

4D Write

-  4D Writeランゲージの紹介
-  WRエリア
-  WRエリアオプション
-  WRエリアコントロール
-  WRスタイルシート
-  WRタブ
-  WRデータベースオブジェクト
-  WRテキスト操作
-  WRドキュメント
-  WRドラッグアンドドロップ
-  WRピクチャコントロール
-  WRプリント
-  WRユーティリティ
-  定数のリスト
-  付録
-  コマンドリスト (文字順)

✿ 4D Writeランゲージの紹介

- ✿ はじめに
- ✿ マルチプラットフォームドキュメントの管理
- ✿ 表記方法について
- ✿ メソッドエディタ内のコマンド
- ✿ 4D Writeエリア内のドキュメント
- ✿ 4D Writeメニュー項目
- ✿ 文字の参照

4D Writeは、4Dにワードプロセッシング用のコマンドと機能を付加するプラグインです。これらのコマンドを利用することで、(下記に挙げるような) 定型的な作業を自動化できます。

- メニューの実行
- ドキュメントを開く／ドキュメントの保存
- ドキュメントのマージン設定
- 表示属性の設定

4Dに追加されるすべての4D Writeコマンドには、先頭にWRという文字がついています。これにより、4Dや4D Write以外のプラグインのコマンドと区別できます。

4D Writeのドキュメント

4D Writeのドキュメントは、*4D Write*ユーザリファレンスと*4D Write*ランゲージリファレンスという2つのマニュアルから成り立っています。本マニュアル (*4D Write*ランゲージリファレンス) は、4D Writeのプログラミングランゲージの使用方法について述べています。4D Writeの使い方に関する情報は*4D Write*ユーザリファレンスを参照してください。

🌿 マルチプラットフォームドキュメントの管理

4D Write自身は、4Dや4D Serverと同様に、マルチプラットフォームプログラムです。つまり、Mac OS上で作成された4D Writeを利用しているデータベースを、何の修正も加えずにWindows上で稼働させること、またはその逆も可能であるということです。これが可能であるのは同一バージョンのソフトウェアをお使いの場合に限られます。しかし、4Dデータベースと4D Writeドキュメントのマルチプラットフォーム管理においては、Mac OSとWindowsのOS間の違いを考慮する必要があります。

Mac OSとWindowsのファイル対応

次の表は、MacintoshとWindowsでの4D Writeドキュメントのファイル対応を示しています。

ドキュメント	Mac OSタイプ	クリエーター	Windows拡張子	仮想タイプ (*)
4D Writeドキュメント	4WR7	4DW7	4W7	4WR7
RTF	TEXT	4DW7	RTF	RTF
Windowsテキストのみ	TEXT	4DW7	TXT	ASCW
Mac OSテキストのみ	TEXT	4DW7	TXT	ASCM
Unicodeテキストドキュメント	TEXT	4DW7	TXT	ASCU
HTMLドキュメント	TEXT	MOSS	HTML	HTML
Word 6/95ドキュメント	W6BN	MSWD	DOC	DOC6
Word 97 PC/98 Mac	W8BN	MSWD	DOC	DOC8

(*) これらのタイプはWR OPEN DOCUMENTやWR SAVE DOCUMENTコマンドでのみ使用されます。

ドキュメント

下記のルールについて認識しておく必要があります。

- Mac OS上で、4D Writeはドキュメントを見分けるためにタイプとクリエータを利用します。例えばタイプが4WR7でクリエータが4DW7であれば、4D Writeドキュメントです。完全なアクセスパスは、それぞれがコロン (:) で区切られたディスク名、フォルダ名、ドキュメント名を含みます。例えば“MyMac:フォルダ1:フォルダ2:MyDatabase”というようなものです。
- Windows上では、4D Writeはドキュメントを見分けるためにファイル名の拡張子を利用します。例えば、4W7であれば4D Writeドキュメントです。完全なアクセスパスは、それぞれが円マーク (¥) で区切られたディスク名、フォルダ名、ドキュメント名を含みます。例えば、“D:¥フォルダ1¥フォルダ2¥MyDatabase”というようなものです。
- Mac OS上で作成されWindows上にコピーされた4D Writeドキュメントは、ファイル名が拡張子付きで保存すれば、そのまま開くことができます。例えばMyDoc.4W7という名称で保存されたMyDocドキュメントは、PCのボリュームにコピーされた後も何も手を加えずに開くことができます。
- Windows上で作成されMac OS上にコピーされた4D Writeドキュメントは、何も手を加えないで開くことができます。

