4D Write Pro Reference* manual.

OB Get ( object ; property {; type} ) -> Function result

Parameter	Type	Description
object	Object, Object Field	→ Structured object
property	Text	→ Name of property to read
type	Longint	→ Type to which to convert the value
Function result	Boolean, Date, Object, Pointer, Real, Text	↻ Current value of property

## Description

The **OB Get** command returns the current value of the *property* of the *object*, optionally converted into the *type* specified.

*object* must have been defined using the **C\_OBJECT** command or designate a 4D object field.

**Note:** This command supports attribute definitions in 4D Write Pro *objects*, like the **WP GET ATTRIBUTES** command (see example 9). However, unlike **WP GET ATTRIBUTES**, **OB Get** does not allow you to handle a picture variable or field directly as an attribute value.

In the *property* parameter, pass the label of the property to be read. Note that the *property* parameter is case sensitive.

By default, 4D returns the value of the property in its original type. You can "force" the typing of the value returned using the optional *type* parameter. To do this, in *type* you pass one of the following constants found in the **Field and Variable Types** theme:

Constant	Type	Value
Is Boolean	Longint	6
Is date	Longint	4
Is longint	Longint	9
Is object	Longint	38
Is pointer	Longint	23
Is real	Longint	1
Is text	Longint	2
Is time	Longint	11

The command returns the value of the *property*. Several types of data are supported. Note that:

- a pointer is returned as such; it can be evaluated using the **JSON Stringify** command,
- dates are returned in the format "YYYY-MM-DDTHH:mm:ss.SSSZ"
- in real values, the decimal separator is always a period "."
- times are returned as a number. Note that **OB SET** stores times as milliseconds, in compliance with the JavaScript standard, while 4D expects a number of seconds. In order for **OB Get** to interpret a stored time correctly, you need to use the Is time constant.

## Example 1

Retrieving a text type value:

```
C_OBJECT($ref)
C_TEXT($FirstName)
OB SET($ref;"FirstName";"Harry")
$FirstName:=OB Get($ref;"FirstName") // $FirstName = "Harry" (text)
```

## Example 2



Retrieving a real number value converted into a longint:

```
OB SET($ref ;"age";42)
$age:=OB Get($ref ;"age") // $age is a real number (default)
$age:=OB Get($ref ;"age";Is longint) // $age is a longint
```

### Example 3

---

Retrieving the values of an object:

```
C_OBJECT($ref1;$ref2)
OB SET($ref1;"LastName";"Smith") // $ref1={"LastName":"Smith"}
OB SET($ref2;"son";$ref1) // $ref2={"son":{"LastName":"Smith"}}
$son:=OB Get($ref2;"son") // $son={"LastName":"john"} (object)
$sonsName:=OB Get($son ;"name") // $sonsName="john" (text)
```

### Example 4

---

Modifying the age of an employee twice:

```
C_OBJECT($ref_john;$ref_jim)
OB SET($ref_john;"name";"John";"age";35)
OB SET($ref_jim;"name";"Jim";"age";40)
APPEND TO ARRAY($myArray;$ref_john) // we create an object array
APPEND TO ARRAY($myArray;$ref_jim)
// we change the age for John from 35 to 25
OB SET($myArray{1};"age";25)
// We replace the age of "John" in the array
For($i;1;Size of array($myArray))
  If(OB Get($myArray{$i};"name")="John")
    OB SET($myArray{$i};"age";36) // instead of 25
  // $ref_john={"name":"John","age":36}
  End if
End for
```

### Example 5

---

Deserializing a data string formatted in ISO:

```
C_OBJECT($object)
C_DATE($birthday)
C_TEXT($birthdayString)
OB SET($object;"Birthday";"1990-12-25T12:00:00Z")
$birthdayString:=OB Get($object;"Birthday")
// $birthdayString="1990-12-25T12:00:00Z"
$birthday:=OB Get($object;"Birthday";Is date)
// $birthday=25/12/90
```

### Example 6

---

Using nested objects:

```
C_OBJECT($ref1;$child;$children)
C_TEXT($childName)
OB SET($ref1;"firstname";"John";"lastname";"Monroe")
//{"firstname":"john","lastname":"Monroe"}
OB SET($children;"children";$ref1)
```

```

$child:=OB Get($children;"children")
// $son = {"firstname":"John","lastname":"Monroe"} (object)
$childName:=OB Get($child;"lastname")
// $childName = "Monroe" (text)
// or
$childName:=OB Get(OB Get($children;"children");"lastname")
// $childName = "Monroe" (text)

```

## Example 7

---

Recovery in 4D of a time stored in an object:

```

C_OBJECT($obj_o)
C_TIME($set_h;$get_h)

$set_h:=?01:00:00?+1
OB SET($obj_o;"myHour";$set_h)
// $obj_o == {"myHour":3601000}
// The time is stored in milliseconds
$get_h:=OB Get($obj_o;"myHour";Is_time)
// $get_h == ?01:00:01?
// The time is read correctly

```

## Example 8

---

Examples of working with 4D object fields:

```

// Define a value
OB SET([People]Identity_OB;"First name";$firstName)
OB SET([People]Identity_OB;"Last name";$lastName)

// Get a value
$firstName:=OB Get([People]Identity_OB;"First name")
$lastName:=OB Get([People]Identity_OB;"Last name")

```

## Example 9

---

In the method of a form containing a 4D Write Pro area, you can write:

```

If(Form event=On Validate)
    OB SET([MyDocuments]My4DWP;"myatt_Last edition by";Current user)
    OB SET([MyDocuments]My4DWP;"myatt_Category";"Memo")
End if

```

You can also read custom attributes of the documents:

```

vAttrib:=OB Get([MyDocuments]My4DWP;"myatt_Last edition by")

```

## Using a 4D Write Pro area

### Managing documents in 4D Write Pro areas

---

In 4D applications, 4D Write Pro documents are created, imported, and exported by means of specific commands found in the **4D Write Pro** theme (**WP EXPORT DOCUMENT**, **WP EXPORT VARIABLE**, **WP Import document**, **WP New**).

You can also associate a 4D Write Pro area with an Object field in a database form. This way, each 4D Write Pro document is automatically saved with the record and stored in the database's data (see **Storing 4D Write Pro documents in 4D Object fields**).

### .4wp document format

---

You can save and reopen 4D Write Pro documents to and from disk without any loss using the native **.4wp** format. The **.4wp** format consists of a zip folder whose name is the document title and whose contents are HTML text and images:

- HTML text combines regular HTML with 4D expressions (which are not computed) as well as 4D-specific tags,
- images are stored in a folder with the same name as the document title, next to the HTML file.

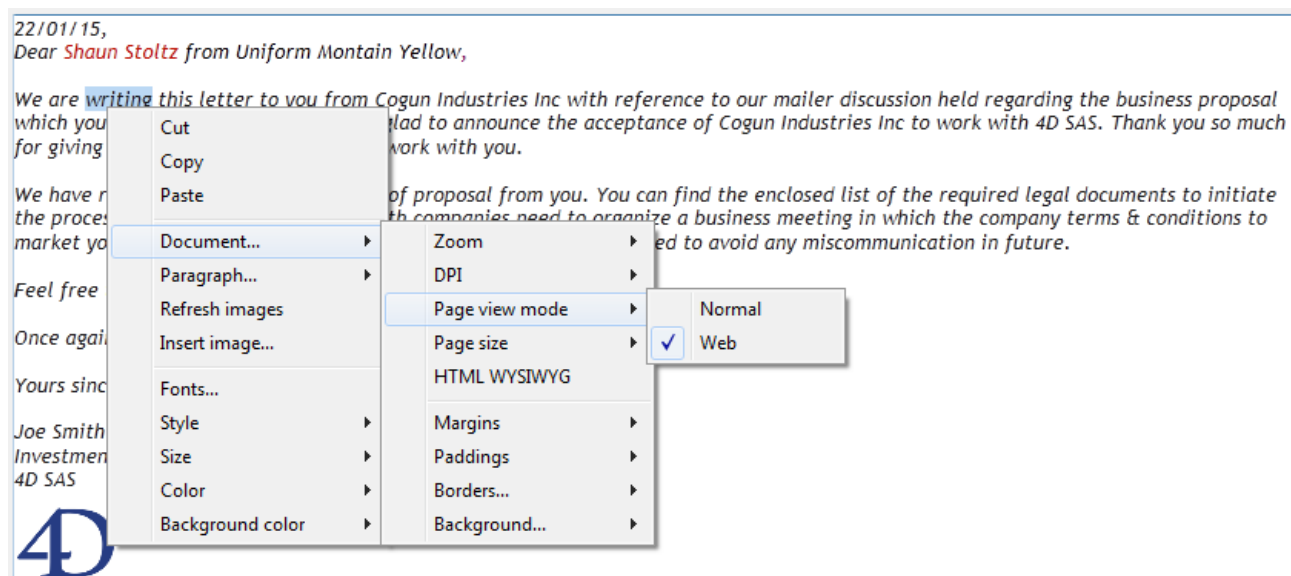
Since **.4wp** documents are based on HTML, they can be imported or opened in any external application supporting HTML.

**Note:** The 4D Write Pro internal document format is a proprietary HTML extension, compatible with HTML5/XHTML5, but which supports its own subset of HTML/CSS attributes and tags, specified in this manual. As a result, only HTML documents exported by 4D Write Pro can be opened by 4D Write Pro without any risk of data loss. Importing HTML documents that were created externally could produce errors.

### User interface

---

If the **Context menu** property is checked for a 4D Write Pro area (see **Defining a 4D Write Pro area**), a comprehensive context menu is available to users in the Application mode:



This menu offers access to all the 4D Write Pro user features.

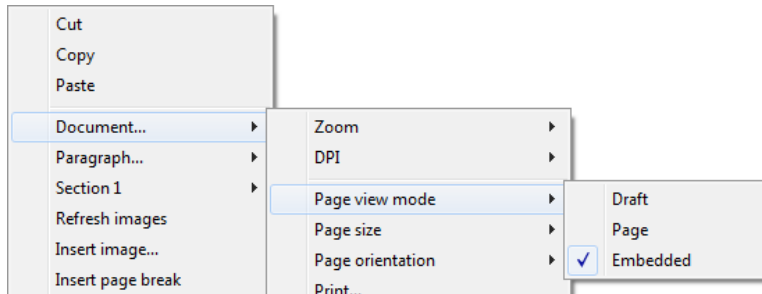
### Selecting the view mode

---

4D Write Pro documents can be displayed in one of three page view modes:

- **Draft:** draft mode with basic properties
- **Page** (default): "print view" mode
- **Embedded:** view mode suitable for embedded areas; it does not display margins, footers, headers, page frames, etc.  
This mode can also be used to produce a Web-like view output (if you also select the 96 dpi resolution and the **HTML WYSIWYG** option).

The page view mode can be configured by means of the area pop-up menu:



**Note:** The page view mode is not stored with the document.

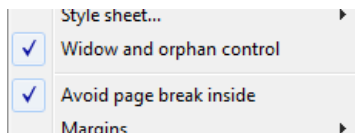
For areas embedded in 4D forms, the view mode can also be set by default using the Property list. In this case, the view mode is stored as a property of the 4D Write Pro form object (for more information, please refer to the [Configuring View properties](#) paragraph).

## Page view features

When the document is in **Page** view mode, the following document properties are displayed for the user:

- Page outlines to represent printing limits
- Page width and Page height (default: 21x29.7 cm)
- Page orientation (default: Portrait)
- Page margin (default: 2.5 cm)

In addition, new paragraph properties are available in the **Paragraph...** submenu:



- **Widow and orphan control:** When this option is checked for a paragraph, 4D Write Pro does not allow widows (last line of a paragraph isolated at the top of a page) or orphans (first line of a paragraph isolated at the bottom of a page) in the document. In the first case, the previous line of the paragraph is added to the top of the page so that two lines are displayed there. In the second case, the single first line is moved onto the next page.
- **Avoid page break inside:** When this option is checked for a paragraph, 4D Write Pro prevents this paragraph from being broken into parts on two or more pages.

You can also use the following commands found in the context menu:

- **Insert page break:** Adds a page break attribute at the cursor location. If any text is selected, it is replaced by the page break.
- **Document.../Page size:** Allows you to select a page size. Various standard page sizes are available.
- **Document.../Page orientation:** Standard orientation (Portrait/Landscape) property.

**Note:** When a document is in **Embedded** or **Draft** view mode, page properties can be set, even if their effect is not visible. In **Draft** view mode, the following paragraph property effects are visible:

- Page height limitation (lines drawn)
- Avoid page break inside property
- Widow and orphan control.

## Handling headers, footers, and sections

---

4D Write Pro documents support headers and footers. These headers and footers are related to sections.

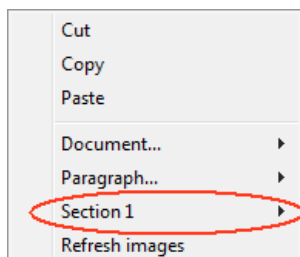
A section is a part of a document which is defined by a page range and can have its own paging and common attributes. A document can contain any number of sections (from just one, up to the total number of pages). Each page can only belong to one section.

You can define a set of headers and footers for each section.

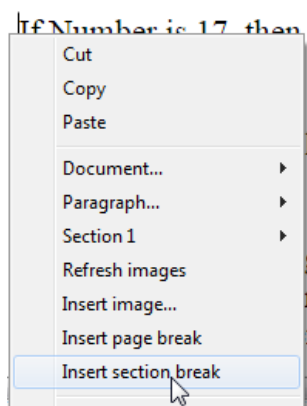
### Defining a section

A section is a subset of continuous pages in a 4D Write Pro document. A document can contain one or more sections. A section can contain any number of pages, from a single page to the total number of pages in the document.

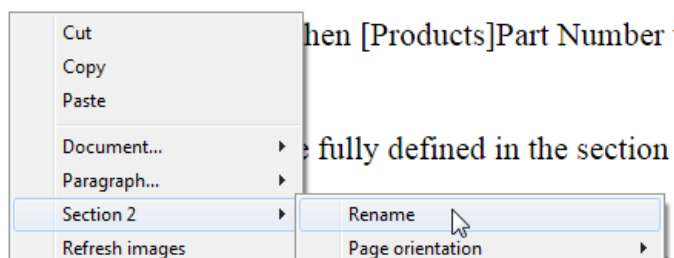
By default, a document contains a single section, named **Section 1**. The 4D Write Pro contextual menu displays this section number wherever you click in the document:



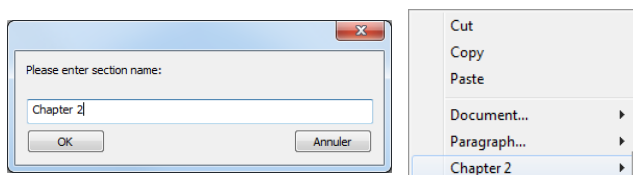
You create a new section by adding a section break in the text flow:



When a section break has been added, the contextual menu displays an incremented number for each section. You can, however, rename any section:



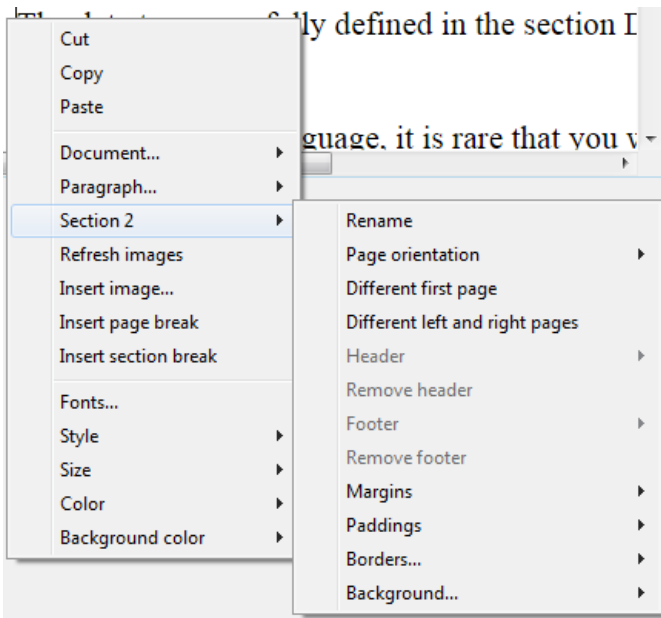
The name you entered is then used as the section name everywhere in the document:



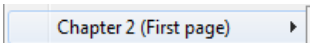
Note that if you have defined a different first page or different left/right pages for a section, the page type is also displayed in the menu (see below).

### Section attributes

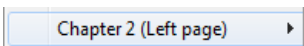
Sections inherit attributes from the document. However, common document attributes, including headers and footers, can be modified separately for each section. The contextual pop-up menu displays the properties and attributes available at the section level:



- **Page orientation:** allows you to set a specific page orientation (Portrait or Landscape) per section
- **Different first page:** allows you to set different attributes for the first page of the section; this feature can be used to create flyleaves, for example. When this attribute is checked, the first page of the section is handled as a subsection itself and can have its own attributes.



- **Different left and right pages:** allows you to set different attributes for left and right pages of the section. When this attribute is checked, left and right pages of the section are handled as subsections and can have their own attributes.



- **Header and Footer** commands: these options allow you to define separate headers and footers. These options are detailed below.
- **Margins / Paddings / Borders / Background:** these attributes can be defined separately for each section. For more information on these attributes, please refer the [4D Write Pro Attributes](#) article.

## Inserting headers and footers

Each section can have specific header and footer. Headers and footers are displayed only when the document page view mode is **Page**.

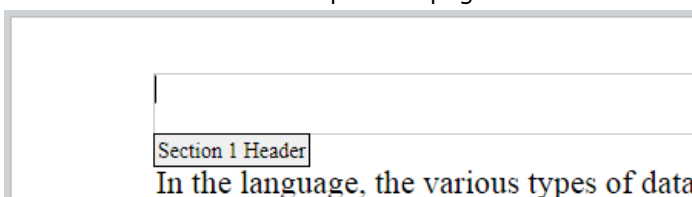
Within a section, you can define up to three different headers and footers, depending on the enabled options:

- first page,
- left page(s),
- right page(s).

To create a header or a footer:

1. Make sure the document is in **Page** view mode.
2. Double-click in the header or footer area of the desired section and page to switch to editing mode.

- The header area is at the top of the page:



- The footer area is at the bottom of the page:

date, 4D will assume that you want to

Section 1 Footer

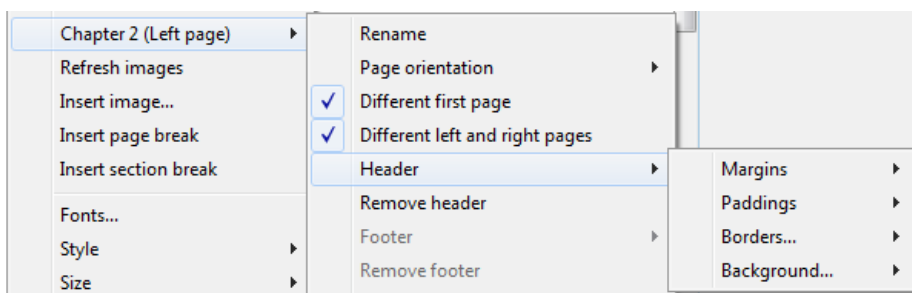
You can then enter any static contents, which will be repeated automatically on each page of the section (except for the first page, if enabled).

Chapter 2 Footer (Left page)

Chapter 2 - How to become famous

You can also insert dynamic contents such as the page number or the page count using the **ST INSERT EXPRESSION** command (for more information, please refer to the [Inserting document and page expressions](#) paragraph).

Once a header or a footer has been defined for a section, you can configure its common attributes using the contextual menu:



For more information on **Margins**, **Paddings**, **Borders**, and **Background** attributes, please refer the [4D Write Pro Attributes](#) section.

You can remove the entire definition of a header or a footer (contents and attributes) by selecting the **Remove header** or **Remove footer** command in the contextual menu.

## Compatibility

4D Write Pro handles headers and footers of documents converted from the 4D Write plug-in.

The following expressions and properties are also supported and converted from the 4D Write plug-in headers and footers:

- page number and page count variables
- distinct first page
- distinct left/right pages

WP EXPORT DOCUMENT ( wpDoc ; filePath {; format {; option}} )

Parameter	Type		Description
wpDoc	Object	→	4D Write Pro variable
filePath	String	→	Path of exported file
format	Longint	→	Document output format
option	Longint	→	Export options

## Description

The **WP EXPORT DOCUMENT** command exports the *wpDoc* 4D Write Pro object to a document on disk according to the *filePath* parameter as well as any optional parameters.

In *wpDoc*, pass the 4D Write Pro object that you want to export.

In *filePath*, pass the destination path and the name of the document to be exported. If you pass only the document name, it will be saved at the same level as the 4D structure file. By default, if you omit the *format* parameter, the command uses the document extension to select the file format.

You can also pass a constant from the **4D Write Pro Constants** theme in the *format* parameter. In this case, 4D adds the appropriate extension to the file name if needed. The following formats are supported:

Constant	Type	Value	Comment
wk 4wp	Longint	4	The 4D Write Pro document is saved in a native archive format (zipped HTML and images saved in a separate folder). 4D specific tags are included and 4D expressions are not computed. This format is particularly suitable for saving and archiving 4D Write Pro documents on disk without any loss.
wk web page complete	Longint	2	.htm or .html extension. The document is saved as standard HTML and its resources are saved separately. 4D tags are removed and expressions are computed. This format is particularly suitable when you want to display a 4D Write Pro document in a web browser.

### Notes:

- "4D specific tags" means 4D XHTML with a 4D namespace and 4D CSS styles.
- Expressions can be frozen at any time before export using **ST FREEZE EXPRESSIONS**.
- For more information about the 4D Write Pro format, refer to **.4wp document format**.

In the *options* parameter, you pass options that will configure the export. You can pass a *longint* value to define the style of the HTML code. The following constants are available:

Constant	Type	Value	Comment
wk html debug	Longint	1	Formatted HTML code ("pretty print"), easier to debug
wk normal	Longint	0	Standard HTML code

- *HTML debug option off (default):*

```
<html xmlns="http://www.w3.org/1999/xhtml"><head><title>New 4D Write Pro Document</title><style type="text/css">body { background-color:#FFFFFF }ul, ol { margin:0;padding:0 }p,li { white-space:pre-wrap;margin:0pt;padding:0pt;font-family:'Times New Roman' }p.Normal,li.Normal { text-align:left }p._p1,li._p1 { font-family:'Arial';font-size:18pt;color:#1D1D1D }img._img1 { width:51pt;height:51pt }</style></head><body><p class="Normal _p1"><span>23/12/14</span></p><p class="Normal _p1">Dear <span style="color:#BE0E00">Shaun Stoltz</span><span style="color:#800062"> <span style="color:#000000">from
```

- *HTML debug option on:*

```
<html xmlns="http://www.w3.org/1999/xhtml"><head><title>New 4D Write Pro Document</title><style type="text/css">body { background-color:#FFFFFF }ul, ol { margin:0;padding:0 }
```



## Example 1

---

You want to export the contents of the *myArea* 4D Write Pro object to a document in your database folder. You can set the debug option using a 'pprint' button:

```
C_TEXT($filePath)
$filePath:=Get 4D folder(Database folder)+"Exported files"+Folder
separator+"WriteProExport.html"
If(pprint=0) //if the debug option is off
    WP EXPORT DOCUMENT(myArea;$filePath;wk_web_page_complete;wk_normal)
Else
    WP EXPORT DOCUMENT(myArea;$filePath;wk_web_page_complete;wk_html_debug)
End if
```

## Example 2

---

You want to export the contents of the *myArea* 4D Write Pro object in .4wp format:

```
C_TEXT($path)
C_LONGINT($docRef)

Case of
: (Form event=On Clicked)

    $path:=Get 4D folder(Database folder)+"Export"+Folder separator
    $path:=Select document($path;".4wp";" title";File name entry)

    If($path#command_5"")
        WP EXPORT DOCUMENT(myArea;document;wk_4wp;wk_normal)
    Else
        ALERT("An error occurred.")
    End if
End case
```

## ST INSERT EXPRESSION

ST INSERT EXPRESSION ( {\* ;} object ; expression {; startSel {; endSel}} )

Parameter	Type	Description
*	Operator	➔ If specified, object is an object name (string) If omitted, object is a field or variable
object	Object	➔ Object name (if * is specified) or Field or variable (if * is omitted)
expression	Text	➔ Expression and (optional) format to insert
startSel	Longint	➔ Start of selection
endSel	Longint	➔ End of selection

### Description

The **ST INSERT EXPRESSION** command inserts a reference to the *expression* in the styled text field or variable designated by the *object* parameter.

Passing the optional \* parameter indicates that the *object* parameter is an object name (string). If you do not pass this parameter, it indicates that the *object* parameter is a field or variable. In this case, you pass a field or variable reference instead of a string (field or variable object only).

In the *expression* parameter, you pass the 4D expression to evaluate in the *object*. A valid 4D expression is a string returning a value. *expression* can be a field, a variable, a 4D command, a statement returning a value, a project method, and so on.

The expression must be placed in quotation marks ("").

**Note:** The *expression* parameter cannot be a Picture type variable.

If *expression* returns a value containing carriage returns and tabs, 4D formats the text according to the object hosting the expression; carriage return characters are interpreted as line breaks.

You can format the expression by including formatting information in the *expression* parameter. In this case, the parameter must be in the form:

```
"String(value;format)"
```

... where *value* contains the expression itself and *format* contains the format to apply. The *format* parameter can have the following values:

- for numbers: any number display format (existing or not), for example `"#command_5_command_5#command_5,#command_5#command_5"`.
- for dates: a number designating an existing date format. You can use the constants of the "**Date Display Formats**" theme, for example `System date short`.
- for times: a number designating an existing time format. You can use the constants of the "**Time Display Formats**" theme, for example `System time short`.