テンプレート

Mac OSとWindowsのクライアント間でサーバのプラットフォームに関係なくテンプレートを共有するための手順を次に示します。

テンプレートのファイル名をAreaName_.4WTとします。

テンプレートは、4Dおよび4D Serverでデータベースフォルダに保存されます（サーバ上で保存する場合、これがデフォルトオプションです）。

WR SET AREA PROPERTYを使用して、4D Serverでテンプレートを（クライアントマシン上に）ローカルに格納する場合は、OSによってそれぞれ以下の場所に格納されます：

- Mac OS上では、**Library:Application Support:4D:4D Write Templates:DatabaseName**
- Windows上では、**Documents and settings¥UserName¥Application data¥4D¥4D Write Templates¥DatabaseName**

表記方法について

本マニュアルで、4D Writeコマンドは特殊フォントを使用した大文字で表記されています（例：*WR ON COMMAND*）。4D Write関数は、先頭文字のみが大文字で表記されています（例：*WR Get styled text*）。

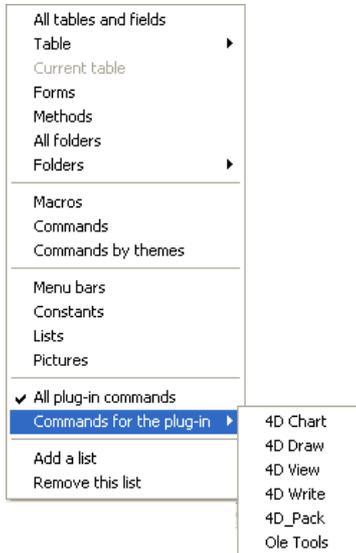
4D Writeコマンドや4D Write関数がメソッドやオブジェクトメソッド中に表示される際には、4Dの標準のコマンドや関数と区別するために、イタリックの書体で表示されます。イタリックでないテキストは、4Dランゲージの単語であることを示します。

```
QUERY ([Templates]; [Templates]ID=vNumber) \ 4Dコマンド
If(Records in selection([Templates])=1)
  WR PICTURE TO AREA(Area; [Templates]Doc) \ 4D Writeコマンド
End if
```

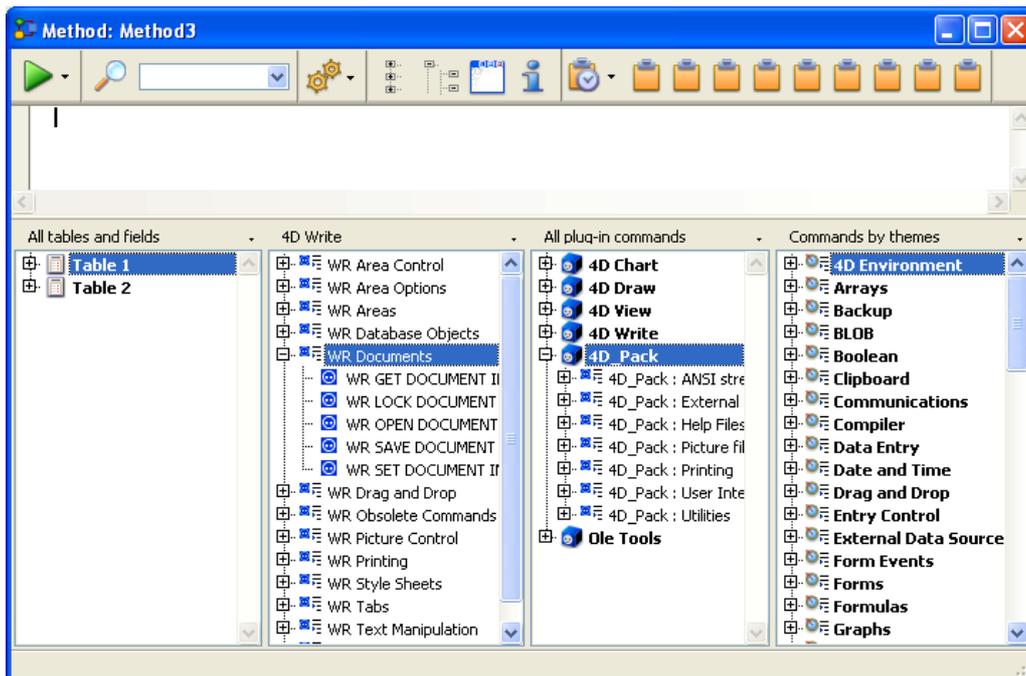
本マニュアルのいくつかの例題において、余白の都合でコードの行が2行目または3行目へと続くものがあります。しかし、これらの例題を入力する場合には、そのコードは1行で記述してください（途中で改行を入力しないでください）。