For example:

```
"String([Table_1]Field_1;System date short)"
```

By default, the expression **values** are displayed in the multi-style text areas. You can force the display of the **references** instead using the **ST SET OPTIONS** command.

The optional *startSel* and *endSel* parameters designate a selection of text in *object*. The *startSel* and *endSel* values express a plain text selection, without taking into account any style tags that may be present.

- If you only pass *startSel*, the result of the expression is inserted at the specified location.
- If you omit *startSel* and *endSel*, the result of the expression is inserted at the location of the cursor.
- If you pass *startSel* and *endSel*, **ST INSERT EXPRESSION** replaces the text in this selection with the result of the *expression*. If the value of *endSel* is greater than the total number of characters in the object, all the characters between *startSel* and the end of the text are replaced by the result of the *expression*.

4D provides predefined constants so that you can designate the selection limits automatically in the *startSel* and *endSel* parameters. These constants are found in the "**Multistyle Text**" theme:

Constant	Type	Value	Comment
ST End highlight	Longint	-1001	Designates last character of current text selection in object (*)
ST End text	Longint	0	Designates last character of text contained in object
ST Start highlight	Longint	-1000	Designates first character of current text selection in object (*)
ST Start text	Longint	1	Designates first character of text contained in object

(\*) You must pass an object name in *object* to be able to use this constant. If you pass a reference to a field or variable, the command is applied to all the text of the object.

**Note:** If *startSel* is greater than *endSel* (except when *endSel* is 0), the command does nothing and the *OK* variable is set to 0.

## Example

---

You want to replace the highlighted text with the result of a project method:

```
ST INSERT EXPRESSION (*;"myText";"MyMethod";ST Start highlight;ST End highlight)
```

## ☰ 4D Write Pro Attributes

---

4D Write Pro attributes allow you to control all the graphical aspects of text and images stored in your documents. These attributes are handled by the following commands:

- **WP SET ATTRIBUTES**
- **WP GET ATTRIBUTES**
- **WP RESET ATTRIBUTES**

**Note:** The generic 4D commands **OB SET** and **OB Get** can also be used to work with 4D Write Pro area attributes, but with a limitation concerning the direct use of pictures (for more information, please refer to the description of these commands).

### Background

---

Background attributes are used to define background effects in your documents. They can be applied to:

Documents	Paragraphs	Characters	Pictures
X	X		X

Constant	Comment
	Specifies painting area of background. Possible values:
wk background clip	<ul style="list-style-type: none"> <li>• <a href="#">wk border box</a> (default): background is painted to outside edge of the border</li> <li>• <a href="#">wk content box</a>: background is painted within the content box</li> <li>• <a href="#">wk padding box</a>: background is painted to outside edge of the padding (or to inside edge of the border, if any)</li> </ul>
	Specifies background color of an element. Possible values:
wk background color	<ul style="list-style-type: none"> <li>• a CSS color ("<a href="#">#command_5010101</a>" or "<a href="#">#command_5FFFFFF</a>" or "red").</li> <li>• a 4D color longint value (see <a href="#">OBJECT SET COLOR</a> command)</li> <li>• a longint array containing an element for each R, G, B component (0-255)</li> </ul>
	Default for documents is " <a href="#">#command_5FFFFFF</a> " and <a href="#">wk transparent</a> , or "transparent" for paragraphs and images.
	Specifies image to use as background. Possible values to set:
wk background image	<ul style="list-style-type: none"> <li>• Image URL (string). Can be absolute or relative to the structure file.</li> <li>• Picture variable or field.</li> </ul> <p>Value returned (<a href="#">WP GET ATTRIBUTES</a>): URI (network URL or data URI). It may not be equal to the initial URL for an image not referenced with the network URL (only network URLs are kept). For local file URLs, the image stream itself is kept in the document and thus the URL returned is a data URI with the image stream encoded in base64.</p>
	Specifies where background image is positioned. Possible values:
wk background origin	<ul style="list-style-type: none"> <li>• <a href="#">wk padding box</a> (default): background image starts at padding (or inside border edge) rectangle</li> <li>• <a href="#">wk border box</a>: background image starts at border (outside edge) rectangle</li> <li>• <a href="#">wk content box</a>: background image starts at content rectangle</li> </ul>
	Specifies horizontal starting position of a background image. Possible values:
wk background position h	<ul style="list-style-type: none"> <li>• <a href="#">wk left</a> (default): background image starts horizontally on left side of the element</li> <li>• <a href="#">wk center</a>: background image starts horizontally at center of the element</li> <li>• <a href="#">wk right</a>: background image starts horizontally on right side of the element</li> </ul>
	Specifies vertical starting position of a background image. Possible values:
wk background position v	<ul style="list-style-type: none"> <li>• <a href="#">wk top</a> (default): background image starts vertically at top of the element</li> <li>• <a href="#">wk middle</a>: background image starts vertically at middle of the element</li> <li>• <a href="#">wk bottom</a>: background image starts vertically at bottom of the element</li> </ul>
	Specifies if and how a background image is repeated. Possible values:
wk background repeat	<ul style="list-style-type: none"> <li>• <a href="#">wk repeat</a> (default): background image is repeated both vertically and horizontally</li> <li>• <a href="#">wk no repeat</a>: background image is not repeated</li> <li>• <a href="#">wk repeat x</a>: background image is repeated only horizontally</li> <li>• <a href="#">wk repeat y</a>: background image is repeated only vertically</li> </ul>
	Specifies horizontal size of background image. Possible values:
wk background size h	<ul style="list-style-type: none"> <li>• <a href="#">wk auto</a> (default): background image contains its width</li> <li>• <a href="#">wk contain</a>: scales image to largest size so that it fits entirely in the content area, while preserving its aspect ratio. This option also modifies the value of the other size attribute.</li> <li>• <a href="#">wk cover</a>: scales background image to be as large as possible so that the background area is entirely covered by the background image, while preserving its aspect ratio. Some parts of the background image may be cropped. This option also modifies the value of the other size attribute.</li> </ul>

- Defined size: background image horizontal size expressed using a real or string value:
  - Real: Size in [wk layout unit](#).
  - String: CSS string with value and unit concatenated. E.g.: 12pt for 12 points, or 1.5cm for 1.5 centimeters. Minimum value: 0pt, maximum value: 10000pt. A relative value (percentage %) is supported.

Specifies vertical size of background image. Possible values:

- [wk auto](#) (default): background image contains its height
- [wk contain](#): scales image to largest size so that it fits entirely in the content area, while preserving its aspect ratio. This option also modifies the value of the other size attribute.
- [wk cover](#): scales background image to be as large as possible so that the background area is entirely covered by the background image, while preserving its aspect ratio. Some parts of the background image may be cropped. This option also modifies the value of the other size attribute.
- Defined size: background image vertical size expressed using a real or string value:
  - Real: Size in [wk layout unit](#).
  - String: CSS string with value and unit concatenated. E.g.: 12pt for 12 points, or 1.5cm for 1.5 centimeters. Minimum value: 0pt, maximum value: 10000pt. A relative value (percentage %) is supported.

wk  
background  
size v

## Borders

---

Border attributes are used to specify the style, width, and color of an element's border. They can be applied to:

Documents	Paragraphs	Characters	Pictures
X	X		X

Constant	Comment
	Sets color of all four borders. Possible values:
wk border color	<ul style="list-style-type: none"> <li>• a CSS color ("#command_5010101" or "#command_5FFFFFFF" or "red").</li> <li>• a 4D color longint value (see <b>OBJECT SET COLOR</b> command)</li> <li>• a longint array containing an element for each R, G, B component (0-255)</li> </ul>
	Default is "#command_5000000" (if string value). If there are multiple colors, <b>WP GET ATTRIBUTES</b> returns an empty string.
	Sets color of bottom border. Possible values:
wk border color bottom	<ul style="list-style-type: none"> <li>• a CSS color ("#command_5010101" or "#command_5FFFFFFF" or "red"). Default is "#command_5000000"</li> <li>• a 4D color longint value (see <b>OBJECT SET COLOR</b> command)</li> <li>• a longint array containing an element for each R, G, B component (0-255)</li> </ul>
	Sets color of left border. Possible values:
wk border color left	<ul style="list-style-type: none"> <li>• a CSS color ("#command_5010101" or "#command_5FFFFFFF" or "red"). Default is "#command_5000000"</li> <li>• a 4D color longint value (see <b>OBJECT SET COLOR</b> command)</li> <li>• a longint array containing an element for each R, G, B component (0-255)</li> </ul>
	Sets color of right border. Possible values:
wk border color right	<ul style="list-style-type: none"> <li>• a CSS color ("#command_5010101" or "#command_5FFFFFFF" or "red"). Default is "#command_5000000"</li> <li>• a 4D color longint value (see <b>OBJECT SET COLOR</b> command)</li> <li>• a longint array containing an element for each R, G, B component (0-255)</li> </ul>
	Sets color of top border. Possible values:
wk border color top	<ul style="list-style-type: none"> <li>• a CSS color ("#command_5010101" or "#command_5FFFFFFF" or "red"). Default is "#command_5000000"</li> <li>• a 4D color longint value (see <b>OBJECT SET COLOR</b> command)</li> <li>• a longint array containing an element for each R, G, B component (0-255)</li> </ul>
	Specifies a rounded border. Possible values:
wk border radius	<ul style="list-style-type: none"> <li>• <u>wk none</u> (default): the border does not have rounded angles</li> <li>• Radius value expressed using an integer or a string value: <ul style="list-style-type: none"> <li>◦ Integer: Radius in <u>wk layout unit</u>.</li> <li>◦ String: CSS string with value and unit concatenated. E.g.: 12pt for 12 points, or 1.5cm for 1.5 centimeters.</li> </ul> </li> </ul>
	Specifies style of all four borders. Possible values:
wk border style	<ul style="list-style-type: none"> <li>• <u>wk none</u> (default): no border</li> <li>• <u>wk hidden</u>: same as <u>wk none</u>, except in border conflict resolution</li> <li>• <u>wk solid</u>: solid border</li> <li>• <u>wk dotted</u>: dotted border</li> <li>• <u>wk dashed</u>: dashed border</li> <li>• <u>wk double</u>: double border</li> <li>• <u>wk groove</u>: 3D groove border (the actual effect depends on the border color)</li> <li>• <u>wk ridge</u>: 3D ridged border (the actual effect depends on the border color)</li> <li>• <u>wk inset</u>: 3D inset border (the actual effect depends on the border color)</li> </ul>
	Specifies style of bottom border. Possible values:
	<ul style="list-style-type: none"> <li>• <u>wk none</u> (default): no bottom border</li> </ul>

- wk border style bottom
- [wk hidden](#): same as [wk none](#), except in border conflict resolution
  - [wk solid](#): solid bottom border
  - [wk dotted](#): dotted bottom border
  - [wk dashed](#): dashed bottom border
  - [wk double](#): double bottom border
  - [wk groove](#): 3D groove bottom border (the actual effect depends on the border color)
  - [wk ridge](#): 3D ridged bottom border (the actual effect depends on the border color)
  - [wk inset](#): 3D inset bottom border (the actual effect depends on the border color)

Specifies style of left border. Possible values:

- wk border style left
- [wk none](#) (default): no left border
  - [wk hidden](#): same as [wk none](#), except in border conflict resolution
  - [wk solid](#): solid left border
  - [wk dotted](#): dotted left border
  - [wk dashed](#): dashed left border
  - [wk double](#): double left border
  - [wk groove](#): 3D groove left border (the actual effect depends on the border color)
  - [wk ridge](#): 3D ridged left border (the actual effect depends on the border color)
  - [wk inset](#): 3D inset left border (the actual effect depends on the border color)

Specifies style of right border. Possible values:

- wk border style right
- [wk none](#) (default): no right border
  - [wk hidden](#): same as [wk none](#), except in border conflict resolution
  - [wk solid](#): solid right border
  - [wk dotted](#): dotted right border
  - [wk dashed](#): dashed right border
  - [wk double](#): double right border
  - [wk groove](#): 3D groove right border (the actual effect depends on the border color)
  - [wk ridge](#): 3D ridged right border (the actual effect depends on the border color)
  - [wk inset](#): 3D inset right border (the actual effect depends on the border color)

Specifies style of top border. Possible values:

- wk border style top
- [wk none](#) (default): no top border
  - [wk hidden](#): same as [wk none](#), except in border conflict resolution
  - [wk solid](#): solid top border
  - [wk dotted](#): dotted top border
  - [wk dashed](#): dashed top border
  - [wk double](#): double top border
  - [wk groove](#): 3D groove top border (the actual effect depends on the border color)
  - [wk ridge](#): 3D ridged top border (the actual effect depends on the border color)
  - [wk inset](#): 3D inset top border (the actual effect depends on the border color)

Specifies width of all four borders. You need to specify the border style before setting the border width. Possible values:

- wk border width
- Width expressed using an integer or a string value:
    - Integer: Width in [wk layout unit](#).
    - String: CSS string with value and unit concatenated. E.g.: 12pt for 12 points, or 1.5cm for 1.5 centimeters
  - Default value: 2pt

Specifies width of bottom border. Possible values:

- wk border width
- Width expressed using an integer or a string value:
    - Integer: Width in [wk layout unit](#).
    - String: CSS string with value and unit concatenated. E.g.: 12pt for 12 points, or 1.5cm



bottom for 1.5 centimeters

- Default value: 2pt

Specifies width of left border. Possible values:

wk  
border  
width left

- Width expressed using an integer or a string value:
  - Integer: Width in [wk layout unit](#).
  - String: CSS string with value and unit concatenated. E.g.: 12pt for 12 points, or 1.5cm for 1.5 centimeters
- Default value: 2pt

Specifies width of right border. Possible values:

wk  
border  
width  
right

- Width expressed using an integer or a string value:
  - Integer: Width in [wk layout unit](#).
  - String: CSS string with value and unit concatenated. E.g.: 12pt for 12 points, or 1.5cm for 1.5 centimeters
- Default value: 2pt

Specifies width of top border. Possible values:

wk  
border  
width top

- Width expressed using an integer or a string value:
  - Integer: Width in [wk layout unit](#).
  - String: CSS string with value and unit concatenated. E.g.: 12pt for 12 points, or 1.5cm for 1.5 centimeters
- Default value: 2pt

wk inside

When the selected area contains several paragraphs, specifies that the attribute should affect only the corresponding inter-paragraph property (not outside). It applies only to border, padding and margin attributes, and must be added to the specified attribute. See example 2 of the **WP SET ATTRIBUTES** command.

wk  
outside

When the selected area contains several paragraphs, specifies that the attribute should affect only the corresponding paragraph external property (not inside). It applies only to border, padding and margin attributes, and must be added to the specified attribute. See example 2 of the **WP SET ATTRIBUTES** command.

## Document

---

Document attributes are used to set or get standard document information such as the document subject, the author's name, the company name and the notes. They can be applied to:

Documents Paragraphs Characters Pictures

X

Constant	Comment
wk author	Specifies name of author of the document (string)
wk company	Specifies a company associated with the document (string)
wk date creation	Returns creation date of document (date). This value is read-only and cannot be set.
wk date modified	Returns last modification date of document (date). This value is read-only and cannot be set.
wk dpi	DPI used for internal pixels <->points conversion (integer). Always 96 (read-only) Specifies unit of dimension of some attributes when value is set or get as a integer or real. Possible values:
wk layout unit	<ul style="list-style-type: none"> <li>• <a href="#">wk unit cm</a> (default): centimeters</li> <li>• <a href="#">wk unit pt</a>: points</li> <li>• <a href="#">wk unit px</a>: pixels</li> <li>• <a href="#">wk unit percent</a> (only for <a href="#">wk line height</a> and <a href="#">wk background size h</a> / <a href="#">wk background size v</a>)</li> <li>• <a href="#">wk unit mm</a>: millimeters</li> <li>• <a href="#">wk unit inch</a>: inches</li> </ul>
wk notes	Specifies comments about the document (string).
wk subject	Specifies document subject (string)
wk title	Specifies document title (string). Default is "New 4D Write Pro Document"
wk version	Returns internal 4DWP version of the document (real). This number is only read using <b>WP GET ATTRIBUTES</b> ; it cannot be set.

## Fonts and text

---

These attributes define the font family, size, and style of the text. They can be applied to:

Documents   Paragraphs   Characters   Pictures  
X

Constant	Comment
wk font	Specifies complete font name with styles, as returned by the <b>FONT STYLE LIST</b> command. If you set an invalid font name, the command does nothing. Default value: "Times New Roman". Specifies thickness of text (depends on available font styles). Possible values:
wk font bold	<ul style="list-style-type: none"> <li>• <u>wk true</u> to set selected characters to bold font style; with the <b>WP GET ATTRIBUTES</b> command, <u>wk true</u> is returned if at least one selected character supports a bold font style.</li> <li>• <u>wk false</u> (default) to remove the bold font style from selected characters if any; with the <b>WP GET ATTRIBUTES</b> command, <u>wk false</u> is returned if none of the selected characters supports a bold font style.</li> </ul>
wk font family	Specifies font family name as defined by <u>wk font</u> . Default value is "Times New Roman". An empty string is returned by the <b>WP GET ATTRIBUTES</b> command if the selected characters contain different font family properties. Specifies italic style of text (depends on available font styles). Possible values:
wk font italic	<ul style="list-style-type: none"> <li>• <u>wk true</u> to set selected characters to italic or oblique font style; with the <b>WP GET ATTRIBUTES</b> command, <u>wk true</u> is returned if at least one selected character supports an italic or oblique font style.</li> <li>• <u>wk false</u> (default) to remove the italic or oblique font style from selected characters if any; with the <b>WP GET ATTRIBUTES</b> command, <u>wk false</u> is returned if none of the selected characters supports an italic or oblique font style.</li> </ul>
wk font size	Specifies font size for text. Possible values (in points only): <ul style="list-style-type: none"> <li>• Real value (default = 12)</li> <li>• CSS string with value and unit concatenated. E.g.: 12pt for 12 points.</li> </ul>
wk text color	Specifies color of text. Possible values: <ul style="list-style-type: none"> <li>• a CSS color ("command_5010101" or "command_5FFFFFF" or "red"). Default is "command_5000000" if string.</li> <li>• a 4D color longint value (see <b>OBJECT SET COLOR</b> command)</li> <li>• a longint array containing an element for each R, G, B component (0-255)</li> </ul>
wk text linethrough color	Specifies color of text linethrough. Possible values: <ul style="list-style-type: none"> <li>• a CSS color ("command_5010101" or "command_5FFFFFF" or "red").</li> <li>• a 4D color longint value (see <b>OBJECT SET COLOR</b> command)</li> <li>• a longint array containing an element for each R, G, B component (0-255)</li> </ul>
wk text linethrough style	Default is "currentColor" if string, or <u>wk default</u> if longint. Specifies style of text linethrough (if any). Possible values: <ul style="list-style-type: none"> <li>• <u>wk none</u> (default): no linethrough effect</li> <li>• <u>wk solid</u>: draw a solid line on the selected text</li> <li>• <u>wk dotted</u>: draw a dotted line on the selected text</li> <li>• <u>wk dashed</u>: draw a dashed line on the selected text</li> <li>• <u>wk double</u>: draw a double line on the selected text</li> <li>• <u>wk semi transparent</u>: dimmed line on the selected text. Can be combined with another line style.</li> <li>• <u>wk word</u>: draw a line on words only (exclude blank spaces). Can be combined with another line style.</li> </ul>
wk text shadow color	Specifies shadow color of the selected text. Possible values: <ul style="list-style-type: none"> <li>• a CSS color ("command_5010101" or "command_5FFFFFF" or "red").</li> <li>• a 4D color longint value (see <b>OBJECT SET COLOR</b> command)</li> <li>• a longint array containing an element for each R, G, B component (0-255)</li> </ul>

- [wk transparent](#) (default)

**wk text shadow offset** Specifies offset for shadow effect. Possible values:

- Size expressed in points. Default value: 1pt

Specifies uppercase and lowercase letters in the text. Possible values:

- [wk capitalize](#): first letters are set to uppercase
- [wk lowercase](#): letters are set to lowercase
- [wk uppercase](#): letters are set to uppercase
- [wk small uppercase](#): letters are set to small uppercase
- [wk none](#) (default): no transformation

Specifies color of text underline. Possible values:

- a CSS color ("[#command\\_5010101](#)" or "[#command\\_5FFFFFF](#)" or "red").
- a 4D color longint value (see **OBJECT SET COLOR** command)
- a longint array containing an element for each R, G, B component (0-255)

Default is "currentColor" if string, or [wk default](#) if longint.

Specifies style of text underline (if any). Possible values:

- [wk none](#) (default): no underline
- [wk solid](#): draw a solid underline
- [wk dotted](#): draw a dotted underline
- [wk dashed](#): draw a dashed underline
- [wk double](#): draw a double underline
- [wk semi transparent](#): dimmed underline. Can be combined with another line style.
- [wk word](#): draw an underline for words only (exclude blank spaces). Can be combined with another line style.

**wk text underline style**

Sets vertical alignment of an element. Can be used with characters, paragraphs, and pictures. Possible values:

- [wk baseline](#) (default): aligns baseline of element with baseline of parent element
- [wk top](#): aligns top of element with top of tallest element on the line
- [wk bottom](#): aligns bottom of element with lowest element on the line
- [wk middle](#): element is placed in middle of parent element
- [wk superscript](#): aligns element as if it were superscript
- [wk subscript](#): aligns element as if it were subscript

**wk vertical align**

For characters, [wk top](#) and [wk bottom](#) have the same effect as [wk baseline](#).

For paragraphs, [wk baseline](#), [wk superscript](#) and [wk subscript](#) have the same effect as [wk top](#).

## Height/Width

---

Height/width attributes are used to set the height and width of paragraphs and images. They can be applied to:

Documents	Paragraphs	Characters	Pictures
	X		X

Constant	Comment
wk height	<p>Sets height of element. The height property does not include padding, borders, or margins; it sets the height of the area inside the padding, border, and margin of the element. Possible values:</p> <ul style="list-style-type: none"> <li>• <a href="#">wk auto</a> (default): height is based upon the contents of the element</li> <li>• Defined size: size expressed using real or string value: <ul style="list-style-type: none"> <li>◦ Real: Size in <a href="#">wk layout unit</a>.</li> <li>◦ String: CSS string with value and unit concatenated. E.g.: 12pt for 12 points, or 1.5cm for 1.5 centimeters. Minimum value: 0pt, maximum value: 10000pt.</li> </ul> </li> </ul> <p>The <a href="#">wk height</a> attribute is overridden by <a href="#">wk min height</a> (if defined).</p> <p><b>Note:</b> In the current implementation, <a href="#">wk height</a> can only be used with pictures.</p> <p>Sets minimum height of the element. It prevents the value of the <a href="#">wk height</a> property from becoming smaller than <a href="#">wk min height</a>. Possible values:</p> <ul style="list-style-type: none"> <li>• <a href="#">wk auto</a> (default): minimum height is based upon the contents of the element</li> <li>• Defined size: size expressed using real or string value: <ul style="list-style-type: none"> <li>◦ Real: Size in <a href="#">wk layout unit</a>.</li> <li>◦ String: CSS string with value and unit concatenated. E.g.: 12pt for 12 points, or 1.5cm for 1.5 centimeters. Minimum value: 0pt, maximum value: 10000pt.</li> </ul> </li> </ul> <p>The <a href="#">wk min height</a> value overrides the <a href="#">wk height</a> attribute.</p> <p><b>Note:</b> In the current implementation, can only be used with pictures.</p> <p>Sets minimum width of element. It prevents the value of the <a href="#">wk width</a> property from becoming smaller than <a href="#">wk min width</a>. Possible values:</p> <ul style="list-style-type: none"> <li>• <a href="#">wk auto</a> (default): minimum width is based upon the contents of the element</li> <li>• Defined size: size expressed using real or string value: <ul style="list-style-type: none"> <li>◦ Real: Size in <a href="#">wk layout unit</a>.</li> <li>◦ String: CSS string with value and unit concatenated. E.g.: 12pt for 12 points, or 1.5cm for 1.5 centimeters. Minimum value: 0pt, maximum value: 10000pt.</li> </ul> </li> </ul> <p>The <a href="#">wk min width</a> value overrides the <a href="#">wk width</a> attribute.</p> <p><b>Note:</b> In the current implementation, can only be used with pictures.</p> <p>Sets width of element. Possible values:</p> <ul style="list-style-type: none"> <li>• <a href="#">wk auto</a> (default): width is based upon the contents of the element</li> <li>• Defined size: size expressed using a real or string value: <ul style="list-style-type: none"> <li>◦ Real: Size in <a href="#">wk layout unit</a>.</li> <li>◦ String: CSS string with value and unit concatenated. E.g.: 12pt for 12 points, or 1.5cm for 1.5 centimeters. Minimum value: 0pt, maximum value: 10000pt.</li> </ul> </li> </ul> <p>The <a href="#">wk width</a> attribute is overridden by <a href="#">wk min width</a> if defined.</p> <p><b>Note:</b> In the current implementation, <a href="#">wk width</a> can only be used with pictures.</p>
wk min height	<p>Sets minimum height of the element. It prevents the value of the <a href="#">wk height</a> property from becoming smaller than <a href="#">wk min height</a>. Possible values:</p> <ul style="list-style-type: none"> <li>• <a href="#">wk auto</a> (default): minimum height is based upon the contents of the element</li> <li>• Defined size: size expressed using real or string value: <ul style="list-style-type: none"> <li>◦ Real: Size in <a href="#">wk layout unit</a>.</li> <li>◦ String: CSS string with value and unit concatenated. E.g.: 12pt for 12 points, or 1.5cm for 1.5 centimeters. Minimum value: 0pt, maximum value: 10000pt.</li> </ul> </li> </ul> <p>The <a href="#">wk min height</a> value overrides the <a href="#">wk height</a> attribute.</p> <p><b>Note:</b> In the current implementation, can only be used with pictures.</p>
wk min width	<p>Sets minimum width of element. It prevents the value of the <a href="#">wk width</a> property from becoming smaller than <a href="#">wk min width</a>. Possible values:</p> <ul style="list-style-type: none"> <li>• <a href="#">wk auto</a> (default): minimum width is based upon the contents of the element</li> <li>• Defined size: size expressed using real or string value: <ul style="list-style-type: none"> <li>◦ Real: Size in <a href="#">wk layout unit</a>.</li> <li>◦ String: CSS string with value and unit concatenated. E.g.: 12pt for 12 points, or 1.5cm for 1.5 centimeters. Minimum value: 0pt, maximum value: 10000pt.</li> </ul> </li> </ul> <p>The <a href="#">wk min width</a> value overrides the <a href="#">wk width</a> attribute.</p> <p><b>Note:</b> In the current implementation, can only be used with pictures.</p>
wk width	<p>Sets width of element. Possible values:</p> <ul style="list-style-type: none"> <li>• <a href="#">wk auto</a> (default): width is based upon the contents of the element</li> <li>• Defined size: size expressed using a real or string value: <ul style="list-style-type: none"> <li>◦ Real: Size in <a href="#">wk layout unit</a>.</li> <li>◦ String: CSS string with value and unit concatenated. E.g.: 12pt for 12 points, or 1.5cm for 1.5 centimeters. Minimum value: 0pt, maximum value: 10000pt.</li> </ul> </li> </ul> <p>The <a href="#">wk width</a> attribute is overridden by <a href="#">wk min width</a> if defined.</p> <p><b>Note:</b> In the current implementation, <a href="#">wk width</a> can only be used with pictures.</p>

## Image

Image attributes are used to handle pictures inserted in the area. They can be applied to:

Documents Paragraphs Characters Pictures

X

Constant	Comment
	<p>Specifies an image. Possible values to set:</p> <ul style="list-style-type: none"> <li>• Image URL (string). Can be absolute or relative to the structure file.</li> <li>• Picture variable or field.</li> </ul>
wk image	<p>Value returned (<b>WP GET ATTRIBUTES</b>): URI (network URL or data URI). It may not be equal to the initial URL for an image not referenced with the network URL (only network URLs are kept). For local file URLs, the image stream itself is kept in the document and thus the URL returned is a data URI with the image stream encoded in base64.</p>
wk image alternative text	<p>Specifies alternative text for image, if image cannot be displayed.</p>
wk vertical align	<p>Sets vertical alignment of an element. Can be used with characters, paragraphs, and pictures. Possible values:</p> <ul style="list-style-type: none"> <li>• <a href="#">wk baseline</a> (default): aligns baseline of element with baseline of parent element</li> <li>• <a href="#">wk top</a>: aligns top of element with top of tallest element on the line</li> <li>• <a href="#">wk bottom</a>: aligns bottom of element with lowest element on the line</li> <li>• <a href="#">wk middle</a>: element is placed in middle of parent element</li> <li>• <a href="#">wk superscript</a>: aligns element as if it were superscript</li> <li>• <a href="#">wk subscript</a>: aligns element as if it were subscript</li> </ul> <p>For characters, <a href="#">wk top</a> and <a href="#">wk bottom</a> have the same effect as <a href="#">wk baseline</a>. For paragraphs, <a href="#">wk baseline</a>, <a href="#">wk superscript</a> and <a href="#">wk subscript</a> have the same effect as <a href="#">wk top</a>.</p>

## Lists

---

4D Write Pro supports two main types of lists:

- unordered lists: where list items are marked with bullets
- ordered lists: where list items are marked with numbers or letters

List attributes are used to configure your lists and set different list item fonts or markers. They can be applied to:

Documents	Paragraphs	Characters	Pictures
	X		

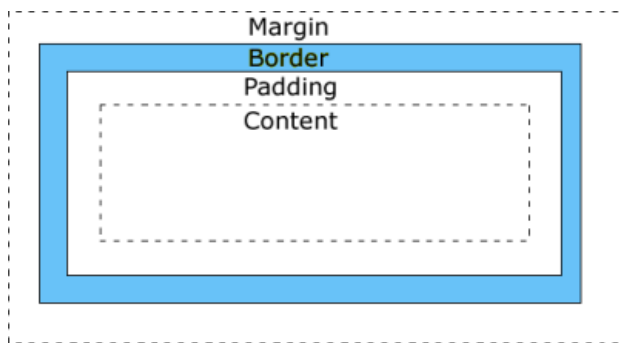
Constant	Comment
wk list font	Specifies complete font name, as returned by the <b>FONT STYLE LIST</b> command, to display the list item marker (but not the paragraph text). If the system does not recognize the font name, it handles the substitution. If you set an invalid font name, the command does nothing. Default value: "Times".
wk list font family	Specifies font family name as defined by <a href="#">wk list font</a> used to display the list item marker (but not the paragraph text). Default value is "Times New Roman".
wk list start number	Sets starting value of an ordered list. Possible values: <ul style="list-style-type: none"> <li>• <a href="#">wk auto</a> (default): starting value depends on previous list items if any.</li> <li>• an integer value: starting value</li> </ul>
wk list string format LTR	List item marker string format for left-to-right paragraph direction. If defined, it overrides default list item marker string format for the list. <ul style="list-style-type: none"> <li>• For unordered lists: string used as list item marker (usually a single character string, e.g. "-")</li> <li>• For ordered lists: string containing the "#command_5" character. "#command_5" is a placeholder for the computed number or letter(s). Default is "#command_5.", so for instance if current list item number is 15 and list style type is decimal, list item marker string will be "15."</li> </ul>
wk list string format RTL	List item marker string format for right-to-left paragraph direction. If defined, it overrides default list item marker string format for the list. <ul style="list-style-type: none"> <li>• For unordered lists: string used as list item marker (usually a single character string, e.g. "-")</li> <li>• For ordered lists: string containing the "#command_5" character. "#command_5" is a placeholder for the computed number or letter(s). Default is "#command_5.", so for instance if current list item number is 15 and list style type is decimal, list item marker string will be "15."</li> </ul>
wk list style image	Specifies an image as the list item marker in an unordered list. Possible values: <ul style="list-style-type: none"> <li>• <a href="#">wk none</a> (default): list item marker is not defined by an image</li> <li>• Local file image URL (string). Can be absolute or relative to the database resource directory</li> </ul> Value returned ( <b>WP GET ATTRIBUTES</b> ): URI (network URL or data URI). It may not be equal to the initial URL for an image not referenced with the network URL (only network URLs are kept). For local file URLs, the image stream itself is kept in the document and thus the URL returned is a data URI with the image stream encoded in base64.
wk list style image height	Sets height of image used as list item marker. Possible values: <ul style="list-style-type: none"> <li>• <a href="#">wk auto</a> (default): height is based upon image size</li> <li>• Defined size: size expressed using real or string value: <ul style="list-style-type: none"> <li>◦ Real: Size in <a href="#">wk layout unit</a>.</li> <li>◦ String: CSS string with value and unit concatenated. E.g.: 12pt for 12 points, or 1.5cm for 1.5 centimeters. Minimum value: 0pt, maximum value: 10000pt.</li> </ul> </li> </ul>
	Specifies type of ordered or unordered list item marker. Possible values are: <ul style="list-style-type: none"> <li>• <a href="#">wk disc</a> (default)</li> <li>• <a href="#">wk circle</a></li> <li>• <a href="#">wk square</a></li> <li>• <a href="#">wk decimal</a>: 1 2 3</li> <li>• <a href="#">wk decimal leading zero</a>: 01 02 03</li> <li>• <a href="#">wk lower latin</a>: a b c</li> <li>• <a href="#">wk lower roman</a>: i ii iii iv</li> <li>• <a href="#">wk upper latin</a>: A B C</li> <li>• <a href="#">wk upper roman</a>: I II III IV</li> <li>• <a href="#">wk lower greek</a>: alpha, beta, gamma, etc.</li> </ul>

- wk list style type
- [wk armenian](#)
  - [wk georgian](#)
  - [wk hebrew](#)
  - [wk hiragana](#)
  - [wk katakana](#)
  - [wk cjk ideographic](#)
  - [wk hollow square](#)
  - [wk diamond](#)
  - [wk club](#)
  - [wk decimal greek](#)
  - [wk custom](#): unordered list with "-" as default list item marker; this is a convenience style used in order to customize a list item marker with [wk list string format LTR](#) or [wk list string format RTL](#) without modifying standard list item markers
  - [wk none](#)

## Margins

---

Margins are the area that is outside the border of an element. They are transparent. The following picture illustrates the various elements that can be configured for a "box" element:



Margin attributes can be applied to:

Documents	Paragraphs	Characters	Pictures
X	X		X

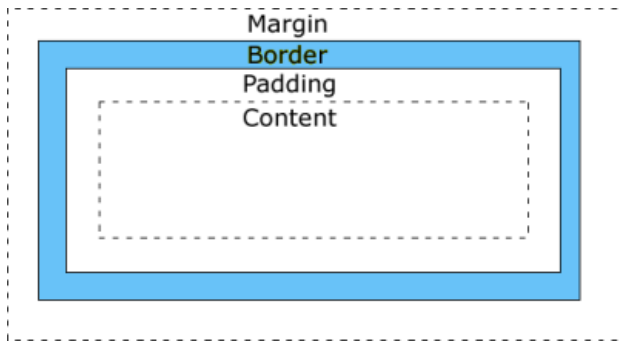


Constant	Comment
wk inside	<p>When the selected area contains several paragraphs, specifies that the attribute should affect only the corresponding inter-paragraph property (not outside). It applies only to border, padding and margin attributes, and must be added to the specified attribute. See example 2 of the <b>WP SET ATTRIBUTES</b> command.</p> <p>Specifies size for all margins of the element. Possible values:</p> <ul style="list-style-type: none"> <li>• Size expressed using an integer or a string value: <ul style="list-style-type: none"> <li>◦ Integer: size in <u>wk layout unit</u>.</li> <li>◦ String: CSS string with value and unit concatenated. E.g.: 12pt for 12 points, or 1.5cm for 1.5 centimeters</li> </ul> </li> <li>• <u>wk none</u> (default): no specific margin</li> </ul>
wk margin	
margin	
wk margin bottom	<p>Specifies size for bottom margin of the element. Possible values:</p> <ul style="list-style-type: none"> <li>• Size expressed using an integer or a string value: <ul style="list-style-type: none"> <li>◦ Integer: size in <u>wk layout unit</u>.</li> <li>◦ String: CSS string with value and unit concatenated. E.g.: 12pt for 12 points, or 1.5cm for 1.5 centimeters</li> </ul> </li> <li>• <u>wk none</u> (default): no specific margin</li> </ul>
wk margin left	<p>Specifies size for left margin of the element. Possible values:</p> <ul style="list-style-type: none"> <li>• Size expressed using an integer or a string value: <ul style="list-style-type: none"> <li>◦ Integer: size in <u>wk layout unit</u>.</li> <li>◦ String: CSS string with value and unit concatenated. E.g.: 12pt for 12 points, or 1.5cm for 1.5 centimeters</li> </ul> </li> <li>• <u>wk none</u> (default): no specific margin</li> </ul>
wk margin right	<p>Specifies size for right margin of the element. Possible values:</p> <ul style="list-style-type: none"> <li>• Size expressed using an integer or a string value: <ul style="list-style-type: none"> <li>◦ Integer: size in <u>wk layout unit</u>.</li> <li>◦ String: CSS string with value and unit concatenated. E.g.: 12pt for 12 points, or 1.5cm for 1.5 centimeters</li> </ul> </li> <li>• <u>wk none</u> (default): no specific margin</li> </ul>
wk margin top	<p>Specifies size for top margin of the element. Possible values:</p> <ul style="list-style-type: none"> <li>• Size expressed using an integer or a string value: <ul style="list-style-type: none"> <li>◦ Integer: size in <u>wk layout unit</u>.</li> <li>◦ String: CSS string with value and unit concatenated. E.g.: 12pt for 12 points, or 1.5cm for 1.5 centimeters</li> </ul> </li> <li>• <u>wk none</u> (default): no specific margin</li> </ul>
wk outside	<p>When the selected area contains several paragraphs, specifies that the attribute should affect only the corresponding paragraph external property (not inside). It applies only to border, padding and margin attributes, and must be added to the specified attribute. See example 2 of the <b>WP SET ATTRIBUTES</b> command.</p>

## Padding

Padding is the white space between the element content and the element border. Padding is affected by the background color of the element.

The following picture illustrates the various elements that can be configured for a "box" element:



Padding attributes can be applied to:

Documents	Paragraphs	Characters	Pictures
X	X		X

Constant	Comment
wk inside	When the selected area contains several paragraphs, specifies that the attribute should affect only the corresponding inter-paragraph property (not outside). It applies only to border, padding and margin attributes, and must be added to the specified attribute. See example 2 of the <b>WP SET ATTRIBUTES</b> command.
wk outside	When the selected area contains several paragraphs, specifies that the attribute should affect only the corresponding paragraph external property (not inside). It applies only to border, padding and margin attributes, and must be added to the specified attribute. See example 2 of the <b>WP SET ATTRIBUTES</b> command.
wk padding	<p>Specifies size of padding for all sides of the element. Possible values:</p> <ul style="list-style-type: none"> <li>• Size expressed using an integer or a string value: <ul style="list-style-type: none"> <li>◦ Integer: size in <a href="#">wk layout unit</a>.</li> <li>◦ String: CSS string with value and unit concatenated. E.g.: 12pt for 12 points, or 1.5cm for 1.5 centimeters</li> </ul> </li> <li>• <a href="#">wk none</a> (default): no specific padding</li> </ul>
wk padding bottom	<p>Specifies size of padding for bottom of the element. Possible values:</p> <ul style="list-style-type: none"> <li>• Size expressed using an integer or a string value: <ul style="list-style-type: none"> <li>◦ Integer: size in <a href="#">wk layout unit</a>.</li> <li>◦ String: CSS string with value and unit concatenated. E.g.: 12pt for 12 points, or 1.5cm for 1.5 centimeters</li> </ul> </li> <li>• <a href="#">wk none</a> (default): no specific padding</li> </ul>
wk padding left	<p>Specifies size of padding for left side of the element. Possible values:</p> <ul style="list-style-type: none"> <li>• Size expressed using an integer or a string value: <ul style="list-style-type: none"> <li>◦ Integer: size in <a href="#">wk layout unit</a>.</li> <li>◦ String: CSS string with value and unit concatenated. E.g.: 12pt for 12 points, or 1.5cm for 1.5 centimeters</li> </ul> </li> <li>• <a href="#">wk none</a> (default): no specific padding</li> </ul>
wk padding right	<p>Specifies size of padding for right side of the element. Possible values:</p> <ul style="list-style-type: none"> <li>• Size expressed using an integer or a string value: <ul style="list-style-type: none"> <li>◦ Integer: size in <a href="#">wk layout unit</a>.</li> <li>◦ String: CSS string with value and unit concatenated. E.g.: 12pt for 12 points, or 1.5cm for 1.5 centimeters</li> </ul> </li> <li>• <a href="#">wk none</a> (default): no specific padding</li> </ul>
wk padding top	<p>Specifies size of padding for top of the element. Possible values:</p> <ul style="list-style-type: none"> <li>• Size expressed using an integer or a string value: <ul style="list-style-type: none"> <li>◦ Integer: size in <a href="#">wk layout unit</a>.</li> <li>◦ String: CSS string with value and unit concatenated. E.g.: 12pt for 12 points, or 1.5cm for 1.5 centimeters</li> </ul> </li> <li>• <a href="#">wk none</a> (default): no specific padding</li> </ul>

## Paragraphs

---

Paragraph attributes are used to define properties for the text organization within a paragraph. They can be applied to:

Documents Paragraphs Characters Pictures  
X

Constant	Comment
wk direction	<p>Specifies text direction of paragraph. Possible values:</p> <ul style="list-style-type: none"> <li>• <a href="#">wk left to right</a> (default)</li> <li>• <a href="#">wk right to left</a></li> </ul>
wk line height	<p>Specifies space between lines. Possible values:</p> <ul style="list-style-type: none"> <li>• <a href="#">wk normal</a> (default): use value based upon text size</li> <li>• Height expressed using an integer or a string value: <ul style="list-style-type: none"> <li>◦ Integer: height in <a href="#">wk layout unit</a>.</li> <li>◦ String: CSS string with value and unit concatenated. E.g.: 12pt for 12 points, or 1.5cm for 1.5 centimeters. A relative value (percentage %) is supported.</li> </ul> </li> </ul>
wk tab stop offsets	<p>Specifies tab stops for the paragraph. Possible values:</p> <ul style="list-style-type: none"> <li>• Scalar value (default is 35.45pt): default offset for the whole paragraph. The <b>WP GET ATTRIBUTES</b> command returns the last offset (which is the default relative offset for offsets beyond the last absolute offset).</li> <li>• Array of tab values: an ordered list of absolute values, starting from the left margin. The tab offset defined by the last value is repeated for each additional tab character entered in the paragraph. If the tab offset is greater than the paragraph width, the text goes on the next line and starts from the first tab value. If a value in the array is smaller than the previous value, it is ignored.</li> </ul> <p><b>Note:</b> You cannot use arrays and scalars in the same call for different attributes.</p> <p>Values are expressed using CSS strings (default) or Real values in <a href="#">wk layout unit</a>. Maximum value is 10000pt.</p>
wk tab stop types	<p>Specifies tab stop type for the paragraph. Possible values:</p> <ul style="list-style-type: none"> <li>• <a href="#">wk left</a> (default): text extends to the right from the tab stop</li> <li>• <a href="#">wk right</a>: text extends to the left from the tab stop until the tab's space is filled</li> <li>• <a href="#">wk center</a>: text is centered at the tab stop</li> <li>• <a href="#">wk decimal</a>: text before the decimal point extends to the left, and text after the decimal point extends to the right</li> <li>• <a href="#">wk bar</a>: a vertical line at the specified position</li> <li>• array of tab stop type values (if tab stops have been defined through an array).</li> </ul>
wk text align	<p>Specifies horizontal alignment of text in the paragraph. Possible values:</p> <ul style="list-style-type: none"> <li>• <a href="#">wk left</a> (default)</li> <li>• <a href="#">wk right</a></li> <li>• <a href="#">wk justify</a></li> <li>• <a href="#">wk center</a></li> </ul>
wk text indent	<p>Specifies indentation of first line in the paragraph. Possible values:</p> <ul style="list-style-type: none"> <li>• Real: Size in <a href="#">wk layout unit</a>. Default is 0.</li> <li>• String: CSS string with value and unit concatenated. E.g.: 12pt for 12 points, or 1.5cm for 1.5 centimeters. Minimum value: 0pt, maximum value: 10000pt.</li> </ul>
wk vertical align	<p>Sets vertical alignment of an element. Can be used with characters, paragraphs, and pictures. Possible values:</p> <ul style="list-style-type: none"> <li>• <a href="#">wk baseline</a> (default): aligns baseline of element with baseline of parent element</li> <li>• <a href="#">wk top</a>: aligns top of element with top of tallest element on the line</li> <li>• <a href="#">wk bottom</a>: aligns bottom of element with lowest element on the line</li> <li>• <a href="#">wk middle</a>: element is placed in middle of parent element</li> <li>• <a href="#">wk superscript</a>: aligns element as if it were superscript</li> </ul>

- [wk subscript](#): aligns element as if it were subscript

For characters, [wk top](#) and [wk bottom](#) have the same effect as [wk baseline](#).

For paragraphs, [wk baseline](#), [wk superscript](#) and [wk subscript](#) have the same effect as [wk top](#).

## Style sheets

---

Style sheet attributes are used to handle specified style sheets. They can be applied to:

Documents	Paragraphs	Characters	Pictures
	X		X

Constant	Comment
	Specifies style sheet to use when adding a new line in the paragraph. Possible values:
wk new line style sheet	<ul style="list-style-type: none"> <li>• existing style sheet name</li> <li>• <a href="#">wk none</a> (default)</li> </ul>
	Specifies current style sheet for the selected element(s). Possible values:
wk style sheet	<ul style="list-style-type: none"> <li>• <a href="#">wk none</a> (default)</li> <li>• existing style sheet name</li> </ul>

# Printing 4D Write Pro documents

---

4D Write Pro documents can be printed in two ways:

- As parts of 4D forms
- As independent documents

## Printing documents in 4D forms

---

You can print 4D Write Pro embedded objects as part of any kind of 4D form (project, table, input, or output) using standard 4D printing commands such as **PRINT SELECTION** or **PRINT RECORD**.

The standard **Print Variable Frame** option is also supported(\*) for 4D Write Pro areas, allowing you to manage size during printing. When this option is checked, the margins (outside and inside) and top border are only applied to the first page, and the margins (outside and inside) and bottom border are only applied to the last page.

Pagination properties of the document are ignored: widow and orphan control is disabled and page breaks are not applied (these properties are only used for page rendering on screen, or for standalone printing of the document). When the **Print Variable Frame** option is selected, only objects located above the form area are printed. For more information about this option, refer to "**Print Variable Frame**" in the Design Reference manual.

(\*) The **Print object** and **Print form** commands are not compatible with this option.

## View mode for printing

Regardless of the **View mode** set for the 4D Write Pro area (see **Configuring View properties**), it is always printed as in the **Embedded** mode when you use a 4D printing command such as **Print form**. In this case, the following Appearance settings are not taken into account for the 4D Write Pro form objects: Page view mode (always "Embedded"), Show headers, Show footers, Show page frame (always "No"), Show hidden characters (always "No").

## Example

The following example shows the effect of the **Print Variable Frame** option on a 4D Write Pro area embedded in the default output form. The following code is executed:

```
ALL RECORDS ([Movies])
ORDER BY ([Movies]Title)
PRINT SELECTION ([Movies])
```

- Here is the result with the Print Variable Frame option **unchecked** (off):

**Mountains of the world**

<b>Name :</b>	<b>Continent :</b>
Everest	Asia

<b>Name :</b>	<b>Continent :</b>
Mont Blanc	Europe


<b>Name :</b>	<b>Continent :</b>
Mount Etna	Europe

<b>Name :</b>	<b>Continent :</b>
Mount Kilimanjaro	Africa

- Here is the result with the Print Variable Frame option **checked** (on):

**Mountains of the world**


<b>Name :</b>	<b>Continent :</b>
Everest	Asia



**Mount Everest**, also known in Nepal as Sagarmāthā and in Tibet as Chomolungma, is Earth's highest mountain. It is located in the Mahalangur mountain range in Nepal.[8][9] Its peak is 8,848 metres (29,029 ft) above sea level.[1] It is not the furthest summit from the centre of the Earth. That honour goes to Mount Chimborazo, in the Andes.[10] The international border between China and Nepal runs across Everest's precise summit point. Its massif includes neighbouring peaks Lhotse, 8,516 m (27,940 ft); Nuptse, 7,855 m (25,771 ft) and Changtse, 7,580 m (24,870 ft).

**Mountains of the world**

<b>Name :</b>	<b>Continent :</b>
Mont Blanc	Europe



Mont Blanc (French pronunciation: [mɔ̃ blɑ̃] or Monte Bianco (Italian pronunciation: [ˈmonte ˈbjaŋko]), both meaning "White Mountain", is the highest mountain in the Alps and the highest peak in Europe outside of the Caucasus range.[2] It rises 4,809 m (15,778 ft)[1][3] above sea level and is ranked 11th in the world in topographic prominence.[4]

(Sample text source: Wikipedia)

## Printing independent documents

---

Starting with 4D v15 R5, 4D Write Pro includes printing features allowing you to print independent 4D Write Pro documents as well as to control standard printing options such as the format, orientation, or page numbers.

### 4D Write Pro commands

Basically, two commands handle the 4D Write Pro printing features: **WP PRINT** and **WP USE PAGE SETUP**.

- **WP PRINT** launches a print job for a 4D Write Pro document or adds the document to a current print job.
- **WP USE PAGE SETUP** modifies the current printer page settings based on the 4D Write Pro document attributes for page size and orientation.

### Configuration notes:

- 4D Write Pro print commands are based on the new internal architecture designed for 64-bit versions of 4D. However, they are still supported in 32-bit versions, just not within print jobs launched with **OPEN PRINTING JOB** (see **WP PRINT**).
- On machines with Windows 7 or Windows Server 2008 R2, make sure that the *Platform Update for Windows 7* has been installed so that the printing features are supported.

### Regular 4D commands

The following 4D commands support 4D Write Pro printing features:

- **SET PRINT OPTION** and **GET PRINT OPTION**: All options are supported for 4D Write Pro documents printed by **WP PRINT**. For [Paper option](#) and [Orientation option](#), you may find it more efficient to call **WP USE PAGE SETUP** in order to easily synchronize these attributes with the 4D Write Pro document settings. The [Page range option](#) (15) allows you to specify the page range to print.
- **PRINT SETTINGS**: Defines print settings for the current printer; if **WP PRINT** is called afterwards, it takes any print settings modified by means of the Print Settings dialog boxes into account (except for margins, which are always based on the 4D Write Pro document).
- **OPEN PRINTING JOB** and **CLOSE PRINTING JOB**: **WP PRINT** can be called between these commands in order to insert one or more 4D Write Pro documents into a single print job.