🌿 メソッドエディタ内のコマンド

4D Writeのコマンドは、4Dメソッドエディタ内表示されます。リストには4D Writeの他、他の利用可能なプラグインコマンドも含まれます：



プラグインコマンドはテーマごとに階層化されます：



プラグインはエクスプローラのプラグインページにも表示されます。

Note: プラグインの定数は標準の4D定数リストに追加されます。

4Dコマンドを配置するのと同じように、4D Writeコマンドをメソッド中に配置することができます。メソッド内に直接入力したりリスト中のコマンド名をダブルクリックしたりできます。

4D Writeコマンドは、すべてのメソッドタイプ、すなわちプロジェクト、トリガ、フォーム、オブジェクトおよびデータベースメソッドで利用できます。その中でも4D Writeコマンドは、4D Writeのドキュメントエリアと同じフォーム上のオブジェクトでアクティブ化されたオブジェクトメソッド内において、特に効果的です。

🌱 4D Writeエリア内のドキュメント

4D Writeドキュメントは、下記の4Dの3つのエリアで使用可能です。

- フォーム内のプラグインエリア
- プラグインウィンドウ
- オフスクリーンエリア

4D Writeドキュメントを利用するには、フォーム上にプラグインエリアを作成するか、またはプラグインウィンドウを開きます。プラグインエリアを作成するには、デザインモードでフォーム上にエリアを描画します。プラグインウィンドウを開くには、**ツールメニュー**から**4D Write**を選択するか、**Open external window**を実行します。

可視エリアを作成することに加えて、不可視オフスクリーンエリアを作成することもできます。詳細は後述の"4D Writeオフスクリーンエリア"を参照してください。

4D WriteエリアのID番号と変数

4D Writeは4D Writeエリア、プラグインウィンドウおよびオフスクリーンエリアの参照を保持するために変数を使います。コマンドや関数にエリアIDを代入した変数を引数として渡し、操作対象のエリアを参照します。後述のコマンドの説明において、*area*引数はドキュメントエリアを特定する変数を参照します。

次の2つのタイプの変数があります。

- プラグインオブジェクト名
4D Writeエリアを作成し変数名を付けると、4Dはエリアを参照するための変数として4D Writeエリアの名前を自動的に認識します。例えば、vLetterという名前のエリアを処理の対象にする場合には、*area*にvLetterを指定します。
- プラグインウィンドウまたはオフスクリーンエリアのために作成した変数
Open external windowや**WR New offscreen area**を使用してプラグインウィンドウやオフスクリーンエリアを作成した場合、関数から返されるエリアID番号を変数に受け取ることができます。他のコマンドや関数でこの変数を使用し、プラグインウィンドウやオフスクリーンエリアに対して処理を行うことができます。値を変数に受け取るには、コード内の行の関数の左側に変数名と代入記号(=)を記述します。

ほとんどの4D Writeコマンドは、実行するために*area*を指定する必要があります。

4D Writeプラグインエリア

4D Writeドキュメントを4Dのフォーム内に表示するためには、フォーム上にプラグインエリアを作成し、他と重複しない変数名を割り当て、プラグインエリアのタイプとして4D Writeを指定します。

4Dでは、このドキュメントをレコードと一緒に保存することができます。

プラグインエリアは、ドキュメントを保存するためによく使用されます。また書式が重要な場合は、テキストフィールドの代わりにプラグインエリアを使用することもあります。

4D Writeプラグインウィンドウエリア

4Dでは、プラグインウィンドウと呼ばれる独立したエリアに4D Writeドキュメントを作成することができます。プラグインウィンドウは、ワードプロセッサとして手紙やメモ、その他のドキュメントにユーザにアクセスさせたい場合に便利です。

メソッドから4D関数**Open external window**を実行すると、指定されたウィンドウを開くと同時に倍長整数の変数にエリアIDを返します。この変数を利用すれば、4D Writeコマンドを実行したい時にはいつでもプラグインウィンドウを指定することができます。

例えば:

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

Open external windowについては、*4D Language Reference*マニュアルを参照してください。

4D Writeオフスクリーンエリア

オフスクリーンエリアはメモリ上に格納され、プログラマやユーザは見ることはできません。必要に応じて、ユーザに表示する前にドキュメントを修正したり、ドキュメントをコピーして元の状態に戻ることができるようにするためにオフスクリーンエリアを使うことができます。

*WR New offscreen area*と*WR PICTURE TO AREA*は、オフスクリーンエリアを作成する際に用います。これらのコマンドの使用後にオフスクリーンエリアを消去し、使用しているメモリを解放することを忘れないでください。

ドキュメントを保存するためにオフスクリーンエリアを作成する場合は、下記のコードをグローバルメソッドに記述します：

```
QUERY ([Employee]; [Employee]ID=vID)
If (Records in selection ([Employee]=1)
  Area:=WR New offscreen area
  WR PICTURE TO AREA (Area; [Employee]Review_)
  `Review の内容をオフスクリーンエリアに格納
  MODIFY RECORD ([Employee])
  `従業員レコードを更新
  WR DELETE OFFSCREEN AREA (Area)
  `オフスクリーンエリアで使用していたメモリを解放
End if
```

フォーム上のボタンを使用して、保存されている元のドキュメントに復帰させることも可能です。

入力フォーム上にボタンを作成し、次のコードを記述します：

```
Review:=WR Area to picture (Area)
`オフスクリーンエリアに格納しておいたオリジナルのドキュメントを
`フォーム上のエリアにコピーする
```

🌱 4D Writeメニュー項目

メソッドを使用して4D Writeメニューにアクセスし、任意のメニュー項目を選択/実行することができます。またメソッドを使用してメニューやメニュー項目の状態を決定することができます。各メニュー項目にはそれぞれ重複しない異なる整数値が割り振られています。それぞれのメニュー項目と対応する整数値の一覧は**付録B: メニュー項目番号**を参照してください。

メニュー項目に割り振られた整数値は、一般的にはメニューとメニュー項目の位置が基本となっています。メニューは左から右の順に昇順で番号が付けられます。例えば**ファイル**メニューは100、**編集**メニューは200となります。同様に、メニュー項目は上から下の順に昇順で番号が付けられています。

これらのメニュー項目の番号は、4D Writeの新しいバージョンで新しいメニュー項目が追加された場合でも変わりません。新規のメニュー項目は、現行のメニュー項目の間に配置されたとしても、別の番号を使用します。この番号付けは一般的なルールから外れますが、メソッド内で使用するメニューの参照番号は確実に残るので、参照番号を更新しなくて済みます。

ドキュメント内の文字は一連番号で参照されます。文字を参照するコマンドは、1文字または一連の文字列を指定することができます。例えば単語、一文、選択されたテキストのブロック全体を指定することができます。

*WR GET SELECTION*を使用して、4D Writeエリア内で選択された文字の位置を取得できます。このコマンドは\$Firstと\$Lastを使用して、選択された一連の文字列を参照します。\$Firstは常に選択された文字の先頭よりも1小さくなります。\$Lastは選択した最後の文字の位置と同じになります。

例題

例えば下記のコードを使用すると、Area内で選択されているテキストの位置を変数\$Firstと\$Lastに返します:

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

4D Writeエリア内のテキストを選択するには、文字を参照する必要があります。ほとんどの場合、コマンドを使用して操作を行う前に、先にテキストを選択する必要があります。

WRエリア

 エリアコマンドについて

 WR Area to blob

 WR Area to picture

 WR BLOB TO AREA

 WR DELETE OFFSCREEN AREA

 WR New offscreen area

 WR PICTURE TO AREA

✚ エリアコマンドについて

このテーマのコマンドや関数を使用して、4Dフォームやオフスクリーン上、およびBLOBやピクチャフィールドに格納された4D Writeエリアを管理できます。

例えば、*WR PICTURE TO AREA*コマンドは引数に渡されたピクチャから4D Writeエリアに4D Writeドキュメントをロードします。

WR Area to blob

WR Area to blob (area ; savedDoc) -> 戻り値

引数	型	説明
area	倍長整数	⇒ 4D Writeエリア
savedDoc	整数	⇒ 1 = ドキュメントが保存されていない場合でもダイアログも表示しない、0 = ドキュメントが保存されていない場合ダイアログを表示する
戻り値	BLOB	⇒ エリアのコンテンツ

説明

WR Area to blobはareaで指定されたエリアの内容をBLOBフィールドまたは変数に代入します。WR Area to blobは、BLOBフィールドまたは変数に代入可能なBLOBを返します。

- savedDocが0の場合で最後に保存されてからドキュメントが変更されている場合、ドキュメントを保存するかどうかを確認するダイアログが表示されます。
- savedDocが1の場合、ドキュメントは保存済みであると見なされるため、ユーザーに警告は表示されません。
- savedDocが省略されている場合、デフォルトの設定を使用します。

例題

Areaの内容をBLOBフィールド“WriteBlobSave”に格納します:

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

WR Area to picture

WR Area to picture (area ; savedDoc ; preview) -> 戻り値

引数	型	説明
area	倍長整数	⇒ 4D Writeエリア
savedDoc	整数	⇒ 1 = ドキュメントが保存されていない場合でもダイアログも表示しない、0 = ドキュメントが保存されていない場合ダイアログを表示する
preview	整数	⇒ 1 = ピクチャを作成する、0 = ピクチャを作成しない
戻り値	ピクチャ	⇒ エリアのコンテンツのピクチャ

説明

WR Area to pictureを使用して、areaによって指定されたエリアの内容をピクチャフィールドや変数に代入できます。4D WriteエリアをWR Area to pictureに渡すことで、ピクチャフィールドや変数に代入可能なピクチャが返されます。

savedDoc:

- savedDocが0の場合で最後に保存されてからドキュメントが変更されている場合、ドキュメントを保存するかどうかを確認するダイアログが表示されます。
- savedDocが1の場合、ドキュメントは保存済みであると見なされるため、ユーザに警告は表示されません。

preview:

- previewが0の場合、ピクチャプレビューは作成されません。
- previewが1の場合、ピクチャプレビューが作成されます。

Note: ピクチャプレビューを作成していない場合、ピクチャは表示されません。

オプション引数が省略されている場合、areaのデフォルトの設定を使用します。

例題 1

Areaとそのプレビューピクチャをピクチャフィールド“WritePictSave”に格納します:

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

例題 2

[Templates]テーブルのカレントレコードに現在選択されているテキストを保存します:

```
WR EXECUTE COMMAND(Area;wr_cmd_copy) `選択されているテキストをコピー
CREATE RECORD ([Templates]) ` [Templates]レコードを作成
Tempo:=WR New offscreen area ` オフスクリーンエリアを作成
WR EXECUTE COMMAND(Tempo;wr_cmd_paste) ` エリアにテキストをペースト
`結果を[Templates]Text_ fieldに格納
[Templates]Text_ :=WR Area to picture(Tempo)
WR DELETE OFFSCREEN AREA(Tempo) ` オフスクリーンエリアを削除
SAVE RECORD ([Templates]) ` [Templates]レコードを保存
```

WR BLOB TO AREA (area ; BLOB)

引数	型		説明
area	倍長整数	→	4D Writeエリア
BLOB	BLOB	→	4D Writeデータを含む変数またはフィールド

説明

WR BLOB TO AREAは、*area*で指定された4D Writeエリアに*blob*の内容を代入します。BLOBの内容は4D Writeのデータと見なされます。

Blobの内容は、名前を使用してBLOBに自動で割り当てられ保存された4D Writeエリアや*WR Area to blob*を使用してBlobに保存されたデータです。

例題 1

"[Templates]Reference_"BLOBフィールドに保存された手紙のテンプレートをロードし、カレントテンプレートとして使用したいものとします。 :

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

例題 2

"[Templates]TheText_"BLOBフィールドに保存されたテキストをコピーし、スクリーン上のカレントエリアにペーストしたいものとします。この例題は高度な用語システムの作成方法を示しています:

```
Temp:=WR New offscreen area
WR BLOB TO AREA (Temp; [Templates]TheText_) `フィールドの内容を展開
WR EXECUTE COMMAND (Temp; wr_cmd_select_all)
WR EXECUTE COMMAND (Temp; wr_cmd_copy)
WR DELETE OFFSCREEN AREA (Temp) `エリアを削除
WR EXECUTE COMMAND (Area; wr_cmd_paste) `ペーストメニューコマンドを実行
```

Note: ピクチャフィールドの中に4D Writeエリアを保存したい場合は、*WR PICTURE TO AREA*コマンドの説明を参照してください。

WR DELETE OFFSCREEN AREA

WR DELETE OFFSCREEN AREA (area)

引数	型		説明
area	倍長整数	⇒	4D Writeエリア

説明

*WR DELETE OFFSCREEN AREA*は、*WR New offscreen area*で作成された4D Writeエリアを削除し、オフスクリーンエリアが使用していたメモリを解放します。

*area*は、フォーム上やウィンドウ内のエリアではなく、オフスクリーンエリアでなければなりません。*WR DELETE OFFSCREEN AREA*はオフスクリーンエリアが不要になったときに使用してください。

例題

次の例は、*WR New offscreen area*を呼び出した後には、対となる*WR DELETE OFFSCREEN AREA*も呼び出す必要があるという例です。

```
NewArea:=WR New offscreen area
  `新しいオフスクリーンエリアを作成
  `何かを行う
WR DELETE OFFSCREEN AREA(NewArea)
  `オフスクリーンエリアを削除
```