PRINT SELECTION ( {aTable}{;}{\* | >} )

Parameter	Type	Description
aTable	Table	→ Table for which to print the selection, or Default table, if omitted
*   >	Operator	→ * to delete the printing dialog boxes, or > to not reinitialize print settings

## Description

**PRINT SELECTION** prints the current selection of *aTable*. The records are printed with the current output form of the table in the current process. **PRINT SELECTION** performs the same action as the Print menu command in the Design environment. If the selection is empty, **PRINT SELECTION** does nothing.

By default, **PRINT SELECTION** displays the printer dialog boxes (in 4D 32-bit versions) or the Print job dialog box (in 4D 64-bit versions) before printing. If the user cancels either of the printer dialog boxes, the command is canceled and the report is not printed.

You can delete these dialog boxes by using either the optional asterisk (\*) parameter or the optional "greater than" (>) parameter:

- The \* parameter causes a print job using the current print parameters (default parameters or those defined by the **PAGE SETUP** and/or **SET PRINT OPTION** commands).
- Furthermore, the > parameter causes a print job without reinitializing the current print parameters. This setting is useful for executing several successive calls to **PRINT SELECTION** (e.g., inside a loop) while maintaining previously set customized print parameters. For an example of the use of this parameter, refer to the **PRINT RECORD** command description.

During printing, the output form method and/or the form's object methods are executed depending on the events that are enabled for the form and objects using the Property List window in the Design environment, as well as on the events actually occurring:

- An [On Header](#) event is generated just before a header area is printed.
- An [On Printing Detail](#) event is generated just before a record is printed.
- An [On Printing Break](#) event is generated just before a break area is printed.
- An [On Printing Footer](#) event is generated just before a footer is printed.

You can check whether **PRINT SELECTION** is printing the first header by testing **Before selection** during an [On Header](#) event. You can also check for the last footer, by testing **End selection** during an [On Printing Footer](#) event. For more information, see the description of these commands, as well as those of **Form event** and **Level**.

To print a sorted selection with subtotals or breaks using **PRINT SELECTION**, you must first sort the selection. Then, in each Break area of the report, include a variable with an object method that assigns the subtotal to the variable. You can also use statistical and arithmetical functions like **Sum** and **Average** to assign values to variables. For more information, see the descriptions of **Subtotal**, **BREAK LEVEL** and **ACCUMULATE**.

**Warning:** Do not use the **PAGE BREAK** command with the **PRINT SELECTION** command. **PAGE BREAK** is to be used with the **Print form** command.

After a call to **PRINT SELECTION**, the OK variable is set to 1 if the printing has been completed. If the printing was interrupted, the OK variable is set to 0 (zero) (i.e., the user clicked Cancel in the printing dialog boxes).

**4D Server:** This command can be executed on 4D Server within the framework of a stored procedure. In this context:

- Make sure that no dialog box appears on the server machine (except for a specific requirement). To do this, it is necessary to call the command with the \* or > parameter.
- In the case of a problem concerning the printer (out of paper, printer disconnected, etc.), no error message is generated.

## Example

---

The following example selects all the records in the [People] table. It then uses the **DISPLAY SELECTION** command to display the records and allows the user to highlight the records to print. Finally, it uses the selected records with the **USE SET** command, and prints them with **PRINT SELECTION**:

```
ALL RECORDS ([People]) ` Select all records  
DISPLAY SELECTION ([People];*) ` Display the records  
USE SET ("UserSet") ` Use only records picked by user  
PRINT SELECTION ([People]) ` Print the records that the user picked
```

## Print Variable Frame

### Principles

---

The **Print Variable Frame** option is available for the following objects:

- Picture type fields and variables,
- Text type fields and variables,
- 4D Write Pro areas (option described in the [Using a 4D Write Pro area](#) section of the 4D Write Pro Reference manual).

This option is found in the "Print" theme of the Property List:



It is also available using the **OBJECT SET PRINT VARIABLE FRAME** and **OBJECT GET PRINT VARIABLE FRAME** commands.

**Note:** Subforms have a similar option. For more information about this, refer to "[Subform Printing](#)" in the [List subforms](#) section.

This property handles the print mode for objects whose size can vary from one record to another depending on their contents. These objects can be set to print with either a fixed or variable frame. Fixed frame objects print within the confines of the object as it was created on the form. Variable frame objects expand during printing to include the entire contents of the object.

Note that the width of objects printed as a variable size is not affected by this option (defined by the object properties); only the height varies automatically based on the contents of the object.

You cannot place more than one variable frame object side-by-side on a form. You can place non-variable frame objects on either side of an object that will be printed with a variable size provided that the variable frame object is at least one line longer than the object beside it and that all objects are aligned on the top. If this condition is not respected, the contents of the other fields will be repeated for every horizontal slice of the variable frame object.

In the context of output forms, you can only place variable size objects in Detail areas.

**Note:** The **Print object** and **Print form** commands do not support this option.

### Pictures

---

Pictures can be printed with either fixed or variable frames if their display format allows it. Only the following display formats allow printing with variable frames:

- Truncated (Centered)
- On Background
- Truncated (Non-centered)

For more information about these picture formats, refer to [Display formats](#).

- If you check the **Print Variable Frame** option, the picture will be printed at a height that takes its size into account. The picture frame will be extended during printing if necessary in order to display the entire picture.
- If you do not check this option, the picture will be printed at a fixed height (set in the form).

### Text

---

- If you check the **Print Variable Frame** option, the text will be printed at a height that takes its size into account. The text field will be extended automatically during printing so that all the text it contains will be printed.
- If you do not check this option, the text will be printed at a fixed height (set in the form).

## Print object

Print object ( { \* ; } object { ; posX { ; posY { ; width { ; height } } } } ) -> Function result

Parameter	Type	Description
*	Operator	➔ If specified, object is an object name (string) If omitted, object is a variable
object	Form object	➔ Object name (if * is specified) or Variable (if * is omitted)
posX	Longint	➔ Horizontal location of object
posY	Longint	➔ Vertical location of object
width	Longint	➔ Width of object (pixels)
height	Longint	➔ Height of object (pixels)
Function result	Boolean	➔ True = object entirely printed; otherwise False

### Description

---

The **Print object** command lets you print the form object(s) designated by the *object* and \* parameters, at the location set by the *posX* and *posY* parameters.

Before calling the **Print object** command, you must designate the table or project form containing the objects to be printed, using the **FORM LOAD** command.

If you pass the optional \* parameter, you indicate that the object parameter is an object name (character string). If you do not pass the \* parameter, you indicate that object is a variable. In this case, you pass a variable reference (object type only) instead of a string.

The *posX* and *posY* parameters specify the starting point for printing the object(s). These values must be expressed in pixels. If these parameters are omitted, the object will be printed according to its location in the form.

The *width* and *height* parameters are used to specify the width and height of the form object. The **Print object** command does not manage objects of variable size. You must use the **OBJECT GET BEST SIZE** command to manage the size of objects. You can also use the **OBJECT GET BEST SIZE** command to find out the most appropriate size for objects containing text. Similarly, **Print object** will not cause automatic page breaks. You must manage them according to your needs.

You can use 4D commands to modify object properties (color, size, etc.) on the fly.

The command returns True if the object has been completely printed and False if this is not the case; in other words, if it was not able to print all the data associated with the object within the set framework. Typically, the command returns False when printing a list box if all the rows of the list box could not be printed. In this case, you can simply call the **Print object** command repeatedly until it returns True: a specific mechanism automatically causes the contents of the object to scroll after each call.

#### Notes:

- In the current version of 4D, only list box type objects have this mechanism (the command always returns True for any other type of object). In forthcoming versions of 4D, this functioning will be extended to other objects with variable contents.
- The **LISTBOX GET PRINT INFORMATION** command lets you check the status of the printing during the operation.

The **Print object** command can only be used in the context of a print job opened beforehand with the **OPEN PRINTING JOB** command. If it is not called in this context, the command does nothing. Several **Print object** commands can be called in the same print job.

**Note:** Hierarchical lists, subforms and Web areas cannot be printed.

### Example 1

---

Example for printing ten objects in a form:

**PRINT SETTINGS**

```

If (OK=1)
  OPEN PRINTING JOB
  If (OK=1)
    FORM LOAD ("PrintForm")
    x:=100
    y:=50
    GET PRINTABLE AREA (hpaper;wpaper)
    For ($i;1;10)
      OBJECT GET BEST SIZE (*;"Obj"+String($i);bestwidth;bestheight)
      $send:=Print object(*;"Obj"+String($i))
      y:=y+bestheight+15
      If (y>hpaper)
        PAGE BREAK (>)
        y:=50
      End if
    End for
  End if
  CLOSE PRINTING JOB
End if

```

## Example 2

---

Example of printing a complete list box:

```

Repeat
  $send:=Print object(*;"mylistbox")
Until ($send)

```

## Filter expressions contained in a 4D Write Pro document

### Overview

---

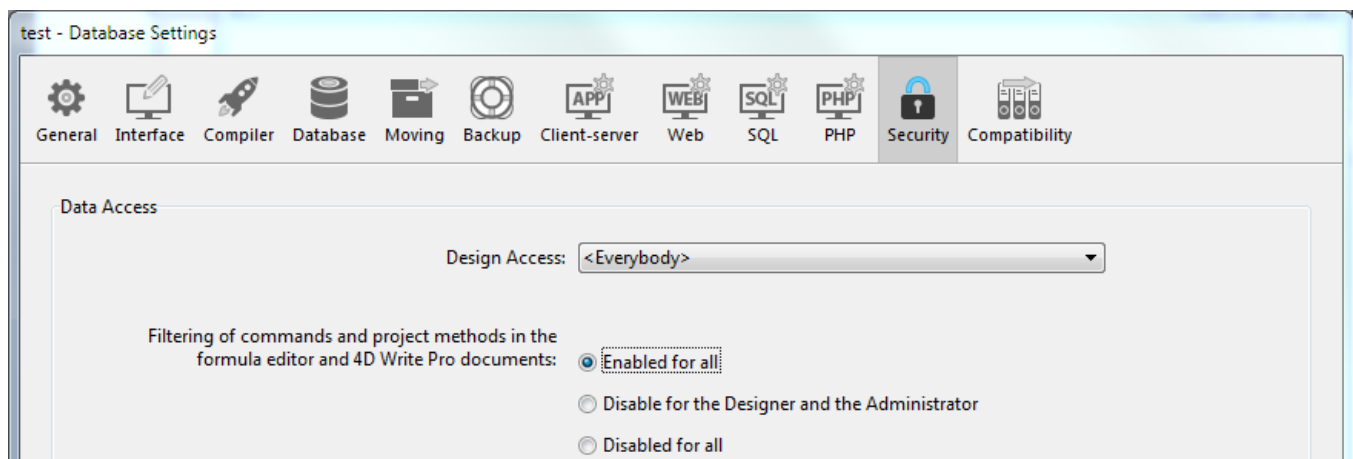
4D Write Pro documents can contain references to dynamic 4D expressions such as variables and fields, but also formulas, project methods, or 4D commands. These references are evaluated when the documents are displayed or printed. For security reasons, evaluation of dynamic expressions must be controlled by the developer to make sure that no inappropriate expression is used and that it will not generate any unexpected changes or side effects in the database. This prevents you from executing commands such as **DELETE SELECTION** or methods like "DeleteOrders".

**Note:** Expressions can be inserted using the **ST INSERT EXPRESSION** command, or by editing 4D Write Pro documents with an HTML editor.

### Support of standard 4D filtering feature

---

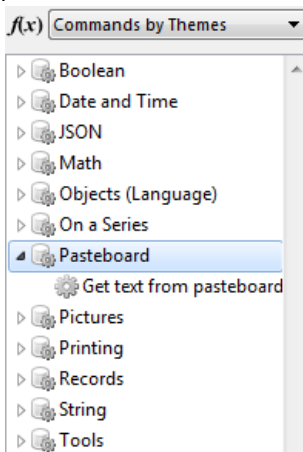
Starting with 4D v16, evaluation of 4D commands and project methods inserted in 4D Write Pro documents is now under the control of the regular 4D filtering option defined at the database level. This option is located on the **Security** page of the "Database Settings" dialog box:



By default in 4D, this option is **Enabled for all**, which means that commands and methods must be explicitly allowed; otherwise evaluation errors are returned. You can disable this filtering partially (for the Designer and the Administrator), or for all users. This option is used in the entire 4D database and controls all user formula evaluations. For more information about this option, please refer to the *Design Reference* manual.

When the option is enabled:

- users can only call commands belonging to the "formula compliant" list. This list is displayed in the right-hand part of the Formula editor:



- users cannot execute any project method. Project methods that you want to allow in 4D Write documents must be explicitly declared using the **SET ALLOWED METHODS** command.

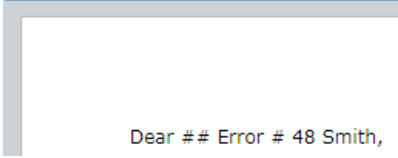
Whenever a "forbidden" 4D method or command is found during the evaluation of an expression in a 4D Write Pro document, the value is replaced by "#command\_5#command\_5 Error #command\_5 48".

## Example

You inserted the following expression in your 4D Write Pro document:

```
ST INSERT EXPRESSION (*;"WriteProArea";"Gender")
```

By default if the security option is checked, the **Gender** method will not be evaluated:

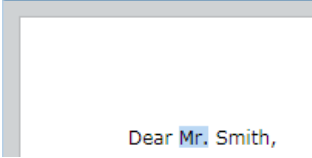


Dear ## Error # 48 Smith,

If you execute the following code:

```
ARRAY TEXT (aTallow;1)
aTallow{1}:="Gender"
SET ALLOWED METHODS (aTallow)
```

The **Gender** method will be evaluated and return a value:



Dear Mr. Smith,

## Compatibility

---

This filtering was not enabled for 4D Write Pro documents in previous releases. If your 4D Write Pro documents were referencing 4D methods, they will no longer be evaluated once the database has been converted to 4D v16 or higher. "#command\_5\_command\_5 Error #command\_5 48" messages will be displayed instead.

In this case, you need to add the methods to the approved list using the **SET ALLOWED METHODS** command.

# Importing 4D Write documents

---

One of the main features of the new 4D Write Pro object is the ability to import and convert existing 4D Write documents. This will allow you to migrate applications that currently rely on the 4D Write plug-in.

## Compatibility notes:

- Only 4D Write documents of the last generation ("4D Write v7") are supported.
- Copying-pasting from a 4D Write document to a 4D Write Pro area is not supported for the moment. A 4D Write document can only be imported using 4D Write Pro language commands.

## How to import a 4D Write document

---

4D Write Pro objects offer two ways to import 4D Write documents:

- For 4D Write files stored on disk, you use the **WP Import document** command,
- For 4D Write areas stored in BLOB fields, you use the **WP New** command.

For more information, please refer to the description of these commands.

## Which properties will be recovered from 4D Write?

---

To facilitate your migration from the 4D Write plug-in to 4D Write Pro, we want to support as many 4D Write features as possible in 4D Write Pro objects.

This paragraph lists the 4D Write plug-in properties that are currently recovered in a 4D Write Pro area after an import using the **WP Import document** or **WP New** commands.

Note however that a few differences can be seen, which are not considered as bugs. This is due, for example, to the default font used in 4D Write Pro for bullets, or small conversions in the Underline type.

## Document info

4D Write plug-in	4D Write Pro
Creation date & time	Available
Modification date & time	Available
Locked	Not available (use read-only object property)
Title	Available
Subject	Available (only plain text)
Author	Available
Company	Available
Notes	Available

## Document view parameters



<b>4D Write plug-in</b>	<b>4D Write Pro</b>
View page mode	Not imported (use Document/Page view mode in the context menu)
View rulers	Not available
View frames	Not available
View header	Not available
View footer	Not available
View first page header	Not available
View first page footer	Not available
View pictures	Not available
View HScrollbar	Not imported (use hor. scrollbar object property)
View VScrollbar	Not imported (use vert. scrollbar object property)
View invisible characters	Not available
View references	Not imported (use <b>ST SET OPTIONS</b> )
View column separators	Not available
View H Splitter	Not available
View V Splitter	Not available
View Wysiwyg	Not available
View zoom	Not imported (use Document/Zoom in the context menu)

## Document parameters

<b>4D Write plug-in</b>	<b>4D Write Pro</b>
Unit	Not available
Language	Not available
Count of columns	Not available
Column spacing	Not available
Widows & orphans	Not available
Default tab	Available
Leading tab	Not available
URL color	Not available
URL visited color	Not available

## Document pagination parameters

4D Write plug-in	4D Write Pro
Page width	Available
Page height	Available
First page number	Available (starting with v16)
First page header & footer are different	Available (starting with v16)
Left & right page header & footer are different	Available (starting with v16)
Page binding	Available (starting with v16)
Opposite pages	Available (starting with v16)
Page margins	Available
Header top margin	Available (starting with v16)
Header bottom margin	Available (starting with v16)
Footer top margin	Available (starting with v16)
Footer bottom margin	Available (starting with v16)
First page top margin	Available (starting with v16)
First page bottom margin	Available (starting with v16)
Header first page top margin	Available (starting with v16)
Header first page bottom margin	Available (starting with v16)
Footer first page top margin	Available (starting with v16)
Footer first page bottom margin	Available (starting with v16)
First page is right	Available (starting with v16)

## Document printing parameters

4D Write plug-in	4D Write Pro
Kind of paper	Not available
Landscape	Not available
Width	Not available
Height	Not available
User margins	Not available
Scale	Not available
X resolution	Not available
Y resolution	Not available

## Images

### Compatibility notes:

- 4D Write Pro does not yet support absolute positioning for images in pages. Only inline images are supported and imported.
- In the 64-bit version of Windows, importing 4D Write documents that contain images having the Mac OS PICT format is not supported. If you want to import documents containing images of this type, you will first need to convert them to another format, or use a 32-bit version of 4D. Keep in mind that the PICT format is obsolete and must no longer be used (see [Pictures in PICT format](#)).

<b>4D Write plug-in</b>	<b>4D Write Pro</b>
X (left)	(& position :absolute) (for images in page only)
Y (top)	(& position :absolute) (for images in page only)
Width	Available
Height	Available
Page number	Not available
Behind	Not available
Not in first page	Not available
Viewport mode (scale to fit, etc.)	Available
Is expression	Not available
Keep size	Not available

## Character properties

<b>4D Write plug-in</b>	<b>4D Write Pro (span properties)</b>
Italic	Available
Bold	Available
Strikeout	Available
Underline	Available
Shadow	Available
Exponent (superscript or subscript)	Available
Capitals (uppercase or small uppercase)	Available
Font Family	Available
Font Size	Available
Text Color	Available
Text Back Color	Available
Underline Color	Available
Strikeout Color	Available
Shadow color	Available
User property	Not available
Spell checking (syntax & grammar on or off)	Not available
Appearance	Not available
Style sheet	Not imported (styles are imported but style sheets are not available)

## Paragraph properties

<b>4D Write plug-in</b>	<b>4D Write Pro</b>
Justification	Available
Interline	Available
Bullet	Available
Left margin	Available
Right margin	Available
Text indent	Available
Border line style	Available
Border line color	Available
Border back color	Available
Left border	Available
Right border	Available
Top border & top inside border	Available
Bottom border & bottom inside border	Available
Border spacing	Available
Style Sheet	Available
Tabulations	Available

## Hyperlinks

<b>4D Write plug-in</b>	<b>4D Write Pro</b>
URL link	Available
4D method link	Not available
Open document link	Not available

## 4D expressions

<b>4D Write plug-in</b>	<b>4D Write Pro</b>
4D expression	Available
Date & Time	Available
HTML expression	Not available
RTF expression	Not available

## Text data

<b>4D Write plug-in</b>	<b>4D Write Pro</b>
Main text data	Available
Header text data	Not available
Footer text data	Not available

## WP Import document

WP Import document ( filePath ) -> Function result

Parameter	Type		Description
filePath	String	→	Path to a 4D Write document (.4w7 or .4wt) or a 4D Write Pro document (.4wp)
Function result	Object	↩	4D Write Pro object

### Description

---

The **WP Import document** command converts an existing 4D Write Pro or 4D Write document (.4wp, .4w7 or .4wt) to a new 4D Write Pro object.

In the *filePath* parameter, pass the path of a document stored on disk. The following types of documents are supported:

- former 4D Write documents (.4w7 or .4wt). For a detailed list of 4D Write features that are currently supported in 4D Write Pro objects, please refer to the [Importing 4D Write documents](#) section.
- 4D Write Pro (.4wp) format documents. For more information about the 4D Write Pro document format, refer to [.4wp document format](#).

You must pass a complete path, unless the document is located at the same level as the structure file, in which case you can just pass its name.

After execution, the command returns the 4D Write Pro object resulting from this conversion.

**Note:** If you want to import a document stored in a 4D BLOB field, you can also consider using the **WP New** command.

An error is returned if the *filePath* parameter is invalid, or if the file is missing or the file format is not supported.

### Example

---

```
C_OBJECT(WPDoc)
WPDoc:=WP Import document("C:\\documents\\4DWriteDocs\\Letter.4w7")
```

## ST SET OPTIONS

```
ST SET OPTIONS ( { * ; } object ; option ; value { ; option2 ; value2 ; ... ; optionN ; valueN } )
```

Parameter	Type	Description
*	Operator	→ If specified, object is an object name (string) If omitted, object is a field or variable
object	Form object	→ Object name (if * is specified) or Field or variable (if * is omitted)
option	Longint	→ Option to set
value	Longint	→ New value of option

### Description

The **ST SET OPTIONS** command modifies one or more operating options for the styled text field or variable designated by the *object* parameter.

Passing the optional \* parameter indicates that the *object* parameter is an object name (string). If you do not pass this parameter, it indicates that the *object* parameter is a field or variable. In this case, you pass a field or variable reference instead of a string (field or variable object only).

Pass the code of the option to modify in *option* and its new value in *value*.

The *option* parameter supports the following constant found in the "Multistyle Text" theme:

Constant	Type	Value	Comment
ST Expressions display mode	Longint	1	The <i>value</i> parameter can contain <a href="#">ST Values</a> or <a href="#">ST References</a>

In the *value* parameter, you can pass one of the following constants:

Constant	Type	Value	Comment
ST References	Longint	1	Display source strings of expressions
ST Values	Longint	0	Display computed values of expressions

Display of values:

Current time:	14:39:10
Field contents:	Bravo

Display of expressions:






























Current time:	String(Current time)
Field contents:	[Table_1]Comment

### Example

The following code lets you switch the display mode of the area:

```
ST GET OPTIONS (*;"StyledText_t";ST Expressions display mode;$exprValue)
If ($exprValue=1)
  ST SET OPTIONS (*;"StyledText_t";ST Expressions display mode;ST Values)
Else
  ST SET OPTIONS (*;"StyledText_t";ST Expressions display mode;ST References)
End if
```

# 4D Write Pro Language

-  About 4D Write Pro objects
-  Using commands from the Objects (Forms) theme
-  Using commands from the Styled Text theme
-  Accessing document contents by programming
-  WP CREATE BOOKMARK New 16.0
-  WP DELETE BOOKMARK New 16.0
-  WP EXPORT DOCUMENT
-  WP EXPORT VARIABLE
-  WP GET ATTRIBUTES
-  WP Get bookmark range New 16.0
-  WP GET BOOKMARKS New 16.0
-  WP Get page count New 16.0
-  WP Get paragraphs
-  WP Get pictures
-  WP Get range
-  WP Get selection
-  WP Import document
-  WP INSERT BREAK New 16.0
-  WP INSERT DOCUMENT New 16.0
-  WP INSERT PICTURE New 16.0
-  WP Is font style supported
-  WP New Updated 16.0
-  WP PRINT Updated 16.0
-  WP RESET ATTRIBUTES
-  WP SELECT
-  WP SET ATTRIBUTES
-  WP USE PAGE SETUP
-  4D Write Pro Constants
-  4D Write Pro Attributes

## About 4D Write Pro objects

### 4D Write Pro objects

---

In 4D applications, 4D Write Pro documents are handled using *Object* type variables (for more information, refer to the **C\_OBJECT** command).

Once referenced in memory, a 4D Write Pro object can be:

- displayed and modified through a 4D form using the form object named "Write Pro Area," provided that the form area has the same name as the object variable (see **Defining a 4D Write Pro area**).
- handled programmatically by commands from several themes (see below).
- exported to an HTML document on disk using the **WP EXPORT DOCUMENT** command.

### Commands that handle 4D Write Pro objects

---

You can handle 4D Write Pro objects using 4D commands from several themes:

- Dedicated commands from the **4D Write Pro** theme; these commands are described in this chapter.
- **Objects (Forms)** commands, to address formatting and object property features; for more information, please refer to the **Using commands from the Objects (Forms) theme** section.
- **Styled Text** commands, to address content features; for more information, please refer to the **Using commands from the Styled Text theme** section.