WR New offscreen area

WR New offscreen area -> 戻り値

引数	型	説明
戻り値	倍長整数	4D Writeエリア参照

説明

*WR New offscreen area*は、ユーザからは見ることのできない4D Writeエリアのためにメモリを確保します。また、この関数は見ることができないエリアにアクセスするための参照番号を返します。*WR New offscreen area*によって返される値は、4D Writeエリアに対するすべての4D Writeコマンドで使用可能です。

この関数を用いて作成したオフスクリーンエリアは、使い終わったら削除することを忘れないでください。

例題

次の例は一時的なオフスクリーンエリアを作成し、印刷後にそれを削除します。

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

WR PICTURE TO AREA (area ; picture)

引数	型		説明
area	倍長整数	⇒	4D Writeエリア
picture	ピクチャー	⇒	フィールドまたは変数

説明

WR PICTURE TO AREAを使用することにより、4D Writeドキュメントを含んだピクチャ変数やフィールドを読み取り、areaによって指定された4D Writeエリアにその内容を開くことができます。areaには表示されているエリアでもオフスクリーンエリアでも指定することができます。

例えばこのコマンドを利用すれば、別のテーブルに保存された4D Writeドキュメントを読み取ることも可能です。

Note: このコマンドは、4D Writeのバージョン6.0.xのファイルフォーマットも読み取ることができます。

例題 1

"[Templates]Reference"ピクチャフィールドに保存した手紙のテンプレートをロードして、カレントテンプレートとして使用したいものとします:

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

例題 2

"[Templates]TheText_"ピクチャフィールドに保存されたテキストをコピーし、スクリーン上のカレントエリアにペーストしたいものとします。この例題は高度な用語システムの作成方法を示しています。:

```
Temp:=WR New offscreen area
WR PICTURE TO AREA (Temp; [Templates]TheText_) `フィールドの内容を展開
WR EXECUTE COMMAND (Temp; wr_cmd_select_all)
WR EXECUTE COMMAND (Temp; wr_cmd_copy)
WR DELETE OFFSCREEN AREA (Temp) `エリアを削除
WR EXECUTE COMMAND (Area; wr_cmd_paste) `ペーストメニューコマンドを実行
```

Note: BLOBフィールドの中に4D Writeエリアを保存した場合は、WR BLOB TO AREAコマンドを使用します。

WRエリアオプション

 エリアオプションコマンドについて

 WR Build preview

 WR GET AREA PROPERTY

 WR GET CURSOR COORDINATES

 WR GET CURSOR POSITION

 WR Get frame

 WR SET AREA PROPERTY

 WR SET CURSOR POSITION

 WR SET FRAME

 WR TEXT ACCESS

✦ エリアオプションコマンドについて

このテーマのコマンドや関数を使用して、ユーザが利用可能な環境を設定することができます。例えば *WR SET CURSOR POSITION* コマンドを使用すれば、4D Writeドキュメントの任意の場所にカーソルを置くことができます。

またユーザに対し4D Writeエリアの更新を禁止したり (*WR TEXT ACCESS*)、エリアのピクチャプレビューを構築 (*WR Build preview*) したりできます。

WR Build preview (area ; page) -> 戻り値

引数	型		説明
area	倍長整数	→	4D Writeエリア
page	倍長整数	→	ピクチャに変換するページ
戻り値	ピクチャー	↩	ページのピクチャ

説明

WR Build previewは、pageに渡された番号のページをピクチャに変換します。このページ番号は環境設定ダイアログで定義されたページの番号付けを考慮します。

ピクチャは、4Dのピクチャフィールドや変数に格納できます。このピクチャはページと同じサイズです。WR SET DOC PROPERTYコマンドで定数wr paper widthとwr paper heightを使うことによって、ピクチャのサイズを調整できます。

Note: WR Area to pictureと異なり、このピクチャは4D Writeデータを含んでいません。

返されるピクチャは、ベクトルベースのピクチャです。Windows上で作られたピクチャはEMFフォーマットを使用しているため、Mac OS上で直接表示したり、そのまま (WRITE PICTURE FILEコマンドを使用するなどして) ピクチャファイルに格納したりすることはできません。WindowsピクチャをMac OS上や他のWindowsアプリケーションで表示したい場合は、以下のコードを使用してピクチャをビットマップに変換する必要があります:

```
myPicture:=myPicture|myPicture
```

EMF (Windowsのみ)と違って、PICTおよびビットマップのピクチャタイプはプラットフォームに依存しません。

Note: 逆にMac OSピクチャは直接使用することができます。

例題

4D WriteドキュメントはBLOBフィールドに保存されています。各ドキュメントの2ページ目だけをプリントしたい場合、プリントフォームに (例えばMyImageという名前の) ピクチャ変数を挿入し、その変数に以下のメソッドを埋め込みます。:

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

以下のプロジェクトメソッドを作成し、実行します:

```
QUERY ([MyTable]) `印刷するセレクションを作成
OUTPUT FORM ([MyTable];"PrintPage2") `PrintPage2は印刷に使用するフォーム
`上記のメソッドで使用するオフスクリーンエリアを作成
NewOffscreen:=WR New offscreen area
PRINT SELECTION ([MyTable]) `セレクションを印刷
WR DELETE OFFSCREEN AREA(NewOffscreen) `オフスクリーンエリアを削除
```

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

引数	型		説明
area	倍長整数	→	4D Writeエリア
option	整数	→	オプション番号
value	整数	←	オプションに対応する現在値
stringValue	文字	←	プロパティ値の文字列

説明

WR GET AREA PROPERTY コマンドを使用して *area* で参照される 4D Write エリアの *option* の現在の *value* を読み取ることができます。

option には **WR Area properties** テーマの定数を渡します。それぞれの定数の説明は *WR SET AREA PROPERTY* コマンドの説明を参照してください。

stringValue 引数は *wr window title* と *wr minimized button title* プロパティで使用できます。 .

例題

エリアが更新されたかどうかを調べる:

```
WR GET AREA PROPERTY(WriteArea;wr_modified;$ve_report)
$vb_writeMODIF:=( $ve_report=wr_dirty_bit_status true)
```

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

引数	型		説明
area	倍長整数	⇒	4D Writeエリア
posHoriz	実数	←	ページ中の横位置
posVert	実数	←	ページ中の縦位置
height	実数	←	カーソルの高さ

説明

WR GET CURSOR COORDINATESはカーソルの座標を、ページ左上からの相対位置で返します。これらの値はドキュメントのカレントのデフォルト単位で表されます。

エリア中でテキストやピクチャが選択されているときにコマンドが実行されると、2つのケースが発生します:

- 選択がプログラムを使用して行われている場合、カーソルは選択の最後に設定されているものとして扱われます。
- 選択が手作業で行われている場合、カーソルはマウスボタンが放された場所に設定されているものとして扱われます。例えば段落が最後の行から最初の行までマウスのドラッグによって選択された場合、カーソルの位置は段落の先頭に設定されます。

height引数にはカーソルの現在の高さが返されます。ピクチャのみが選択されている場合、ピクチャの高さが返されます。

例題

Print formコマンドを使用して4D Writeエリアを印刷できます。原則として、これらのエリアは固定サイズで印刷されます。この例題では4DのプリントコマンドとWR GET CURSOR COORDINATESコマンドを使用して、内容に応じた高さの4D Writeエリアを印刷する方法を示します。

- Print formコマンドから呼び出されるフォームメソッドは以下の通り:

```
If (Form event=On Printing Detail)
  GET OBJECT RECT (4DWriteArea; $left; $top; $right; $bottom)
  $markerpos:=Get print marker (Form detail)
  $areaheight:=$bottom-$top ` 4D Writeエリアの高さ
  $newheight:=sizecalcul
  ` sizecalcul メソッドはその内容に基づき4D Writeエリアの高さを返す
  ` このメソッドは後述
  $shift:=$newheight-$areaheight
  MOVE OBJECT (4DWriteArea; 0; 0; 0; $shift) ` 4D Writeエリアをリサイズ
  SET PRINT MARKER (Form detail; $markerpos+$shift) ` マーカーを移動
End if
```

- 以下はsizecalculメソッドです:

```
$area:=WR New offscreen area
WR BLOB TO AREA ($area; [Table 1]Write_)
WR SET DOC PROPERTY ($area; wr_unit; 2) &NBSP; &NBSP; ` ピクセルを単位とする

WR SET SELECTION ($area; 1; 1) &NBSP; &NBSP; ` テキストの開始
WR GET CURSOR COORDINATES ($area; $hor; $startvert; $cursor1)
WR SET SELECTION ($area; 1000000; 1000000) &NBSP; &NBSP; ` テキストの終わり
WR GET CURSOR COORDINATES ($area; $hor; $vert; $cursor2)

WR DELETE OFFSCREEN AREA ($area)
$0:=Trunc (( $vert-$startvert+$cursor1+$cursor2) * 0.75; 0)
```