## Using commands from the Objects (Forms) theme

---

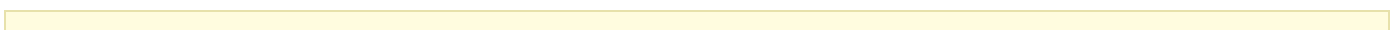
The following commands support 4D Write Pro form objects:

Command	Comments
<b>OBJECT DUPLICATE</b>	
<b>OBJECT Get auto spellcheck/OBJECT SET AUTO SPELLCHECK</b>	
<b>OBJECT Get border style/OBJECT SET BORDER STYLE</b>	
<b>OBJECT Get context menu/OBJECT SET CONTEXT MENU</b>	
<b>OBJECT GET COORDINATES/OBJECT SET COORDINATES</b>	
<b>OBJECT Get data source/OBJECT SET DATA SOURCE</b>	
<b>OBJECT GET DRAG AND DROP OPTIONS/OBJECT SET DRAG AND DROP OPTIONS</b>	
<b>OBJECT Get enabled/OBJECT SET ENABLED</b>	
<b>OBJECT Get enterable/OBJECT SET ENTERABLE</b>	
<b>OBJECT GET EVENTS/OBJECT SET EVENTS</b>	
<b>OBJECT Get focus rectangle invisible/OBJECT SET FOCUS RECTANGLE INVISIBLE</b>	
<b>OBJECT Get font/OBJECT SET FONT</b>	Applied to current selection (if any)
<b>OBJECT Get font size/OBJECT SET FONT SIZE</b>	Applied to current selection (if any)
<b>OBJECT Get font style/OBJECT SET FONT STYLE</b>	Applied to current selection (if any)
<b>OBJECT Get horizontal alignment/OBJECT SET HORIZONTAL ALIGNMENT</b>	Applied to current selection (if any). Support of the <a href="#">wk_justify</a> constant for 4D Write Pro areas
<b>OBJECT GET RESIZING OPTIONS/OBJECT SET RESIZING OPTIONS</b>	
<b>OBJECT SET COLOR</b>	Applied to current selection (if any)
<b>OBJECT GET RGB COLORS/OBJECT SET RGB COLORS</b>	Applied to current selection (if any)
<b>OBJECT Get type</b>	
<b>OBJECT Get vertical alignment/OBJECT SET VERTICAL ALIGNMENT</b>	Vertical alignment of paragraphs: only has an effect when paragraph height is greater than paragraph text height
<b>OBJECT Get visible/OBJECT SET VISIBLE</b>	
<b>OBJECT Is styled text</b>	Returns true
<b>OBJECT MOVE</b>	
<b>OBJECT GET SCROLL POSITION/OBJECT SET SCROLL POSITION</b>	
<b>OBJECT GET SUBFORM CONTAINER SIZE</b>	
<b>OBJECT Get name</b>	
<b>OBJECT Get pointer</b>	

Any OBJECT commands not listed above are not applicable to 4D Write Pro areas.

## Example

You want to set the horizontal alignment of a 4D Write Pro area to the center:



**Case of**

: (**Form event=On Clicked**)

**OBJECT SET HORIZONTAL ALIGNMENT**(\*;"myObject";Align center)

hAlignLeft:=0 //store property for next display

hAlignRight:=0

**End case**

## Using commands from the Styled Text theme

---

The following commands support 4D Write Pro objects:

Command	Comments
<b>ST COMPUTE EXPRESSIONS</b>	
<b>ST FREEZE EXPRESSIONS</b>	
<b>ST GET ATTRIBUTES/ST SET ATTRIBUTES</b>	
<b>ST Get content type</b>	A new type (6) has been added for the image content type
<b>ST Get expression / ST INSERT EXPRESSION</b>	Starting with 4D v16, expressions are filtered by default in 4D Write Pro documents and must be explicitly allowed. For more information, refer to the <a href="#">Filter expressions contained in a 4D Write Pro document</a> section.
<b>ST GET OPTIONS/ST SET OPTIONS</b>	
<b>ST Get plain text/ST SET PLAIN TEXT</b>	
<b>ST Get text / ST SET TEXT</b>	
<b>ST GET URL / ST INSERT URL</b>	

### Example

---

You want to replace the selection in a 4D Write Pro area with the contents of a variable:

```
C_TEXT(fullName)

Case of
  : (Form event=On_Clicked)
    ST INSERT EXPRESSION(myArea;"fullName";ST Start highlight;ST End highlight)
End case
```

### Inserting document and page expressions

---

You can insert special expressions related to document attributes or page attributes using the **ST INSERT EXPRESSION** command.

Expression syntax	Availability	Type	Description
\$wp_title	all parts in the document	Text	Title defined in <a href="#">wk_title</a> attribute
\$wp_author	all parts in the document	Text	Author defined in <a href="#">wk_author</a> attribute
\$wp_subject	all parts in the document	Text	Subject defined in <a href="#">wk_subject</a> attribute
\$wp_company	all parts in the document	Text	Company defined in <a href="#">wk_company</a> attribute
\$wp_notes	all parts in the document	Text	Notes defined in <a href="#">wk_notes</a> attribute
\$wp_dateCreation	all parts in the document	Date	Date creation defined in <a href="#">wk_date_creation</a> attribute
\$wp_dateModified	all parts in the document	Date	Date modified defined in <a href="#">wk_date_modified</a> attribute
\$wp_pageNumber	Header & footer - error everywhere else	Longint	Page number as it is defined: from the document start (default) or from the section page start if it is defined by section page start
\$wp_pageCount	Header & footer - error everywhere else	LongInt	Page count : total count of pages

To insert an expression, make sure the cursor is located in the appropriate area (header, footer, or document body) and call the **ST INSERT EXPRESSION** command. For example, to insert the page number in the selected footer area:

```
ST INSERT EXPRESSION (*;"4DWPArea";"$wp_pageNumber")
```

The following document design can be defined, for example:

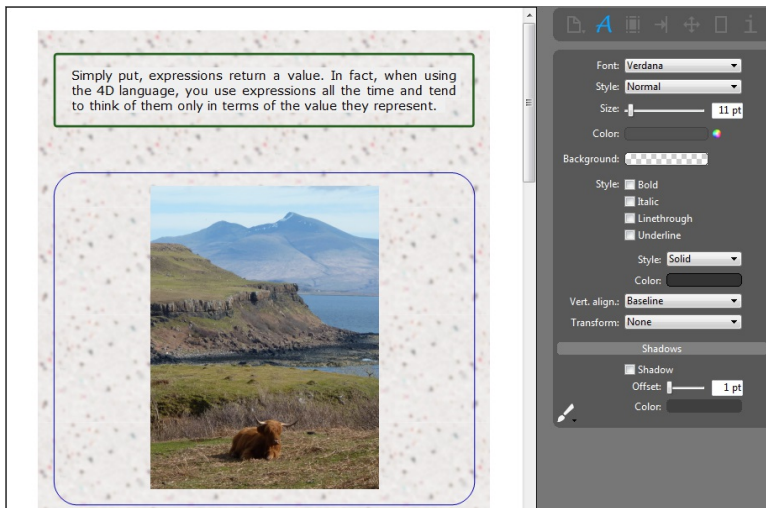
The image shows a document layout with the following elements:

- First header:** 4D Write Pro New Document
- Section Header:** First Section (new name) Header (First page)
- First footer:** Page 1 / 10
- Second header:** 10/05/16 2 17:08:18 Page 2 / 10

## Accessing document contents by programming

4D Write Pro provides a full set of commands allowing you to modify text and image attributes in your documents. Thanks to these features, 4D developers can design their own user interfaces (using buttons, menus, check boxes, etc.) for 4D Write Pro documents. Commands can be applied to whole documents or to specific portions (ranges), either based on user selections or custom values. Available properties include document units, margins, padding, background, paragraph settings, colors, fonts, font styles, as well as image properties.

For example, the **4D Write Pro area** library object makes extensive use of these features to provide a sophisticated 4D Write Pro interface:



## Selection range commands

Several commands are dedicated to handling selections in the documents. Since selected text can contain (invisible) formatting tags, 4D Write Pro works with *ranges*. A range is an object that represents a portion of a 4D Write document.

- **WP Get range**( *wpArea ; startRange ; endRange* ) -> *rangeObj*: returns a new range corresponding to boundaries you passed as parameters.
- **WP Get selection**( {\*;} *wpArea* ) -> *rangeObj*: returns a new range corresponding to the current user selection.
- **WP Get pictures**( *rangeObj* ) -> *rangeObj*: returns a new range containing only the pictures.
- **WP Get paragraphs**( *rangeObj* ) -> *rangeObj*: returns a new range containing only the paragraphs.
- **WP SELECT**( {\*;} *wpArea* {; *rangeObj*} {; *startRange ; endRange*} ): selects the text corresponding to the range.

## Bookmark commands

4D Write Pro allows you to create and work with dynamic references to parts of your documents, called **bookmarks**. A bookmark is a named reference attached to a specific range in a 4D Write Pro document.

Bookmarks are dynamic, which means that if the user moves, adds or removes text belonging to the bookmark, the associated range will be updated automatically and the bookmark will continue to reference the same content within the document. For example:

- You create a bookmark named "MyBM" that references the "Hello world" text on page 20 of your document.
- Then you insert 50 pages at the beginning of the document.
- You will still be able to access the same "Hello world" text automatically, now on page 70 of the document, by means of the "MyBM" bookmark.

A document can contain an unlimited number of bookmarks. Several bookmarks can reference the same range, and bookmark ranges can be interleaved. However, each bookmark name must be unique in the document. Bookmarks are not imported when using the **MissingRef** or **WP New** command (bookmarks in the destination document cannot be overwritten).

Once created, a bookmark is stored within the document. It is saved with the document, and can be handled by several different commands. It can also be used to reference parts of a template document. These parts can then be assembled automatically with data from the database to produce dynamic output documents such as invoices or catalogs.

Several commands allow you to create, remove, and use bookmarks:

- **WP CREATE BOOKMARK** to create a new bookmark from a range,
- to get all bookmarks defined in a document,
- **WP Get bookmark range** to retrieve a range from an existing bookmark,
- **WP DELETE BOOKMARK** to delete a bookmark.

## Attribute handling commands

---

The following commands can get or set any attributes of the document:

- **WP SET ATTRIBUTES**( *rangeObj* | *wpDoc* ; *attribName* ; *attribValue* { ; *attribName2* ; *attribValue2* ; ... ; *attribNameN* ; *attribValueN* } ): sets one or more attribute/value pairs in the document or range.
- **WP GET ATTRIBUTES**( *rangeObj* | *wpDoc* ; *attribName* ; *attribValue* { ; *attribName2* ; *attribValue2* ; ... ; *attribNameN* ; *attribValueN* } ): gets the current value of attributes in the document or range.
- **WP RESET ATTRIBUTES**( *rangeObj* ; *attribName* { ; *attribName2* ; ... ; *attribNameN* } ): resets attribute values in the document or range.

Attributes are detailed in the **4D Write Pro Attributes** section.

## Font handling command

---

This command allows you to get information about style support for a range:

- **WP Is font style supported**( *rangeObj* ; *wpFontStyle* ) -> *true* or *false*: allows you to know if a range supports a given style (useful to design an interface).

## WP CREATE BOOKMARK

WP CREATE BOOKMARK ( rangeObj ; bkName )

Parameter	Type		Description
rangeObj	Object	→	4D Write Pro range
bkName	String	→	Name of bookmark to create

### Description

---

The **WP CREATE BOOKMARK** command creates a new bookmark named *bkName* based upon the 4D Write Pro *rangeObj* in the range's parent document.

Bookmarks are named references to ranges, which allow you to access and reuse specific parts of the document, for example for templating purposes. For more information, please refer to the section.

In *bkName*, pass the name for the new bookmark. A bookmark name must be compliant with HTML/CSS names, i.e. it must only contain alphanumeric characters (invalid characters, such as space characters, are automatically removed). Bookmark names must be unique within the document. If a bookmark with the same name already exists in the document, it is overwritten.

You can create as many bookmarks as you want within the same document. Multiple bookmarks can be created using the exact same range. Once created, a bookmark is automatically stored in the parent document and is saved with the document itself.

### Example 1

---

You want to create a new bookmark referencing the currently selected text in the document. You can write:

```
C_OBJECT($range)
$range:=WP Get selection(*;"WPDocument")
WP CREATE BOOKMARK($range;"my_bookmark")
```

### Example 2

---

You want to rename an existing bookmark. To do this, you need to create a new bookmark with the same range, and then delete the old one:

```
C_TEXT($bookmarkOldName)
C_TEXT($bookmarkNewName)
C_LONGINT($p)
C_OBJECT($wpRange)

$bookmarkOldName:="MyBookmark"
$bookmarkNewName:="MyNewBookmark"

ARRAY TEXT($_bookmarks;0)
WP GET BOOKMARKS(WParea;$_bookmarks)

$p:=Find in array($_bookmarks;$bookmarkOldName)
If($p>0)
    $wpRange:=WP Get bookmark range(WParea;$bookmarkOldName)
    WP DELETE BOOKMARK(WParea;$bookmarkOldName)
    WP CREATE BOOKMARK($wpRange;$bookmarkNewName)
End if
```



## WP DELETE BOOKMARK

WP DELETE BOOKMARK ( wpDoc ; bkName )

Parameter	Type		Description
wpDoc	Object	→	4D Write Pro document
bkName	String	→	Name of bookmark to delete

### Description

---

The **WP DELETE BOOKMARK** command removes the bookmark named *bkName* from *wpDoc*.  
If the *bkName* bookmark does not exist in *wpDoc*, the command does nothing.

### Example

---

You want to rename an existing bookmark. To do this, you need to create a new bookmark with the same range, and then delete the old one:

```
C_TEXT ($bookmarkOldName)
C_TEXT ($bookmarkNewName)
C_LONGINT ($p)
C_OBJECT ($wpRange)

$bookmarkOldName:="MyBookmark"
$bookmarkNewName:="MyNewBookmark"

ARRAY TEXT ($_bookmarks;0)
WP GET BOOKMARKS (WParea;$_bookmarks)

$p:=Find in array ($_bookmarks;$bookmarkOldName)
If ($p>0)
    $wpRange:=WP Get bookmark range (WParea;$bookmarkOldName)
    WP DELETE BOOKMARK (WParea;$bookmarkOldName)
    WP CREATE BOOKMARK ($wpRange;$bookmarkNewName)
End if
```

## WP EXPORT VARIABLE

WP EXPORT VARIABLE ( wpDoc ; destination ; format {; option} )

Parameter	Type		Description
wpDoc	Object	→	4D Write Pro variable
destination	Text variable, BLOB variable	←	Variable to receive exported contents
format	Longint	→	Variable output format
option	Longint, String	→	Export options

### Description

The **WP EXPORT VARIABLE** command exports the *wpDoc* 4D Write Pro object to the 4D *destination* variable in the specified *format*.

In *wpDoc*, pass the 4D Write Pro object that you want to export.

In *destination*, pass the variable that you want to fill with the exported 4D Write Pro object. The type of this variable depends on the export format specified in the *format* parameter:

- If you pass the native .4wp format, the variable will be of the Blob type,
- If you pass an HTML format, the variable will be of the Text type.

In the *format* parameter, pass a constant from the **4D Write Pro Constants** theme to set the export format you want to use. Each format is related to a specific use. The following formats are supported:

Constant	Type	Value	Comment
wk 4wp	Longint	4	The 4D Write Pro document is saved in a native archive format (zipped HTML and images saved in a separate folder). 4D specific tags are included and 4D expressions are not computed. This format is particularly suitable for saving and archiving 4D Write Pro documents on disk without any loss.
wk mime html	Longint	1	4D Write Pro document is saved as standard MIME HTML with HTML documents and images embedded as MIME parts (encoded in base64). Expressions are computed and 4D specific tags are removed. This format is particularly suitable for sending HTML emails with the <b>SMTP_QuickSend</b> command.
wk web page html 4D	Longint	3	4D Write Pro document is saved as HTML and includes 4D specific tags; each expression is inserted as a non-breaking space. Since this format is lossless, it is appropriate for storing purposes in a text field.

### Notes:

- "4D specific tags" means 4D XHTML with a 4D namespace and 4D CSS styles.
- Expressions can be frozen at any time before export using **ST FREEZE EXPRESSIONS**.
- For more information about the 4D Write Pro document format, refer to **.4wp document format**.

In the *options* parameter, you pass options that will configure the export. You can pass either:

- a *longint* value to define the style of the HTML code; the following constants are available:

Constant	Type	Value	Comment
wk html debug	Longint	1	Formatted HTML code ("pretty print"), easier to debug
wk normal	Longint	0	Standard HTML code

- *HTML debug option off (default):*

```
<html xmlns="http://www.w3.org/1999/xhtml"><head><title>New 4D Write Pro Document</title><style
type="text/css">body { background-color:#FFFFFF }ul, ol { margin:0;padding:0 }p,li { white-space:pre-
wrap;margin:0pt;padding:0pt;font-family:'Times New Roman' }p.Normal,li.Normal { text-align:left }p._p1,li._p1
{ font-family:'Arial';font-size:18pt;color:#1D1D1D }img._img1 { width:51pt;height:51pt }</style></head>
<body><p class="Normal_p1"><span>23/12/14</span></p><p class="Normal_p1">Dear <span
style="color:#BEE0E0">Shaun Stoltz</span><span style="color:#800062"> <span style="color:#000000">from
```

- o *HTML debug option on:*

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<title>New 4D Write Pro Document</title>
<style type="text/css">
body { background-color:#FFFFFF }
ul, ol { margin:0;padding:0 }
```

- or a *string*. The following property is supported (only when the [wk mime html](#) format is used):
  - o CID host domain name: host domain that will added to generated CID URLs including an '@' as separator. For instance, if you pass "gmail.com", '123@gmail.com' will be inserted if the CID unique ID is 123. By default if omitted, only the CID unique ID is used (accepted by most mail servers).

## Example

You want to send an email containing styled text, 4D references and images. You can use a 4D Write Pro area exported in MIME HTML format and sent using 4D Internet Commands:

```
C_LONGINT($smtpid_t;$err_t;$smtpOption_t;$smtpPort_t)
C_TEXT($str;$emailBody_t;$smtpHost_t;$emailTo_t;$emailFrom_t;$smtpPass_t)

//export area in appropriate format
WP_EXPORT VARIABLE(myWPArea;$str;wk_mime_html)
$emailTo_t:="johnsmith@4d.com"
$emailFrom_t:"testWritePro@gmail.com"
$emailBody_t:=$str
$smtpHost_t:"smtp.gmail.com"
$smtpOption_t:=9
$smtpPort_t:=465
$smtpPass_t:"QRN_on_bretzelburg"

$err_t:=SMTP_QuickSend($smtpHost_t;$emailFrom_t;$emailTo_t;$emailTitle_t;\
$emailBody_t;$smtpOption_t;$smtpPort_t;$smtpUser_t;$smtpPass_t)

If(($err_t=0))
ALERT("Email sent to "+emailTo_t)
Else
ALERT("Error in parameters, please try again.")
End if
```

## WP GET ATTRIBUTES

```
WP GET ATTRIBUTES ( rangeObj | wpDoc ; attribName ; attribValue { ; attribName2 ; attribValue2 ; ... ;  
attribNameN ; attribValueN } )
```

Parameter	Type	Description
rangeObj   wpDoc	Object	→ 4D Write Pro range or document
attribName	String	→ Name of attribute to get
attribValue	String, Real, Boolean, Array	← Current value of attribute for text range

### Description

---

The **WP GET ATTRIBUTES** command returns the value of any attribute in a 4D Write Pro range or document. This command gives you access to any kind of 4D Write Pro internal attributes: character, paragraph, document, or image.

In the first parameter, you can pass either a 4D Write Pro range object (*rangeObj*) or a 4D Write document reference (*wpDoc*). A *rangeObj* is a part of a 4D Write Pro document that can be created by different commands. The following table provides the scope of the **WP GET ATTRIBUTES** command depending on the target object parameter and the attributes:

Parameter	Common attributes (except "verticalAlign")	Document-only attributes	Paragraph-only attributes	Character-only attributes (and "verticalAlign")
<i>rangeObj</i> from <b>WP Get paragraphs</b>	paragraphs	document	paragraphs	paragraphs
<i>rangeObj</i> from <b>WP Get pictures</b>	images	document	-	-
<i>rangeObj</i> from <b>WP Get range</b> or <b>WP Get selection</b>	paragraphs for the range	document	paragraphs for the range (intersecting paragraphs)	characters for the range
<i>wpDoc</i>	document	document	paragraphs for all the document	characters for all the document

Common attributes (such as "margin", "padding", etc.) mean common to the document, paragraphs, and/or images.

For a comprehensive list of attributes to pass in *attribName*, as well as their respective values, please refer to the **4D Write Pro Attributes** section.

If there are different values for the same attribute in the range or document passed as a parameter, the command returns:

- for numerical values, wk mixed
- for an array, an empty array (tab stops, color if *attribValue* is defined as array), with an exception for wk text shadow offset for which the array value will always contain 2 entries which may be set separately to wk mixed if either horizontal offset or vertical offset (or both) are mixed.
- for string values, an empty string
- for picture values, an empty picture.

### Example

---

You want to get the background color of the selected area:

```
$range:=WP Get selection(*;"WParea")  
WP GET ATTRIBUTES($range;wk_background_color;$bcol)
```

## WP Get bookmark range

WP Get bookmark range ( wpDoc ; bkName ) -> Function result

Parameter	Type		Description
wpDoc	Object	→	4D Write Pro document
bkName	Text	→	Name of bookmark whose range you want to get
Function result	Object	↩	Range of bookmark

### Description

---

The **WP Get bookmark range** command returns a text range object (*rangeObj*) containing the range for the bookmark with the specified *bkName* in *wpDoc*.

If the *bkName* bookmark does not exist in *wpDoc*, an empty *rangeObj* object is returned.

### Example

---

You want to show the range of the "MyBookmark" bookmark in your document:

```
C_OBJECT($wpRange)
$wpRange:=WP Get bookmark range(WParea;"MyBookmark")
WP SELECT(WParea;$wpRange)
```

## WP GET BOOKMARKS

WP GET BOOKMARKS ( wpDoc ; arrBKNames )

Parameter	Type		Description
wpDoc	Object	⇒	4D Write Pro document
arrBKNames	Text array	⇐	Array of bookmark names

### Description

---

The **WP GET BOOKMARKS** command returns an array containing the names of all bookmarks defined in *wpDoc*. After the command is executed, the *arrBKNames* is filled with all the bookmark names in the document. In the array, names are sorted by bookmark position inside the document. If several bookmarks begin at the same position, they are sorted in alphabetical order.

### Example

---

You want to know the number of bookmarks defined in your document:

```
ARRAY TEXT($_bookmarks;0)
WP GET BOOKMARKS(WParea;$_bookmarks)
ALERT("The document contains "+Size of array($_bookmarks)+" bookmarks.")
```

## WP Get page count

WP Get page count ( wpDoc ) -> Function result

Parameter	Type		Description
wpDoc	Object	→	4D Write Pro document
Function result	Longint	↩	Number of pages in document

### Description

---

The **WP Get page count** command returns the total number of pages defined in the *wpDoc* 4D Write Pro document.

### Example

---

You want to know the total number of 4D Write Pro document pages stored in the "Manual" field within the current selection of items. You can write:

```
C_LONGINT($pageCount)
C_LONGINT($totalCount)
FIRST RECORD([Items])
While(Not(End selection([Items])))
    $pageCount:=WP Get page count([Items]Manual)
    $totalCount:=$totalCount+$pageCount
NEXT RECORD([Items])
End while
ALERT("Total number of manual pages: "+String($totalCount))
```

## WP Get paragraphs

WP Get paragraphs ( rangeObj ) -> Function result

Parameter	Type		Description
rangeObj	Object	→	Range from which to get paragraphs
Function result	Object	↩	Range addressing only paragraphs

### Description

---

The **WP Get paragraphs** command returns a specific range object that addresses only the paragraphs contained in the *rangeObj* you passed as parameter. The paragraph range object returned must be used by **WP GET ATTRIBUTES** and **WP SET ATTRIBUTES** to handle paragraph attributes only.

In *rangeObj*, pass a valid 4D Write Pro standard range object. A *rangeObj* is a part of a 4D Write Pro document; it can be created by the **WP Get selection** or **WP Get range** command.

### Example

---

You want to define padding for the paragraphs only:

```
$oParagraphs:=WP Get paragraphs($oSelection)  
WP SET ATTRIBUTES($oParagraphs;wk_padding;20)
```



## WP Get pictures

WP Get pictures ( rangeObj ) -> Function result

Parameter	Type		Description
rangeObj	Object	→	Range from which to get pictures
Function result	Object	↩	Range object containing pictures only

### Description

---

The **WP Get pictures** command returns a specific range object that addresses only the pictures contained in the *rangeObj* you passed as parameter. The image range object returned must be used by **WP GET ATTRIBUTES** and **WP SET ATTRIBUTES** to handle picture attributes only.

In *rangeObj*, pass a valid 4D Write Pro standard range object. A *rangeObj* is a part of a 4D Write Pro document; it can be created by the **WP Get selection** or **WP Get range** command.

### Example

---

You want to change the border color of pictures only:

```
$oPicts:=WP Get pictures($oSelection)
WP SET ATTRIBUTES($oPicts;wk border color;"blue")
```

## WP Get range

WP Get range ( wpArea ; startRange ; endRange ) -> Function result

Parameter	Type		Description
wpArea	Object	→	4D Write Pro object variable or field
startRange	Longint	→	Starting offset of range in the area
endRange	Longint	→	Ending offset of range in the area
Function result	Object	↪	Range object

### Description

The **WP Get range** command returns a new text range object (*rangeObj*) containing the selection between *startRange* and *endRange* in the *wpArea* 4D Write Pro area.

Pass a 4D Write Pro object variable or field in *wpArea*. If no valid 4D Write Pro area is passed in the *wpArea* parameter, an empty *rangeObj* is returned.

In *startRange* and *endRange*, pass values corresponding to the position of the first and last characters to select in the document. You can pass wk start text in *startRange* to define the beginning of the document, and wk end text in *endRange* to define the end of the document. Keep in mind that a 4D Write Pro document not only contains visible text but also formatting tags that are included in the range.

The command returns a new *rangeObj*. A *rangeObj* is a 4D Write Pro text range object which can be used to handle attributes on a text selection (with the new **WP GET ATTRIBUTES** and **WP SET ATTRIBUTES** commands). It contains 3 private read-only attributes (*wk range start*, *wk range end* and *wk range owner*) that are used to defined the range itself.

### Example

You want to select a range of 12 characters starting from the beginning of the 4D Write Pro field. The field is displayed in a form object:

#### Description

The DIALOG command presents the form form to the user. This command is often used to get information from the user through the use of variables, or to present information to the user, such as options for performing an operation. It is common to display the form inside a modal window created with the Open window command.

If you execute:

```
$range2:=WP Get range([SAMPLE]WP;wk_start_text;12)
WP SELECT(*;"WParea";$range2)
```

...the result is:

#### Description

The DIALOG command presents the form form to the user. This command is often used to get information from the user through the use of variables, or to present information to the user, such as options for performing an operation. It is common to display the form inside a modal window created with the Open window command.

## WP Get selection

WP Get selection ( { \* ; } wpArea ) -> Function result

Parameter	Type	Description
*	Operator	➔ If specified, wpArea is a form object name (string). If omitted, wpArea is an object field or variable.
wpArea	String, Object	➔ Form object name (if * is specified) or 4D Write Pro object variable or field (if * is omitted)
Function result	Object	➔ Range object

### Description

---

The **WP Get selection** command returns a new text range object (*rangeObj*) based upon the currently selected text in the *wpArea* 4D Write Pro area.

If you pass the optional \* parameter, you indicate that *wpArea* is a form object name (string). If you do not pass this parameter, you indicate that *wpArea* is a 4D Write Pro object variable or field. If no valid 4D Write Pro area is passed in the *wpArea* parameter, an empty *rangeObj* is returned.

This command returns a new *rangeObj*. A *rangeObj* is a 4D Write Pro text range object which can be used to handle attributes on a text selection (with the new **WP GET ATTRIBUTES** and **WP SET ATTRIBUTES** commands). It contains 3 private read-only attributes (*wk range start*, *wk range end* and *wk range owner*) that are used to defined the range itself.

### Example

---

You want to get the selected text from a 4D Write Pro area:

```
$range:=WP Get selection (*;"WParea")
```

WP INSERT BREAK ( rangeObj ; breakType ; mode {; rangeUpdate} )

Parameter	Type		Description
rangeObj	Object	➔	4D Write Pro range object
breakType	Longint	➔	Type of break to insert
mode	Longint	➔	Insertion mode
rangeUpdate	Longint	➔	Range update mode

## Description

The **WP INSERT BREAK** command inserts a new break of the *breakType* type in the *rangeObj* range according to the specified insertion *mode* and *rangeUpdate* parameter.

In *rangeObj*, pass a valid 4D Write Pro standard range object. A *rangeObj* is a part of a 4D Write Pro document; it can be created using the **WP Get selection**, **WP Get bookmark range** or **WP Get range** commands.

In *breakType*, pass one of the following constants from the **4D Write Pro Constants** theme to define the type of break to insert:

Constant	Type	Value	Comment
wk line break	Longint	0	Line break (in the same paragraph)
wk page break	Longint	2	Page break: defines a new page
wk section break	Longint	1	Section break: defines a new section

In the *mode* parameter, pass a constant to indicate the insertion mode to be used for the break in the destination *rangeObj* range:

Constant	Type	Value	Comment
wk append	Longint	2	Insert contents at end of range
wk prepend	Longint	1	Insert contents at beginning of range
wk replace	Longint	0	Replace range contents

In the optional *rangeUpdate* parameter, you can pass one of the following constants to specify whether or not the inserted contents are included in the resulting range:

Constant	Type	Value	Comment
wk exclude from range	Longint	1	Inserted contents not included in updated range
wk include in range	Longint	0	Inserted contents included in updated range (default)

If you do not pass the *rangeUpdate* parameter, by default the inserted contents are included in the resulting range.

## Example

While building invoices, you want to insert page breaks except on the last page:

```

$nbInvoices:=Records in selection([INVOICE])
For($j;1;$nbInvoices)
  ... //processing invoices
  If($j#command_5$nbInvoices) //insert page break except for last page
    WP INSERT BREAK($buildRange;wk page break;wk append;wk exclude from range)
  End if
End for

```

WP INSERT DOCUMENT ( rangeObj ; wpDoc ; mode {; rangeUpdate} )

Parameter	Type		Description
rangeObj	Object	→	4D Write Pro target range
wpDoc	Object	→	4D Write Pro document to insert
mode	Longint	→	Insertion mode
rangeUpdate	Longint	→	Range update mode

## Description

The **WP INSERT DOCUMENT** command inserts the *wpDoc* document in the *rangeObj* range according to the specified insertion *mode* and *rangeUpdate* parameter.

In *rangeObj*, pass a valid 4D Write Pro standard range object. A *rangeObj* is a part of a 4D Write Pro document; it can be created using the **WP Get selection**, **WP Get bookmark range** or **WP Get range** commands.

The inserted *wpDoc* document can be any 4D Write Pro document object created using the **WP New** or **WP Import document** command. Only the body children elements are inserted (sections and bookmarks in the destination range are preserved). In addition, the elements are copied, so *wpDoc* can be re-used several times.

In the *mode* parameter, pass one or a combination of the following constants from the **4D Write Pro Constants** theme to indicate the insertion mode to be used for the document in the destination *rangeObj* range:

Constant	Type	Value	Comment
wk append	Longint	2	Insert contents at end of range
wk prepend	Longint	1	Insert contents at beginning of range
wk replace	Longint	0	Replace range contents

You can combine one of the previous constants with the following insertion options:

Constant	Type	Value	Comment
wk freeze expressions	Longint	64	Freeze expressions at the moment of the insertion
wk inherit style from paragraph	Longint	32	Inserted contents inherits character style from the paragraph default character style.
wk keep paragraph styles	Longint	128	Keep destination paragraph styles

In the optional *rangeUpdate* parameter, you can pass one of the following constants to specify whether or not the inserted contents are included in the resulting range:

Constant	Type	Value	Comment
wk exclude from range	Longint	1	Inserted contents not included in updated range
wk include in range	Longint	0	Inserted contents included in updated range (default)

If you do not pass a *rangeUpdate* parameter, by default the inserted contents are included in the resulting range.

## Example 1

You want to replace the contents of a document by the text selected in another one:

```
$tempRange:=WP Get selection(WPTemplate) //we retrieve the user selection in the WPTemplate document
$doctoCopy:=WP New($tempRange) //create a new document based on WPTemplate
WP INSERT DOCUMENT(WPDoc;$doctoCopy;wk_replace) //replace contents of WPDoc by the contents of the new document
```

## Example 2

You have defined a template document with different preformatted parts, each of them being stored as a bookmark. When producing a final document from the template, you can extract any bookmark as a new document and insert it in the final document.

```
ARRAY TEXT($_BookmarkNames;0)
WP GET BOOKMARKS ([TEMPLATES]WP;$_BookmarkNames) //get the bookmarks from the template
$targetRange:=WP New //create an empty document (will be the final document)

$P:=Find in array($_BookmarkNames;"Main_Header") //handle the main header part
If($P>0)
    $Range:=WP Get bookmark range(WParea;$_BookmarkNames{$P}) //select the range
    $RangeDoc:=WP New($Range) //create a new document from the range
    WP INSERT DOCUMENT($targetRange;$RangeDoc;wk append+wk freeze expressions) //wk append=after
replacement, $targetRange is equal to end of replaced text
End if
```

WP INSERT PICTURE ( rangeObj ; picture ; mode {; rangeUpdate} )

Parameter	Type	Description
rangeObj	Object	→ Range object
picture	Picture, String	→ Picture field or variable, or path to picture file on disk
mode	Longint	→ Insertion mode
rangeUpdate	Longint	→ Range update mode

## Description

The **WP INSERT PICTURE** command inserts the *picture* in the *rangeObj* according to the specified insertion *mode* and *rangeUpdate* parameter. The picture will be inserted as a character in the *rangeObj*.

In *rangeObj*, pass a valid 4D Write Pro standard range object. A *rangeObj* is a part of a 4D Write Pro document; it can be created using the **WP Get selection**, **WP Get bookmark range** or **WP Get range** commands.

In *picture*, you can pass:

- either a 4D picture field or variable,
- or a string containing a path to a picture file stored on disk, expressed using the system syntax.  
If you use a string, you can pass either a full pathname, or a pathname relative to the database structure file. You can also pass a file name, in which case the file must be located next to the database structure file. If you pass a file name, you need to indicate the file extension.

Any picture format supported by 4D can be used (see the **Pictures** section). You can get the list of available picture formats using the **PICTURE CODEC LIST** command. If the *picture* encapsulates several formats (codecs), 4D Write Pro only keeps one format for display and one format for printing (if different) in the document; the "best" formats are automatically selected.

In the *mode* parameter, pass one of the following constants to indicate the insertion mode to be used on the picture in the document:

Constant	Type	Value	Comment
wk append	Longint	2	Insert contents at end of range
wk prepend	Longint	1	Insert contents at beginning of range
wk replace	Longint	0	Replace range contents

In the optional *rangeUpdate* parameter, you can pass one of the following constants to specify whether or not the inserted picture is included in the range:

Constant	Type	Value	Comment
wk exclude from range	Longint	1	Inserted contents not included in updated range
wk include in range	Longint	0	Inserted contents included in updated range (default)

If you do not pass a *rangeUpdate* parameter, by default the inserted picture is included in the range.

## Example

In the following example, a user selects the picture they want to insert into the range object and will be warned if this picture could not be inserted:

```
C_OBJECT($wpRange)
$wpRange:=WP Get selection([EXAMPLES]wpDoc)

C_BOOLEAN($fail)
$fail:=False
```

```
//ask user to choose a picture on the disk that they want to insert
$imgRef:=Open document("")
//if user does not cancel
If(OK=1)
//if the file is a supported picture file
  If(Is picture file(document))
// insert picture selected by user
    WP INSERT PICTURE($wpRange;document;wk_replace)
  Else
    $fail:=True
  End if
Else
  $fail:=True
End if
//if the insertion failed, alert the user
If($fail)
  ALERT("Picture insertion failed")
End if
```



## WP Is font style supported

WP Is font style supported ( rangeObj ; wpFontStyle ) -> Function result

Parameter	Type	Description
rangeObj	Object	⇒ Range object to parse
wpFontStyle	Longint	⇒ Font style constant: wk font bold, wk font italic, wk text underline style, wk text linethrough style
Function result	Boolean	⇒ True if any part of range supports wpFontStyle; False otherwise

### Description

---

The **WP Is font style supported** command returns True if the *wpFontStyle* style is supported by any part of the text in *rangeObj*.

In *rangeObj*, pass a valid 4D Write Pro range object. A *rangeObj* is a part of a 4D Write Pro document; it can be created by the [WP Get selection](#) or [WP Get range](#) command, or [WP Get paragraphs](#) / [WP Get pictures](#).

In *wpFontStyle*, pass one of the following style constants from the "4D Write Pro" constant theme:

Constant	Type	Value	Comment
wk font bold	String	fontBold	<p>Specifies thickness of text (depends on available font styles). Possible values:</p> <ul style="list-style-type: none"> <li>• <u>wk true</u> to set selected characters to bold font style; with the <b>WP GET ATTRIBUTES</b> command, <u>wk true</u> is returned if at least one selected character supports a bold font style.</li> <li>• <u>wk false</u> (default) to remove the bold font style from selected characters if any; with the <b>WP GET ATTRIBUTES</b> command, <u>wk false</u> is returned if none of the selected characters supports a bold font style.</li> </ul>
wk font italic	String	fontItalic	<p>Specifies italic style of text (depends on available font styles). Possible values:</p> <ul style="list-style-type: none"> <li>• <u>wk true</u> to set selected characters to italic or oblique font style; with the <b>WP GET ATTRIBUTES</b> command, <u>wk true</u> is returned if at least one selected character supports an italic or oblique font style.</li> <li>• <u>wk false</u> (default) to remove the italic or oblique font style from selected characters if any; with the <b>WP GET ATTRIBUTES</b> command, <u>wk false</u> is returned if none of the selected characters supports an italic or oblique font style.</li> </ul>
wk text linethrough style	String	textLinethroughStyle	<p>Specifies style of text linethrough (if any). Possible values:</p> <ul style="list-style-type: none"> <li>• <u>wk none</u> (default): no linethrough effect</li> <li>• <u>wk solid</u>: draw a solid line on the selected text</li> <li>• <u>wk dotted</u>: draw a dotted line on the selected text</li> <li>• <u>wk dashed</u>: draw a dashed line on the selected text</li> <li>• <u>wk double</u>: draw a double line on the selected text</li> <li>• <u>wk semi transparent</u>: dimmed line on the selected text. Can be combined with another line style.</li> <li>• <u>wk word</u>: draw a line on words only (exclude blank spaces). Can be combined with another line style.</li> </ul>
wk text underline style	String	textUnderlineStyle	<p>Specifies style of text underline (if any). Possible values:</p> <ul style="list-style-type: none"> <li>• <u>wk none</u> (default): no underline</li> <li>• <u>wk solid</u>: draw a solid underline</li> <li>• <u>wk dotted</u>: draw a dotted underline</li> <li>• <u>wk dashed</u>: draw a dashed underline</li> <li>• <u>wk double</u>: draw a double underline</li> <li>• <u>wk semi transparent</u>: dimmed underline. Can be combined with another line style.</li> <li>• <u>wk word</u>: draw an underline for words only (exclude blank spaces). Can be combined with another line style.</li> </ul>

Typically, this command is provided to allow developers to implement custom interface objects, such as a pop-up menu offering style options based on the selected text.

WP New {{ source }} -> Function result

Parameter	Type	Description
source	String, BLOB, Object	→ String: 4D HTML source, BLOB: 4D Write Blob document (.4w7/.4wt) or 4D Write Pro document (.4wp) Object: a 4D Write Pro object range
Function result	Object	↩ 4D Write Pro object

## Description

The **WP New** command creates and returns a 4D Write Pro object.

By default, if you omit the *source* parameter, the command returns an empty 4D Write Pro object.

You can also use the *source* parameter, in which case the new 4D Write Pro object will be filled with the contents of the *source*. You can pass:

- a *string* parameter: In this case, you pass a 4D HTML source, i.e. a text exported by **WP EXPORT VARIABLE** with the [wk web page html 4D](#) option. This text can contain references (4D tags and expressions) and embedded images.
- a *blob* parameter: In this case, you pass either:
  - a 4D Write Pro (.4wp) format document stored in a BLOB. For more information about the 4D Write Pro document format, please refer to [.4wp document format](#).
  - a legacy 4D Write area loaded in a BLOB (BLOBs containing .4w7 or .4wt documents are supported). For a detailed list of 4D Write features that are currently supported in 4D Write Pro objects, please refer to the [Importing 4D Write documents](#) section.  
If you want to import a 4D Write document (.4w7 or .4wt) stored on disk, you can also consider using the [WP Import document](#) command.
- an *object* parameter: In this case, you pass a 4D Write Pro range object. **WP New** will return a new document created from the specified range. Note that, if the range is not equal to the full document range, only the first section is exported and bookmarks are not exported, if any.

The returned object is a complete document that can be passed to the [MissingRef](#) command, for example.

## Example 1

You want to create an empty 4D Write Pro object:

```
myWPObject:=WP New
```

## Example 2

You want to create a 4D Write Pro object containing a simple 4D expression reference:

```
C_TEXT(myText)
myText:="Today is "
ST_INSERT_EXPRESSION(myText;"string(current date;System date long)";ST_End_text)
myWPA:=WP New(myText)
```

## Example 3

You want to initialize your Write Pro area with a previously-created template:

```

//Export template from an existing area
C_TEXT(wpTemplate)
WP EXPORT VARIABLE(myWPArea;wpTemplate;wk_web_page_html_4D)

// use the template for a new area
C_OBJECT(myNewWPA)
myNewWPA:=WP New(wpTemplate)

```

## Example 4

---

You want to import a 4D Write document stored in a 4D field of the current record into a new 4D Write Pro area:

```

C_OBJECT(wpArea)
wpArea=WP New([Templates]Reference_)

```

## Example 5

---

You have defined a template document with different preformatted parts, each of them being stored as a bookmark. When producing a final document from the template, you can extract any bookmark as a new document and insert it in the final document.

```

ARRAY TEXT($_BookmarkNames;0)
WP GET BOOKMARKS([TEMPLATES]WP;$_BookmarkNames) //get the bookmarks from the template
$targetRange:=WP New //create an empty document (will be the final document)

$P:=Find in array($_BookmarkNames;"Main_Header") //handle the main header part
If($P>0)
    $Range:=WP Get bookmark range(WParea;$_BookmarkNames{$P}) //select the range
    $RangeDoc:=WP New($Range) //create a new document from the range
    WP INSERT DOCUMENT($targetRange;$RangeDoc;wk_append+wk_freeze_expressions) //wk append=after
replacement, $targetRange is equal to end of replaced text
End if

```

WP PRINT ( wpDoc {; printLayout} )

Parameter	Type	Description
wpDoc	Object	→ Name of 4D Write Pro document
printLayout	Longint	→ Print layout for 4D Write Pro document: 0 (default)=4D Write Pro layout, 1=HTML WYSIWYG

## Description

The **WP PRINT** command launches a print job for the 4D Write Pro document specified in *wpDoc*, or adds the document to the current print job if it is called between **OPEN PRINTING JOB** and **CLOSE PRINTING JOB** (only on 64-bit versions of 4D, see below). **WP PRINT** uses print settings defined by the 4D **PRINT SETTINGS** or **SET PRINT OPTION** commands, except for page margins which are always based on the 4D Write Pro document page settings. **WP PRINT** uses current page setup options (such as page size and orientation), or those of the document if **WP USE PAGE SETUP** was called previously.

The optional *printLayout* parameter can be used to set the HTML WYSIWYG view for the print output. You can pass one of the following constants from the "4D Write Pro" theme:

Constant	Type	Value	Comment
wk 4D Write Pro layout	Longint	0	Standard 4D Write Pro layout, which can include some specific style attributes
wk html wysiwyg	Longint	1	In this layout, any 4D Write Pro advanced attributes which are not compliant with all browsers are removed

If *printLayout* is omitted, 4D Write Pro layout (0) is used by default.

**Note:** When printed with **WP PRINT**, 4D Write Pro documents are always printed as in Page view mode, regardless of the View property setting for the area (see **Configuring View properties**).

## Note for 32-bit versions

The **WP PRINT** command is supported in 4D 32-bit versions but with the following limitation: it cannot be called within a 4D print job started with **OPEN PRINTING JOB**. If the command is called in a print job on a 32-bit version of 4D, an error is returned.

## Example

You want to print a 4D Write Pro document in standard or HTML wysiwyg layout depending on the value of a variable:

```
// print using a specific layout HTML wysiwyg or 4D Write Pro Layout
If(rb_htmlwysiwyg=1)
  WP PRINT(writeProDoc;wk_html_wysiwyg)
Else
  WP PRINT(writeProDoc;wk_4D_Write_Pro_layout)
End if
```

## WP RESET ATTRIBUTES

WP RESET ATTRIBUTES ( rangeObj ; attribName { ; attribName2 ; ... ; attribNameN } )

Parameter	Type		Description
rangeObj	Object	→	4D Write Pro range
attribName	String	→	Name of attribute(s) to remove

### Description

The **WP RESET ATTRIBUTES** command allows you to reset the value of one or more attributes in the 4D Write Pro *rangeObj*. This command can remove any kind of 4D Write Pro internal attribute: character, paragraph, document, or image.

A *rangeObj* is a part of a 4D Write Pro document that can be created by different commands. The following table provides the scope of the **WP RESET ATTRIBUTES** command depending on the target object and attributes:

Parameter	Common attributes (except "verticalAlign")	Document-only attributes	Paragraph-only attributes	Character-only attributes (and "verticalAlign")
<i>rangeObj</i> from <b>WP Get paragraphs</b>	paragraphs	document	paragraphs	paragraphs
<i>rangeObj</i> from <b>WP Get pictures</b>	images	document	-	-
<i>rangeObj</i> from <b>WP Get range</b> or <b>WP Get selection</b>	paragraphs for the range	document	paragraphs for the range (intersecting paragraphs)	characters for the range
<i>wpDoc</i>	document	document	paragraphs for all the document	characters for all the document

When an attribute value is removed using the **WP RESET ATTRIBUTES** command, it is reset to the default value. Default values are listed in the **4D Write Pro Attributes** section.

If the attribute to be reset was not defined in the *rangeObj*, the command does nothing.

### Example

You want to remove several attributes from the following selection:

Description

The DIALOG command presents the form form to the user. This command is often used to get information from the user through the use of variables, or to present information to the user, such as options for performing an operation.

It is common to display the form inside a modal window created with the Open window command.

You can execute:

```
$range:=WP Get selection (*;"WParea")
WP RESET ATTRIBUTES ($range;wk_padding)
WP RESET ATTRIBUTES ($range;wk_background_color)
WP RESET ATTRIBUTES ($range;wk_text_underline_style)
WP RESET ATTRIBUTES ($range;wk_margin)
WP RESET ATTRIBUTES ($range;wk_border_style)
```

---

The resulting document is:

## Description

The DIALOG command presents the form form to the user. *This command is often used to get information from the user through the use of variables, or to **present information to the user**, such as options for performing an operation.*

It is common to display the form inside a modal window created with the Open window command.

```
WP SELECT ( { * ; } wpArea { ; rangeObj } { ; startRange ; endRange } )
```

Parameter	Type	Description
*	Operator	➡ If specified, <i>wpArea</i> is a form object name (string). If omitted, <i>wpArea</i> is an object field or variable.
<i>wpArea</i>	String, Object	➡ Form object name (if * is specified) or 4D Write Pro object variable or field (if * is omitted)
<i>rangeObj</i>	Object	➡ Range object to apply to create a selection
<i>startRange</i>	Longint	➡ Starting offset of text range
<i>endRange</i>	Longint	➡ Ending offset of text range

## Description

---

The **WP SELECT** command creates a new text selection in the *wpArea* 4D Write Pro area, based upon the *rangeObj* or a new range defined by *startRange* and *endRange*.

If you pass the optional \* parameter, you indicate that *wpArea* is a form object name (string). If you do not pass this parameter, you indicate that *wpArea* is a 4D Write Pro object variable or field. If no valid 4D Write Pro area is passed in the *wpArea* parameter, the command does nothing.

To define the selection range, you can either pass an existing range object in *rangeObj*, or pass a pair of *startRange* / *endRange* boundaries:

- **First syntax:** `WP SELECT({ * ; } wpArea ; rangeObj)`  
A *rangeObj* is a part of a 4D Write Pro document; it can be created by different commands such as **WP Get range**, **WP Get selection**, **WP Get paragraphs**, or **WP Get pictures**.
- **Second syntax:** `WP SELECT({ * ; } wpArea ; startRange ; endRange)`  
In this case, in *startRange* and *endRange* you pass values corresponding to the position of the first and last characters to select in the document. You can pass wk start text in *startRange* to define the beginning of the document, and wk end text in *endRange* to define the end of the document. Keep in mind that a 4D Write Pro document not only contains visible text but also formatting tags that are included in the range.

## Example

---

The following code:

```
$range2:=WP Get range ([SAMPLE]WP;wk_start_text;12)
WP SELECT (*;"WParea";$range2)
```

... will have the same result as:

```
WP SELECT (*;"WParea";wk_start_text;12)
```



## WP SET ATTRIBUTES

```
WP SET ATTRIBUTES ( rangeObj | wpDoc ; attribName ; attribValue {; attribName2 ; attribValue2 ; ... ;  
attribNameN ; attribValueN} )
```

Parameter	Type		Description
rangeObj   wpDoc	Object	→	4D Write Pro range or document
attribName	String	→	Name of attribute to set
attribValue	String, Real, Boolean	→	New attribute value

### Description

---

The **WP SET ATTRIBUTES** command allows you to set the value of any attribute in a 4D Write Pro range or document. This command gives you access to any kind of 4D Write Pro internal attribute: character, paragraph, document, or image.

In the first parameter, you can pass either a 4D Write Pro range object (*rangeObj*) or a 4D Write document reference (*wpDoc*). A *rangeObj* is a part of a 4D Write Pro document that can be created by different commands. The following table provides the scope of the **WP SET ATTRIBUTES** command depending on the target object and attributes:

Parameter	Common attributes (except "verticalAlign")	Document-only attributes	Paragraph-only attributes	Character-only attributes (and "verticalAlign")
<i>rangeObj</i> from <b>WP Get paragraphs</b>	paragraphs	document	paragraphs	paragraphs
<i>rangeObj</i> from <b>WP Get pictures</b>	images	document	-	-
<i>rangeObj</i> from <b>WP Get range</b> or <b>WP Get selection</b>	paragraphs for the range	document	paragraphs for the range (intersecting paragraphs)	characters for the range
<i>wpDoc</i>	document	document	paragraphs for all the document	characters for all the document

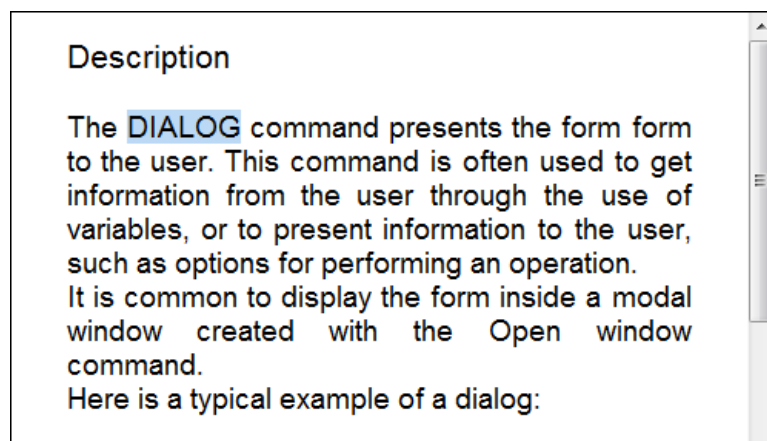
Common attributes (such as "margin", "padding", etc.) mean common to the document, paragraphs, and/or pictures.

For a comprehensive list of attributes to pass in *attribName*, as well as their respective values, please refer to the **4D Write Pro Attributes** section.

### Example 1

---

In this 4D Write Pro area, you selected a word:



If you execute the following code:

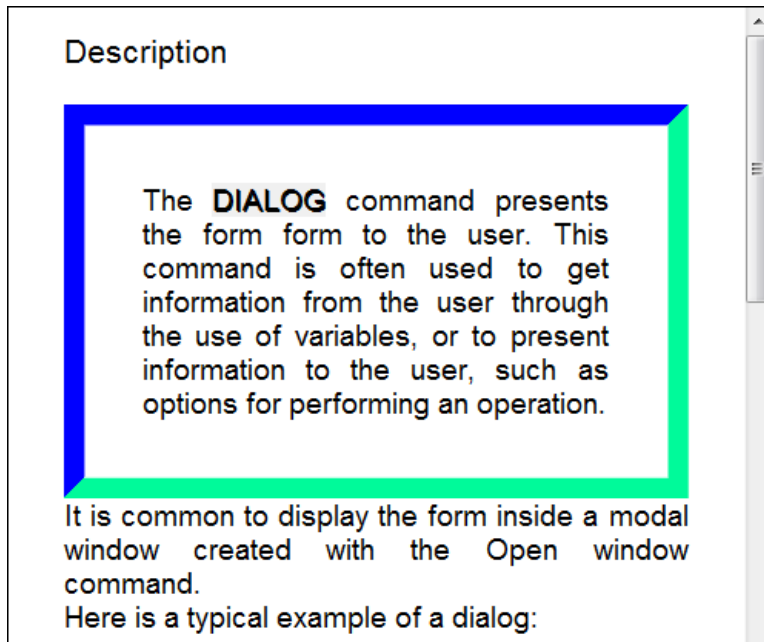
```

$range:=WP Get selection(*;"WParea") //get the selected range

// set the shadow offset in pt for the selected text
WP SET ATTRIBUTES($range;wk text shadow offset;1)
//set the paragraph padding
WP SET ATTRIBUTES($range;wk padding;1)
//define a border of 10 pt
WP SET ATTRIBUTES($range;wk border style;wk solid)
WP SET ATTRIBUTES($range;wk border width;10)
//set the border colors
WP SET ATTRIBUTES($range;wk border color;"blue")
WP SET ATTRIBUTES($range;wk border color bottom;"#command_500FA9A") //medium green
WP SET ATTRIBUTES($range;wk border color right;"#command_500FA9A")

```

You get the following result:



## Example 2

This example illustrates the use of wk inside and wk outside constants:

```

$wprange:=WP Get selection(writeProdoc)
WP SET ATTRIBUTES($wprange;wk border style+wk inside;wk dotted)
WP SET ATTRIBUTES($wprange;wk border style+wk outside;wk solid)
WP SET ATTRIBUTES($wprange;wk border color+wk outside;"#command_500FA9A")

```

Assuming all of the contents were selected, the result is:

```

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Sed non risus.
Suspendisse lectus tortor, dignissim sit amet, adipiscing nec, ultricies sed,
dolor.
.....
Cras elementum ultrices diam. Maecenas ligula massa, varius a, semper
congue, euismod non, mi. Proin porttitor, orci nec nonummy molestie, enim
est eleifend mi, non fermentum diam nisl sit amet erat. Duis semper.
.....
Duis arcu massa, scelerisque vitae, consequat in, pretium a, enim.
Pellentesque congue. Ut in risus volutpat libero pharetra tempor. Cras
vestibulum bibendum augue. Praesent egestas leo in pede. Praesent blandit
odio eu enim.
.....
Pellentesque sed dui ut augue blandit sodales. Vestibulum ante ipsum primis
in faucibus orci luctus et ultrices posuere cubilia Curae; Aliquam nibh. Mauris
ac mauris sed pede pellentesque fermentum. Maecenas adipiscing ante non
diam sodales hendrerit.

```

## WP USE PAGE SETUP

WP USE PAGE SETUP ( wpDoc )

Parameter	Type		Description
wpDoc	Object	→	Name of 4D Write Pro document

### Description

---

The **WP USE PAGE SETUP** command modifies the current printer page settings based on the 4D Write Pro document attributes for page size and orientation. This command should be called just before **WP PRINT** in order to synchronize the current printer page settings with the 4D Write Pro document page settings.

Other settings are defined by the 4D **PRINT SETTINGS** command. Current print settings are set for the whole 4D session.

### Example

---

Before printing a document, you want to reset its size and orientation to values stored in the document:

```
WP USE PAGE SETUP(writeProDoc)
```



Constant	Type	Value	Comment
wk 4D Write Pro layout	Longint	0	Standard 4D Write Pro layout, which can include some specific style attributes 4D Write Pro document is saved in a native archive format (zipped HTML and images saved in a separate folder). 4D specific tags are included and 4D expressions are not computed. This format is particularly suitable for saving and archiving 4D Write Pro documents on disk without any loss.
wk 4wp	Longint	4	4D Write Pro document is saved in a native archive format (zipped HTML and images saved in a separate folder). 4D specific tags are included and 4D expressions are not computed. This format is particularly suitable for saving and archiving 4D Write Pro documents on disk without any loss.
wk append	Longint	2	Insert contents at end of range
wk armenian	Longint	19	Traditional Armenian numbering style used (value for <a href="#">wk list style type</a> )
wk author	String	author	Specifies name of author of the document (string)
wk auto	Longint	0	Value of property (constant) to which it is applied is adjusted automatically according to content or context of the element. Specifies painting area of background. Possible values:
wk background clip	String	backgroundClip	<ul style="list-style-type: none"> <li><a href="#">wk border box</a> (default): background is painted to outside edge of the border</li> <li><a href="#">wk content box</a>: background is painted within the content box</li> <li><a href="#">wk padding box</a>: background is painted to outside edge of the padding (or to inside edge of the border, if any)</li> </ul>
wk background color	String	backgroundColor	<p>Specifies background color of an element. Possible values:</p> <ul style="list-style-type: none"> <li>a CSS color ("#command_5010101" or "#command_5FFFFFFF" or "red").</li> <li>a 4D color longint value (see <b>OBJECT SET COLOR</b> command)</li> <li>a longint array containing an element for each R, G, B component (0-255)</li> </ul> <p>Default for documents is "#command_5FFFFFFF" and <a href="#">wk transparent</a>, or "transparent" for paragraphs and images.</p>
wk background image	String	backgroundImage	<p>Specifies image to use as background. Possible values to set:</p> <ul style="list-style-type: none"> <li>Image URL (string). Can be absolute or relative to the structure file.</li> <li>Picture variable or field.</li> </ul> <p>Value returned (<b>WP GET ATTRIBUTES</b>): URI (network URL or data URI). It may not be equal to the initial URL for an image not referenced with the network URL (only network URLs are kept). For local file URLs, the image stream itself is kept in the document and thus the URL returned is a data URI with the image stream encoded in base64.</p>
wk background origin	String	backgroundOrigin	<p>Specifies where background image is positioned. Possible values:</p> <ul style="list-style-type: none"> <li><a href="#">wk padding box</a> (default): background image starts at padding (or inside border edge) rectangle</li> <li><a href="#">wk border box</a>: background image starts at border (outside edge) rectangle</li> <li><a href="#">wk content box</a>: background image starts at content rectangle</li> </ul> <p>Specifies horizontal starting position of a background image. Possible values:</p> <ul style="list-style-type: none"> <li><a href="#">wk left</a> (default): background image starts horizontally on left</li> </ul>
wk			

wk background position h	String	backgroundPositionH	<p>side of the element</p> <ul style="list-style-type: none"> <li>• <a href="#">wk center</a>: background image starts horizontally at center of the element</li> <li>• <a href="#">wk right</a>: background image starts horizontally on right side of the element</li> </ul> <p>Specifies vertical starting position of a background image. Possible values:</p>
wk background position v	String	backgroundPositionV	<ul style="list-style-type: none"> <li>• <a href="#">wk top</a> (default): background image starts vertically at top of the element</li> <li>• <a href="#">wk middle</a>: background image starts vertically at middle of the element</li> <li>• <a href="#">wk bottom</a>: background image starts vertically at bottom of the element</li> </ul> <p>Specifies if and how a background image is repeated. Possible values:</p>
wk background repeat	String	backgroundRepeat	<ul style="list-style-type: none"> <li>• <a href="#">wk repeat</a> (default): background image is repeated both vertically and horizontally</li> <li>• <a href="#">wk no repeat</a>: background image is not repeated</li> <li>• <a href="#">wk repeat x</a>: background image is repeated only horizontally</li> <li>• <a href="#">wk repeat y</a>: background image is repeated only vertically</li> </ul> <p>Specifies horizontal size of background image. Possible values:</p>
wk background size h	String	backgroundSizeH	<ul style="list-style-type: none"> <li>• <a href="#">wk auto</a> (default): background image contains its width</li> <li>• <a href="#">wk contain</a>: scales image to largest size so that it fits entirely in the content area, while preserving its aspect ratio. This option also modifies the value of the other size attribute.</li> <li>• <a href="#">wk cover</a>: scales background image to be as large as possible so that the background area is entirely covered by the background image, while preserving its aspect ratio. Some parts of the background image may be cropped. This option also modifies the value of the other size attribute.</li> <li>• Defined size: background image horizontal size expressed using a real or string value: <ul style="list-style-type: none"> <li>◦ Real: Size in <a href="#">wk layout unit</a>.</li> <li>◦ String: CSS string with value and unit concatenated. E.g.: 12pt for 12 points, or 1.5cm for 1.5 centimeters. Minimum value: 0pt, maximum value: 10000pt. A relative value (percentage %) is supported.</li> </ul> </li> </ul> <p>Specifies vertical size of background image. Possible values:</p>
wk background size v	String	backgroundSizeV	<ul style="list-style-type: none"> <li>• <a href="#">wk auto</a> (default): background image contains its height</li> <li>• <a href="#">wk contain</a>: scales image to largest size so that it fits entirely in the content area, while preserving its aspect ratio. This option also modifies the value of the other size attribute.</li> <li>• <a href="#">wk cover</a>: scales background image to be as large as possible so that the background area is entirely covered by the background image, while preserving its aspect ratio. Some parts of the background image may be cropped. This option also modifies the value of the other size attribute.</li> <li>• Defined size: background image vertical size expressed using a real or string value: <ul style="list-style-type: none"> <li>◦ Real: Size in <a href="#">wk layout unit</a>.</li> </ul> </li> </ul>

- String: CSS string with value and unit concatenated. E.g.: 12pt for 12 points, or 1.5cm for 1.5 centimeters. Minimum value: 0pt, maximum value: 10000pt. A relative value (percentage %) is supported.

wk bar Longint 4

Inserts a vertical bar at tab position (value for [wk tab stop types](#))

wk baseline Longint 4

Aligns baseline of element with baseline of parent element (value for [wk vertical align](#))

wk border box Longint 0

Background is clipped to the border box (value for [wk background clip](#))

Sets color of all four borders. Possible values:

- a CSS color ("#command\_5010101" or "#command\_5FFFFFFF" or "red").
- a 4D color longint value (see **OBJECT SET COLOR** command)
- a longint array containing an element for each R, G, B component (0-255)

wk border color String borderColor

Default is "#command\_5000000" (if string value). If there are multiple colors, **WP GET ATTRIBUTES** returns an empty string.

Sets color of bottom border. Possible values:

wk border color bottom String borderColorBottom

- a CSS color ("#command\_5010101" or "#command\_5FFFFFFF" or "red"). Default is "#command\_5000000"
- a 4D color longint value (see **OBJECT SET COLOR** command)
- a longint array containing an element for each R, G, B component (0-255)

Sets color of left border. Possible values:

wk border color left String borderColorLeft

- a CSS color ("#command\_5010101" or "#command\_5FFFFFFF" or "red"). Default is "#command\_5000000"
- a 4D color longint value (see **OBJECT SET COLOR** command)
- a longint array containing an element for each R, G, B component (0-255)

Sets color of right border. Possible values:

wk border color right String borderColorRight

- a CSS color ("#command\_5010101" or "#command\_5FFFFFFF" or "red"). Default is "#command\_5000000"
- a 4D color longint value (see **OBJECT SET COLOR** command)
- a longint array containing an element for each R, G, B component (0-255)

Sets color of top border. Possible values:

wk border color top String borderColorTop

- a CSS color ("#command\_5010101" or "#command\_5FFFFFFF" or "red"). Default is "#command\_5000000"
- a 4D color longint value (see **OBJECT SET COLOR** command)
- a longint array containing an element for each R, G, B component (0-255)

Specifies a rounded border. Possible values:

wk border radius String borderRadius

- [wk none](#) (default): the border does not have rounded angles
- Radius value expressed using an integer or a string value:
  - Integer: Radius in [wk layout unit](#).
  - String: CSS string with value and unit concatenated. E.g.:

12pt for 12 points, or 1.5cm for 1.5 centimeters.

Specifies style of all four borders. Possible values:

wk border  
style

String

borderStyle

- [wk none](#) (default): no border
- [wk hidden](#): same as [wk none](#), except in border conflict resolution
- [wk solid](#): solid border
- [wk dotted](#): dotted border
- [wk dashed](#): dashed border
- [wk double](#): double border
- [wk groove](#): 3D groove border (the actual effect depends on the border color)
- [wk ridge](#): 3D ridged border (the actual effect depends on the border color)
- [wk inset](#): 3D inset border (the actual effect depends on the border color)

Specifies style of bottom border. Possible values:

wk border  
style  
bottom

String

borderStyleBottom

- [wk none](#) (default): no bottom border
- [wk hidden](#): same as [wk none](#), except in border conflict resolution
- [wk solid](#): solid bottom border
- [wk dotted](#): dotted bottom border
- [wk dashed](#): dashed bottom border
- [wk double](#): double bottom border
- [wk groove](#): 3D groove bottom border (the actual effect depends on the border color)
- [wk ridge](#): 3D ridged bottom border (the actual effect depends on the border color)
- [wk inset](#): 3D inset bottom border (the actual effect depends on the border color)

Specifies style of left border. Possible values:

wk border  
style left

String

borderStyleLeft

- [wk none](#) (default): no left border
- [wk hidden](#): same as [wk none](#), except in border conflict resolution
- [wk solid](#): solid left border
- [wk dotted](#): dotted left border
- [wk dashed](#): dashed left border
- [wk double](#): double left border
- [wk groove](#): 3D groove left border (the actual effect depends on the border color)
- [wk ridge](#): 3D ridged left border (the actual effect depends on the border color)
- [wk inset](#): 3D inset left border (the actual effect depends on the border color)

Specifies style of right border. Possible values:

wk border

String

borderStyleRight

- [wk none](#) (default): no right border
- [wk hidden](#): same as [wk none](#), except in border conflict resolution
- [wk solid](#): solid right border
- [wk dotted](#): dotted right border
- [wk dashed](#): dashed right border



style right	String	borderStyleRight	<ul style="list-style-type: none"> <li>• <a href="#">wk double</a>: double right border</li> <li>• <a href="#">wk groove</a>: 3D groove right border (the actual effect depends on the border color)</li> <li>• <a href="#">wk ridge</a>: 3D ridged right border (the actual effect depends on the border color)</li> <li>• <a href="#">wk inset</a>: 3D inset right border (the actual effect depends on the border color)</li> </ul>
wk border style top	String	borderStyleTop	<p>Specifies style of top border. Possible values:</p> <ul style="list-style-type: none"> <li>• <a href="#">wk none</a> (default): no top border</li> <li>• <a href="#">wk hidden</a>: same as <a href="#">wk none</a>, except in border conflict resolution</li> <li>• <a href="#">wk solid</a>: solid top border</li> <li>• <a href="#">wk dotted</a>: dotted top border</li> <li>• <a href="#">wk dashed</a>: dashed top border</li> <li>• <a href="#">wk double</a>: double top border</li> <li>• <a href="#">wk groove</a>: 3D groove top border (the actual effect depends on the border color)</li> <li>• <a href="#">wk ridge</a>: 3D ridged top border (the actual effect depends on the border color)</li> <li>• <a href="#">wk inset</a>: 3D inset top border (the actual effect depends on the border color)</li> </ul> <p>Specifies width of all four borders. You need to specify the border style before setting the border width. Possible values:</p> <ul style="list-style-type: none"> <li>• Width expressed using an integer or a string value: <ul style="list-style-type: none"> <li>◦ Integer: Width in <a href="#">wk layout unit</a>.</li> <li>◦ String: CSS string with value and unit concatenated. E.g.: 12pt for 12 points, or 1.5cm for 1.5 centimeters</li> </ul> </li> <li>• Default value: 2pt</li> </ul>
wk border width	String	borderWidth	<p>Specifies width of bottom border. Possible values:</p> <ul style="list-style-type: none"> <li>• Width expressed using an integer or a string value: <ul style="list-style-type: none"> <li>◦ Integer: Width in <a href="#">wk layout unit</a>.</li> <li>◦ String: CSS string with value and unit concatenated. E.g.: 12pt for 12 points, or 1.5cm for 1.5 centimeters</li> </ul> </li> <li>• Default value: 2pt</li> </ul>
wk border width bottom	String	borderWidthBottom	<p>Specifies width of left border. Possible values:</p> <ul style="list-style-type: none"> <li>• Width expressed using an integer or a string value: <ul style="list-style-type: none"> <li>◦ Integer: Width in <a href="#">wk layout unit</a>.</li> <li>◦ String: CSS string with value and unit concatenated. E.g.: 12pt for 12 points, or 1.5cm for 1.5 centimeters</li> </ul> </li> <li>• Default value: 2pt</li> </ul>
wk border width left	String	borderWidthLeft	<p>Specifies width of right border. Possible values:</p> <ul style="list-style-type: none"> <li>• Width expressed using an integer or a string value: <ul style="list-style-type: none"> <li>◦ Integer: Width in <a href="#">wk layout unit</a>.</li> <li>◦ String: CSS string with value and unit concatenated. E.g.: 12pt for 12 points, or 1.5cm for 1.5 centimeters</li> </ul> </li> <li>• Default value: 2pt</li> </ul>
wk border width right	String	borderWidthRight	<p>Specifies width of top border. Possible values:</p>

wk border width top	String	borderWidthTop	<ul style="list-style-type: none"> <li>Width expressed using an integer or a string value: <ul style="list-style-type: none"> <li>Integer: Width in <a href="#">wk layout unit</a>.</li> <li>String: CSS string with value and unit concatenated. E.g.: 12pt for 12 points, or 1.5cm for 1.5 centimeters</li> </ul> </li> <li>Default value: 2pt</li> </ul>
wk bottom	Longint	1	Sets position of background image (value for <a href="#">wk background position v</a> ) or bottom of element aligned with lowest element on line (value for <a href="#">wk vertical align</a> )
wk capitalize	Longint	1	Transforms first character of every word to uppercase (value for <a href="#">wk text transform</a> )
wk center	Longint	2	Centers text or image (value for <a href="#">wk background position h</a> , <a href="#">wk text align</a> , and/or <a href="#">wk tab stop types</a> )
wk circle	Longint	11	Circle-shaped glyph used (value for <a href="#">wk list style type</a> )
wk cjk ideographic	Longint	24	Plain ideographic numbers used (value for <a href="#">wk list style type</a> )
wk club	Longint	27	Club-shaped glyph used (value for <a href="#">wk list style type</a> )
wk company	String	company	Specifies a company associated with the document (string)
wk contain	Longint	-1	Scales image to largest size such that its width and height fits inside content area (value for <a href="#">wk background size h</a> and/or <a href="#">wk background size v</a> )
wk content box	Longint	2	Background clipped to content box (value for <a href="#">wk background clip</a> ) or background image starts from upper left corner of content (value for <a href="#">wk background origin</a> )
wk cover	Longint	-2	Scales background image to smallest size so that background area is completely covered by it (value for <a href="#">wk background size h</a> and/or <a href="#">wk background size v</a> )
wk custom	Longint	29	Custom marker used (value for <a href="#">wk list style type</a> )
wk dashed	Longint	3	Dashed line used (value for <a href="#">wk text linethrough style</a> and/or <a href="#">wk text underline style</a> )
wk date creation	String	dateCreation	Returns creation date of document (date). This value is read-only and cannot be set.
wk date modified	String	dateModified	Returns last modification date of document (date). This value is read-only and cannot be set.
wk decimal	Longint	3	Decimal alignment (value for <a href="#">wk tab stop types</a> ) or numbers used (value for <a href="#">wk list style type</a> )
wk decimal greek	Longint	28	Greek numerals used (value for <a href="#">wk list style type</a> )
wk decimal leading zero	Longint	13	Decimal numbers padded with initial zeros used (value for <a href="#">wk list style type</a> )
wk default	Longint	-1	Default value of property (constant) is used.
wk diamond	Longint	26	Diamond-shaped glyph used (value for <a href="#">wk list style type</a> )
wk direction	String	direction	Specifies text direction of paragraph. Possible values: <ul style="list-style-type: none"> <li><a href="#">wk left to right</a> (default)</li> <li><a href="#">wk right to left</a></li> </ul>
wk disc	Longint	10	Filled circle marker used (value for <a href="#">wk list style type</a> )
wk dotted	Longint	2	Dotted line or border used (value for <a href="#">wk border style</a> , <a href="#">wk text linethrough style</a> and/or <a href="#">wk text underline style</a> )
			Double line or border used (value for <a href="#">wk border style</a> , <a href="#">wk text</a>

wk double	Longint	4	Double line or border used (value for <a href="#">wk border style</a> , <a href="#">wk text linethrough style</a> and/or <a href="#">wk text underline style</a> )
wk dpi	String	dpi	DPI used for internal pixels <->points conversion (integer). Always 96 (read-only)
wk end text	Longint	0	Sets end of document as end of text range
wk exclude from range	Longint	1	Inserted contents not included in updated range
wk false	Longint	0	
wk font	String	font	Specifies complete font name with styles, as returned by the <b>FONT STYLE LIST</b> command. If you set an invalid font name, the command does nothing. Default value: "Times New Roman". Specifies thickness of text (depends on available font styles). Possible values: <ul style="list-style-type: none"> <li>• <a href="#">wk true</a> to set selected characters to bold font style; with the <b>WP GET ATTRIBUTES</b> command, <a href="#">wk true</a> is returned if at least one selected character supports a bold font style.</li> <li>• <a href="#">wk false</a> (default) to remove the bold font style from selected characters if any; with the <b>WP GET ATTRIBUTES</b> command, <a href="#">wk false</a> is returned if none of the selected characters supports a bold font style.</li> </ul>
wk font bold	String	fontBold	
wk font family	String	fontFamily	Specifies font family name as defined by <a href="#">wk font</a> . Default value is "Times New Roman". An empty string is returned by the <b>WP GET ATTRIBUTES</b> command if the selected characters contain different font family properties. Specifies italic style of text (depends on available font styles). Possible values: <ul style="list-style-type: none"> <li>• <a href="#">wk true</a> to set selected characters to italic or oblique font style; with the <b>WP GET ATTRIBUTES</b> command, <a href="#">wk true</a> is returned if at least one selected character supports an italic or oblique font style.</li> <li>• <a href="#">wk false</a> (default) to remove the italic or oblique font style from selected characters if any; with the <b>WP GET ATTRIBUTES</b> command, <a href="#">wk false</a> is returned if none of the selected characters supports an italic or oblique font style.</li> </ul>
wk font italic	String	fontItalic	
wk font size	String	fontSize	Specifies font size for text. Possible values (in points only): <ul style="list-style-type: none"> <li>• Real value (default = 12)</li> <li>• CSS string with value and unit concatenated. E.g.: 12pt for 12 points.</li> </ul>
wk freeze expressions	Longint	64	Freeze expressions at the moment of the insertion
wk georgian	Longint	20	Traditional Georgian numbering used (value for <a href="#">wk list style type</a> )
wk groove	Longint	6	3D grooved border used (value for <a href="#">wk border style</a> )
wk hebrew	Longint	21	Traditional Hebrew numbering used (value for <a href="#">wk list style type</a> ) Sets height of element. The height property does not include padding, borders, or margins; it sets the height of the area inside the padding, border, and margin of the element. Possible values: <ul style="list-style-type: none"> <li>• <a href="#">wk auto</a> (default): height is based upon the contents of the element</li> </ul>

- Defined size: size expressed using real or string value:
  - Real: Size in [wk layout unit](#).
  - String: CSS string with value and unit concatenated. E.g.: 12pt for 12 points, or 1.5cm for 1.5 centimeters. Minimum value: 0pt, maximum value: 10000pt.

The [wk height](#) attribute is overridden by [wk min height](#) (if defined).

**Note:** In the current implementation, [wk height](#) can only be used with pictures.

No border used. Same as none (no border) except it takes precedence over all other conflicting borders (value for [wk border style](#))

Traditional Hiragana numbering used (value for [wk list style type](#))

Hollow square glyph used (value for [wk list style type](#))

Formatted HTML code ("pretty print"), easier to debug

In this layout, any 4D Write Pro advanced attributes which are not compliant with all browsers are removed

Specifies an image. Possible values to set:

- Image URL (string). Can be absolute or relative to the structure file.
- Picture variable or field.

Value returned (**WP GET ATTRIBUTES**): URI (network URL or data URI). It may not be equal to the initial URL for an image not referenced with the network URL (only network URLs are kept). For local file URLs, the image stream itself is kept in the document and thus the URL returned is a data URI with the image stream encoded in base64.

Specifies alternative text for image, if image cannot be displayed.

Inserted contents included in updated range (default)

Inserted contents inherits character style from the paragraph default character style.

3D inset border used (value for [wk border style](#))

When the selected area contains several paragraphs, specifies that the attribute should affect only the corresponding inter-paragraph property (not outside). It applies only to border, padding and margin attributes, and must be added to the specified attribute. See example 2 of the **WP SET ATTRIBUTES** command.

Available for 4D Write Pro areas only

Traditional Katakana numbering used (value for [wk list style type](#))

Keep destination paragraph styles

Specifies unit of dimension of some attributes when value is set or

wk height String height

wk hidden Longint 5

wk hiragana Longint 22

wk hollow square Longint 25

wk html debug Longint 1

wk html wysiwyg Longint 1

wk image String image

wk image alternative text String imageAltText

wk include in range Longint 0

wk inherit style from paragraph Longint 32

wk inset Longint 8

wk inside String Inside

wk justify Longint 5

wk katakana Longint 23

wk keep paragraph styles Longint 128

get as a integer or real. Possible values:

wk layout unit	String	userUnit	<ul style="list-style-type: none"><li>• <a href="#">wk unit cm</a> (default): centimeters</li><li>• <a href="#">wk unit pt</a>: points</li><li>• <a href="#">wk unit px</a>: pixels</li><li>• <a href="#">wk unit percent</a> (only for <a href="#">wk line height</a> and <a href="#">wk background size h / wk background size v</a>)</li><li>• <a href="#">wk unit mm</a>: millimeters</li><li>• <a href="#">wk unit inch</a>: inches</li></ul>
wk left	Longint	0	Aligns text or tab to the left (value for <a href="#">wk text align</a> or <a href="#">wk tab stop types</a> ) or sets starting position of background image (value for <a href="#">wk background position h</a> )
wk left to right	Longint	0	Left-to-right text/writing direction used (value for <a href="#">wk direction</a> )
wk line break	Longint	0	Line break (in the same paragraph)  Specifies space between lines. Possible values: <ul style="list-style-type: none"><li>• <a href="#">wk normal</a> (default): use value based upon text size</li><li>• Height expressed using an integer or a string value:<ul style="list-style-type: none"><li>◦ Integer: height in <a href="#">wk layout unit</a>.</li><li>◦ String: CSS string with value and unit concatenated. E.g.: 12pt for 12 points, or 1.5cm for 1.5 centimeters. A relative value (percentage %) is supported.</li></ul></li></ul>
wk line height	String	lineHeight	
wk list auto	Longint	2147483647	Restores/applies automatic list style values
wk list font	String	listFont	Specifies complete font name, as returned by the <b>FONT STYLE LIST</b> command, to display the list item marker (but not the paragraph text). If the system does not recognize the font name, it handles the substitution. If you set an invalid font name, the command does nothing. Default value: "Times".
wk list font family	String	listFontFamily	Specifies font family name as defined by <a href="#">wk list font</a> used to display the list item marker (but not the paragraph text). Default value is "Times New Roman".  Sets starting value of an ordered list. Possible values: <ul style="list-style-type: none"><li>• <a href="#">wk auto</a> (default): starting value depends on previous list items if any.</li><li>• an integer value: starting value</li></ul>
wk list start number	String	listStartNumber	List item marker string format for left-to-right paragraph direction. If defined, it overrides default list item marker string format for the list.
wk list string format LTR	String	listStringFormatLtr	<ul style="list-style-type: none"><li>• For unordered lists: string used as list item marker (usually a single character string, e.g. "-")</li><li>• For ordered lists: string containing the "#command_5" character. "#command_5" is a placeholder for the computed number or letter(s). Default is "#command_5.", so for instance if current list item number is 15 and list style type is decimal, list item marker string will be "15."</li></ul>
wk list			List item marker string format for right-to-left paragraph direction. If defined, it overrides default list item marker string format for the list. <ul style="list-style-type: none"><li>• For unordered lists: string used as list item marker (usually a</li></ul>

<p>wk list string format RTL</p>	<p>String</p>	<p>listStringFormatRtl</p>	<p>single character string, e.g. "-")</p> <ul style="list-style-type: none"> <li>For ordered lists: string containing the "#command_5" character. "#command_5" is a placeholder for the computed number or letter(s). Default is "#command_5.", so for instance if current list item number is 15 and list style type is decimal, list item marker string will be "15."</li> </ul>
			<p>Specifies an image as the list item marker in an unordered list. Possible values:</p> <ul style="list-style-type: none"> <li><a href="#">wk none</a> (default): list item marker is not defined by an image</li> <li>Local file image URL (string). Can be absolute or relative to the database resource directory</li> </ul>
<p>wk list style image</p>	<p>String</p>	<p>listStyleImage</p>	<p>Value returned (<b>WP GET ATTRIBUTES</b>): URI (network URL or data URI). It may not be equal to the initial URL for an image not referenced with the network URL (only network URLs are kept). For local file URLs, the image stream itself is kept in the document and thus the URL returned is a data URI with the image stream encoded in base64.</p>
<p>wk list style image height</p>	<p>String</p>	<p>listStyleImageHeight</p>	<p>Sets height of image used as list item marker. Possible values:</p> <ul style="list-style-type: none"> <li><a href="#">wk auto</a> (default): height is based upon image size</li> <li>Defined size: size expressed using real or string value: <ul style="list-style-type: none"> <li>Real: Size in <a href="#">wk layout unit</a>.</li> <li>String: CSS string with value and unit concatenated. E.g.: 12pt for 12 points, or 1.5cm for 1.5 centimeters. Minimum value: 0pt, maximum value: 10000pt.</li> </ul> </li> </ul>
			<p>Specifies type of ordered or unordered list item marker. Possible values are:</p> <ul style="list-style-type: none"> <li><a href="#">wk disc</a> (default)</li> <li><a href="#">wk circle</a></li> <li><a href="#">wk square</a></li> <li><a href="#">wk decimal</a>: 1 2 3</li> <li><a href="#">wk decimal leading zero</a>: 01 02 03</li> <li><a href="#">wk lower latin</a>: a b c</li> <li><a href="#">wk lower roman</a>: i ii iii iv</li> <li><a href="#">wk upper latin</a>: A B C</li> <li><a href="#">wk upper roman</a>: I II III IV</li> <li><a href="#">wk lower greek</a>: alpha, beta, gamma, etc.</li> <li><a href="#">wk armenian</a></li> <li><a href="#">wk georgian</a></li> <li><a href="#">wk hebrew</a></li> <li><a href="#">wk hiragana</a></li> <li><a href="#">wk katakana</a></li> <li><a href="#">wk cjk ideographic</a></li> <li><a href="#">wk hollow square</a></li> <li><a href="#">wk diamond</a></li> <li><a href="#">wk club</a></li> <li><a href="#">wk decimal greek</a></li> <li><a href="#">wk custom</a>: unordered list with "-" as default list item marker; this is a convenience style used in order to customize a list item marker with <a href="#">wk list string format LTR</a> or <a href="#">wk list string format RTL</a> without modifying standard list item markers</li> <li><a href="#">wk none</a></li> </ul>
<p>wk list style type</p>	<p>String</p>	<p>listStyleType</p>	

wk lower greek	Longint	18	Lowercase classical Greek used (value for <a href="#">wk list style type</a> )
wk lower latin	Longint	14	Lowercase ASCII letters used (value for <a href="#">wk list style type</a> )
wk lower roman	Longint	15	Lowercase Roman numerals used (value for <a href="#">wk list style type</a> )
wk lowercase	Longint	2	Changes all characters to lowercase (value for <a href="#">wk text transform</a> )
wk margin	String	margin	<p>Specifies size for all margins of the element. Possible values:</p> <ul style="list-style-type: none"> <li>• Size expressed using an integer or a string value: <ul style="list-style-type: none"> <li>◦ Integer: size in <a href="#">wk layout unit</a>.</li> <li>◦ String: CSS string with value and unit concatenated. E.g.: 12pt for 12 points, or 1.5cm for 1.5 centimeters</li> </ul> </li> <li>• <a href="#">wk none</a> (default): no specific margin</li> </ul> <p>Specifies size for bottom margin of the element. Possible values:</p> <ul style="list-style-type: none"> <li>• Size expressed using an integer or a string value: <ul style="list-style-type: none"> <li>◦ Integer: size in <a href="#">wk layout unit</a>.</li> <li>◦ String: CSS string with value and unit concatenated. E.g.: 12pt for 12 points, or 1.5cm for 1.5 centimeters</li> </ul> </li> <li>• <a href="#">wk none</a> (default): no specific margin</li> </ul> <p>Specifies size for left margin of the element. Possible values:</p> <ul style="list-style-type: none"> <li>• Size expressed using an integer or a string value: <ul style="list-style-type: none"> <li>◦ Integer: size in <a href="#">wk layout unit</a>.</li> <li>◦ String: CSS string with value and unit concatenated. E.g.: 12pt for 12 points, or 1.5cm for 1.5 centimeters</li> </ul> </li> <li>• <a href="#">wk none</a> (default): no specific margin</li> </ul> <p>Specifies size for right margin of the element. Possible values:</p> <ul style="list-style-type: none"> <li>• Size expressed using an integer or a string value: <ul style="list-style-type: none"> <li>◦ Integer: size in <a href="#">wk layout unit</a>.</li> <li>◦ String: CSS string with value and unit concatenated. E.g.: 12pt for 12 points, or 1.5cm for 1.5 centimeters</li> </ul> </li> <li>• <a href="#">wk none</a> (default): no specific margin</li> </ul> <p>Specifies size for top margin of the element. Possible values:</p> <ul style="list-style-type: none"> <li>• Size expressed using an integer or a string value: <ul style="list-style-type: none"> <li>◦ Integer: size in <a href="#">wk layout unit</a>.</li> <li>◦ String: CSS string with value and unit concatenated. E.g.: 12pt for 12 points, or 1.5cm for 1.5 centimeters</li> </ul> </li> <li>• <a href="#">wk none</a> (default): no specific margin</li> </ul>
wk margin bottom	String	marginBottom	
wk margin left	String	marginLeft	
wk margin right	String	marginRight	
wk margin top	String	marginTop	
wk middle	Longint	2	Sets position of background image (value for <a href="#">wk background position v</a> ) or places element in middle of parent element (value for <a href="#">wk vertical align</a> )
wk mime html	Longint	1	<p>4D Write Pro document is saved as standard MIME HTML with HTML documents and images embedded as MIME parts (encoded in base64). Expressions are computed and 4D specific tags are removed. This format is particularly suitable for sending HTML emails with the <b>SMTP_QuickSend</b> command.</p> <p>Sets minimum height of the element. It prevents the value of the <a href="#">wk height</a> property from becoming smaller than <a href="#">wk min height</a>.</p>

Possible values:

- [wk auto](#) (default): minimum height is based upon the contents of the element
- Defined size: size expressed using real or string value:
  - Real: Size in [wk layout unit](#).
  - String: CSS string with value and unit concatenated. E.g.: 12pt for 12 points, or 1.5cm for 1.5 centimeters. Minimum value: 0pt, maximum value: 10000pt.

wk min height      String      minHeight

The [wk min height](#) value overrides the [wk height](#) attribute.

**Note:** In the current implementation, can only be used with pictures.

Sets minimum width of element. It prevents the value of the [wk width](#) property from becoming smaller than [wk min width](#). Possible values:

- [wk auto](#) (default): minimum width is based upon the contents of the element
- Defined size: size expressed using real or string value:
  - Real: Size in [wk layout unit](#).
  - String: CSS string with value and unit concatenated. E.g.: 12pt for 12 points, or 1.5cm for 1.5 centimeters. Minimum value: 0pt, maximum value: 10000pt.

wk min width      String      minWidth

The [wk min width](#) value overrides the [wk width](#) attribute.

**Note:** In the current implementation, can only be used with pictures.

wk mixed      Longint      -2147483648

Returned when there are different values for an attribute in the range or document

Specifies style sheet to use when adding a new line in the paragraph. Possible values:

wk new line style sheet      String      newLineStyleSheet

- existing style sheet name
- [wk none](#) (default)

wk no repeat      Longint      3

Background image will not be repeated (value for [wk background repeat](#))

wk none      Longint      0

wk normal      Longint      0

Standard HTML code

wk notes      String      notes

Specifies comments about the document (string).

wk outset      Longint      9

3D outset border used (value for [wk border style](#))

wk outside      String      Outside

When the selected area contains several paragraphs, specifies that the attribute should affect only the corresponding paragraph external property (not inside). It applies only to border, padding and margin attributes, and must be added to the specified attribute. See example 2 of the **WP SET ATTRIBUTES** command.

Specifies size of padding for all sides of the element. Possible values:

- Size expressed using an integer or a string value:
  - Integer: size in [wk layout unit](#).
  - String: CSS string with value and unit concatenated. E.g.: 12pt for 12 points, or 1.5cm for 1.5 centimeters

wk padding      String      padding



- [wk none](#) (default): no specific padding

Specifies size of padding for bottom of the element. Possible values:

- Size expressed using an integer or a string value:
  - Integer: size in [wk layout unit](#).
  - String: CSS string with value and unit concatenated. E.g.: 12pt for 12 points, or 1.5cm for 1.5 centimeters
- [wk none](#) (default): no specific padding

wk padding bottom      String      paddingBottom

Background clipped to padding box (value for [wk background clip](#)) or background image starts from upper left corner of padding edge (value for [wk background origin](#))

Specifies size of padding for left side of the element. Possible values:

- Size expressed using an integer or a string value:
  - Integer: size in [wk layout unit](#).
  - String: CSS string with value and unit concatenated. E.g.: 12pt for 12 points, or 1.5cm for 1.5 centimeters
- [wk none](#) (default): no specific padding

wk padding left      String      paddingLeft

Specifies size of padding for right side of the element. Possible values:

- Size expressed using an integer or a string value:
  - Integer: size in [wk layout unit](#).
  - String: CSS string with value and unit concatenated. E.g.: 12pt for 12 points, or 1.5cm for 1.5 centimeters
- [wk none](#) (default): no specific padding

wk padding right      String      paddingRight

Specifies size of padding for top of the element. Possible values:

- Size expressed using an integer or a string value:
  - Integer: size in [wk layout unit](#).
  - String: CSS string with value and unit concatenated. E.g.: 12pt for 12 points, or 1.5cm for 1.5 centimeters
- [wk none](#) (default): no specific padding

wk padding top      String      paddingTop

wk page break      Longint      2      Page break: defines a new page

wk prepend      Longint      1      Insert contents at beginning of range

wk range end      String      rangeEnd      (Read-only range attribute)

wk range owner      String      rangeOwner      (Read-only range attribute)

wk range start      String      rangeStart      (Read-only range attribute)

wk range type      String      rangeType      (Read-only range attribute) Type of 4D Write Pro range. Can be 0: default range (Default), 1: paragraph range, 2: image range

wk repeat      Longint      0      Background image repeated both vertically and horizontally (value for [wk background repeat](#))

wk repeat x      Longint      1      Background image repeated only horizontally (value for [wk background repeat](#))

wk repeat y      Longint      2      Background image repeated only vertically (value for [wk background repeat](#))

wk replace      Longint      0      Replace range contents

wk ridge      Longint      7      3D ridged border used (value for [wk border style](#))

wk right	Longint	1	Aligns text or tab to the right (value for <a href="#">wk text align</a> or <a href="#">wk tab stop types</a> ) or sets starting position of background image (value for <a href="#">wk background position h</a> )
wk right to left	Longint	1	Right-to-left direction used (value for <a href="#">wk direction</a> )
wk section break	Longint	1	Section break: defines a new section
wk semi transparent	Longint	5	Semi-transparent line used (value for <a href="#">wk text linethrough style</a> and/or <a href="#">wk text underline style</a> )
wk small uppercase	Longint	4	Transforms all characters to small uppercase (value for <a href="#">wk text transform</a> )
wk solid	Longint	1	Solid line or border used (value for <a href="#">wk border style</a> , <a href="#">wk text linethrough style</a> and/or <a href="#">wk text underline style</a> )
wk square	Longint	12	Square marker used (value for <a href="#">wk list style type</a> )
wk start text	Longint	1	Sets beginning of document as start of text range
wk style sheet	String	styleSheet	<p>Specifies current style sheet for the selected element(s). Possible values:</p> <ul style="list-style-type: none"> <li>• <a href="#">wk none</a> (default)</li> <li>• existing style sheet name</li> </ul>
wk subject	String	subject	Specifies document subject (string)
wk subscript	Longint	6	Aligns element as subscript (value for <a href="#">wk vertical align</a> )
wk superscript	Longint	5	Aligns element as superscript (value for <a href="#">wk vertical align</a> )
wk tab stop offsets	String	tabStopOffsets	<p>Specifies tab stops for the paragraph. Possible values:</p> <ul style="list-style-type: none"> <li>• Scalar value (default is 35.45pt): default offset for the whole paragraph. The <b>WP GET ATTRIBUTES</b> command returns the last offset (which is the default relative offset for offsets beyond the last absolute offset).</li> <li>• Array of tab values: an ordered list of absolute values, starting from the left margin. The tab offset defined by the last value is repeated for each additional tab character entered in the paragraph. If the tab offset is greater than the paragraph width, the text goes on the next line and starts from the first tab value. If a value in the array is smaller than the previous value, it is ignored.</li> </ul> <p><b>Note:</b> You cannot use arrays and scalars in the same call for different attributes.</p> <p>Values are expressed using CSS strings (default) or Real values in <a href="#">wk layout unit</a>. Maximum value is 10000pt.</p> <p>Specifies tab stop type for the paragraph. Possible values:</p> <ul style="list-style-type: none"> <li>• <a href="#">wk left</a> (default): text extends to the right from the tab stop</li> <li>• <a href="#">wk right</a>: text extends to the left from the tab stop until the tab's space is filled</li> <li>• <a href="#">wk center</a>: text is centered at the tab stop</li> <li>• <a href="#">wk decimal</a>: text before the decimal point extends to the left, and text after the decimal point extends to the right</li> <li>• <a href="#">wk bar</a>: a vertical line at the specified position</li> </ul>
wk tab stop types	String	tabStopTypes	

- array of tab stop type values (if tab stops have been defined through an array).

Specifies horizontal alignment of text in the paragraph. Possible values:

wk text align      String      textAlign

- [wk left](#) (default)
- [wk right](#)
- [wk justify](#)
- [wk center](#)

Specifies color of text. Possible values:

wk text color      String      color

- a CSS color ("command\_5010101" or "command\_5FFFFFFF" or "red"). Default is "command\_5000000" if string.
- a 4D color longint value (see **OBJECT SET COLOR** command)
- a longint array containing an element for each R, G, B component (0-255)

Specifies indentation of first line in the paragraph. Possible values:

wk text indent      String      textIndent

- Real: Size in [wk layout unit](#). Default is 0.
- String: CSS string with value and unit concatenated. E.g.: 12pt for 12 points, or 1.5cm for 1.5 centimeters. Minimum value: 0pt, maximum value: 10000pt.

Specifies color of text linethrough. Possible values:

wk text linethrough color      String      textLinethroughColor

- a CSS color ("command\_5010101" or "command\_5FFFFFFF" or "red").
- a 4D color longint value (see **OBJECT SET COLOR** command)
- a longint array containing an element for each R, G, B component (0-255)

Default is "currentColor" if string, or [wk default](#) if longint.

Specifies style of text linethrough (if any). Possible values:

wk text linethrough style      String      textLinethroughStyle

- [wk none](#) (default): no linethrough effect
- [wk solid](#): draw a solid line on the selected text
- [wk dotted](#): draw a dotted line on the selected text
- [wk dashed](#): draw a dashed line on the selected text
- [wk double](#): draw a double line on the selected text
- [wk semi transparent](#): dimmed line on the selected text. Can be combined with another line style.
- [wk word](#): draw a line on words only (exclude blank spaces). Can be combined with another line style.

Specifies shadow color of the selected text. Possible values:

wk text shadow color      String      textShadowColor

- a CSS color ("command\_5010101" or "command\_5FFFFFFF" or "red").
- a 4D color longint value (see **OBJECT SET COLOR** command)
- a longint array containing an element for each R, G, B component (0-255)
- [wk transparent](#) (default)

Specifies offset for shadow effect. Possible values:

wk text shadow offset      String      textShadowOffset

- Size expressed in points. Default value: 1pt

Specifies uppercase and lowercase letters in the text. Possible values:

wk text transform      String      textTransform

- [wk capitalize](#): first letters are set to uppercase
- [wk lowercase](#): letters are set to lowercase
- [wk uppercase](#): letters are set to uppercase
- [wk small uppercase](#): letters are set to small uppercase
- [wk none](#) (default): no transformation

Specifies color of text underline. Possible values:

wk text underline color      String      textUnderlineColor

- a CSS color ("[#command\\_5010101](#)" or "[#command\\_5FFFFFFF](#)" or "red").
- a 4D color longint value (see **OBJECT SET COLOR** command)
- a longint array containing an element for each R, G, B component (0-255)

Default is "currentColor" if string, or [wk default](#) if longint.

Specifies style of text underline (if any). Possible values:

wk text underline style      String      textUnderlineStyle

- [wk none](#) (default): no underline
- [wk solid](#): draw a solid underline
- [wk dotted](#): draw a dotted underline
- [wk dashed](#): draw a dashed underline
- [wk double](#): draw a double underline
- [wk semi transparent](#): dimmed underline. Can be combined with another line style.
- [wk word](#): draw an underline for words only (exclude blank spaces). Can be combined with another line style.

wk title      String      title

Specifies document title (string). Default is "New 4D Write Pro Document"

wk top      Longint      0

Sets position of background image (value for [wk background position v](#)) or aligns element with top of tallest element on the line (value for [wk vertical align](#))

wk transparent      Longint      -1

Specifies color is transparent (value for [wk background color](#) or [wk text shadow color](#))

wk true      Longint      1

wk unit cm      String      cm

Unit used is centimeters (value for [wk layout unit](#))

wk unit inch      String      in

Unit used is inches (value for [wk layout unit](#))

wk unit mm      String      mm

Unit used is millimeters (value for [wk layout unit](#))

wk unit percent      String      [#command\\_537](#);

Unit used is a percentage (value for [wk layout unit](#))

wk unit pt      String      pt

Unit used is points (value for [wk layout unit](#))

wk unit px      String      px

Unit used is pixels (value for [wk layout unit](#))

wk upper latin      Longint      16

Uppercase ASCII letters used (value for [wk list style type](#))

wk upper roman      Longint      17

Uppercase Roman numerals used (value for [wk list style type](#))

wk uppercase      Longint      3

Changes all characters to uppercase (value for [wk text transform](#))

wk value unit not percentage      Longint      -100000

Returned to the **WP GET ATTRIBUTES** command when the current value unit is not percentage and you passed a string variable (result is invalid).

wk value unit percentage	Longint	-100001	Returned to the <b>WP GET ATTRIBUTES</b> command when the current value unit is percentage and you passed a numeric variable (result is invalid).
wk version	String	version	Returns internal 4DWP version of the document (real). This number is only read using <b>WP GET ATTRIBUTES</b> ; it cannot be set.
wk vertical align	String	verticalAlign	<p>Sets vertical alignment of an element. Can be used with characters, paragraphs, and pictures. Possible values:</p> <ul style="list-style-type: none"> <li>• <a href="#">wk baseline</a> (default): aligns baseline of element with baseline of parent element</li> <li>• <a href="#">wk top</a>: aligns top of element with top of tallest element on the line</li> <li>• <a href="#">wk bottom</a>: aligns bottom of element with lowest element on the line</li> <li>• <a href="#">wk middle</a>: element is placed in middle of parent element</li> <li>• <a href="#">wk superscript</a>: aligns element as if it were superscript</li> <li>• <a href="#">wk subscript</a>: aligns element as if it were subscript</li> </ul> <p>For characters, <a href="#">wk top</a> and <a href="#">wk bottom</a> have the same effect as <a href="#">wk baseline</a>.</p> <p>For paragraphs, <a href="#">wk baseline</a>, <a href="#">wk superscript</a> and <a href="#">wk subscript</a> have the same effect as <a href="#">wk top</a>.</p>
wk web page complete	Longint	2	.htm or .html extension. Document is saved as standard HTML and its resources are saved separately. 4D tags are removed and expressions are computed. This format is particularly suitable when you want to display a 4D Write Pro document in a web browser.
wk web page html 4D	Longint	3	4D Write Pro document is saved as HTML and includes 4D specific tags; each expression is inserted as a non-breaking space. Since this format is lossless, it is appropriate for storing purposes in a text field.
wk width	String	width	<p>Sets width of element. Possible values:</p> <ul style="list-style-type: none"> <li>• <a href="#">wk auto</a> (default): width is based upon the contents of the element</li> <li>• Defined size: size expressed using a real or string value: <ul style="list-style-type: none"> <li>◦ Real: Size in <a href="#">wk layout unit</a>.</li> <li>◦ String: CSS string with value and unit concatenated. E.g.: 12pt for 12 points, or 1.5cm for 1.5 centimeters. Minimum value: 0pt, maximum value: 10000pt.</li> </ul> </li> </ul> <p>The <a href="#">wk width</a> attribute is overridden by <a href="#">wk min width</a> if defined.</p> <p><b>Note:</b> In the current implementation, <a href="#">wk width</a> can only be used with pictures.</p>
wk word	Longint	6	Underline words only (exclude blank spaces) (value for <a href="#">wk text linethrough style</a> and/or <a href="#">wk text underline style</a> )