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

引数	型		説明
area	倍長整数	→	4D Writeエリア
page	倍長整数	←	選択部分の属するページ番号
column	倍長整数	←	選択部分の段組数
line	倍長整数	←	段組内の行番号
position	倍長整数	←	対象となる段組内の選択位置

説明

WR GET CURSOR POSITIONは、areaで指定された4D Writeエリア内の選択された部分の位置を返します。

- page: 返されるページは、ドキュメントの最初のページ番号から最後のページ番号まで間の番号です。これらのページ番号は、設定されていればカスタムページ番号が考慮されます。
- column: この値は1から全段組数までの間の数値です。
- line: この値は1から段組内の全行数までの間の数値です。
- position: この値は1から行内の全文字数までの間の数値です。

選択された中にいくつかの文字が含まれている場合、最初の文字の位置が返されます。

WR SET CURSOR POSITIONでこの値を引数として指定すれば、後でこの位置に戻ることができます。

WR Get frameを使えば、どのエリアにカーソルがあるのかを確認することができます。

例題

テキスト中のカーソルの現在位置を変えずに、ドキュメントのヘッダにロゴを挿入したい場合は、挿入ボタンに下記のメソッドを埋め込みます:

```

C_LONGINT($frame;$Col;$Line;$Pos)
C_REAL($PictWidth;$PictHeight;$OrigWidth;$OrigHeight;$HeadTopMargin)
  `ドキュメントのどのフレームにカーソルがあるか?
$frame:=WR Get frame(Area)
  `現在のカーソルの位置を取得する
WR GET CURSOR POSITION(Area;$Page;$Col;$Line;$Pos)
  `カレントエリアをドキュメントのヘッダに切り替える
WR SET FRAME(Area;wr_right_header)
  `挿入したいロゴを含むレコードをロードする
ALL RECORDS ([Interface])
  `ロゴを挿入
WR INSERT PICTURE(Area;[Interface]Logo;0)
  `ロゴを選択し、サイズを取得する
WR SELECT(Area;4;1)
WR GET PICTURE SIZE(Area;$PictWidth;$PictHeight;$OrigWidth;$OrigHeight)
  `ヘッダの高さはピクチャに合わせる
$HeadTopMargin:=WR Get doc property(Area;wr_header_top_margin)
WR SET DOC PROPERTY(Area;wr_text_top_margin;$HeadTopMargin+$PictHeight)
WR SET DOC PROPERTY(Area;wr_header_bottom_margin;$PictHeight)
  `カーソルがあったフレームに戻る
WR SET FRAME(Area;$frame)
  `元々の位置にカーソルを配置する
WR SET CURSOR POSITION(Area;$Page;$Col;$Line;$Pos)

```

WR Get frame

WR Get frame (area) -> 戻り値

引数	型		説明
area	倍長整数	→	4D Writeエリア
戻り値	倍長整数	↩	カーソルのあるページエリア

説明

WR Get frameは、カーソルポイントまたは選択部分が属しているページの番号を返します。

以下の値が返されます:

定数	型	値
wr text frame	倍長整数	0
wr right header	倍長整数	1
wr right footer	倍長整数	2
wr left header	倍長整数	3
wr left footer	倍長整数	4
wr first header	倍長整数	5
wr first footer	倍長整数	6

これらの値を、番号または定数（上記参照）によって入力することができます。

例題

[WR GET CURSOR POSITION](#)と[WR SET CURSOR POSITION](#)コマンドの例題参照

WR SET AREA PROPERTY

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

引数	型		説明
area	倍長整数	⇒	4D Writeエリア
option	整数	⇒	オプション番号
value	整数	⇒	オプションに対応する値
stringValue	文字	⇒	オプションに対応するプロパティ文字列

説明

WR SET AREA PROPERTYを利用すれば、areaで指定された4D Writeエリアにoptionに対するvalueを変更することができます。

areaが0である場合、WR SET AREA PROPERTYは、コマンド呼び出し後に開かれる4D Writeエリアそれぞれに反映されます。この場合、**On Startupデータベースメソッド**でこのコマンドを呼び出すことを推奨します。

optionには“WR Area properties”テーマの定数を渡します。設定する値には“WR Parameters”テーマの定数を使用できます。定数とそれに対応する値の説明は以下の表の通りです。

stringValue引数はwr_window titleおよびwr_minimized_button_titleプロパティで使用できます。

定数	型	値	コメント
wr confirm dialog	倍長整数	0	確認ダイアログボックスの表示ステータス設定: <u>wr no dialog</u> (0), <u>wr display dialog</u> (1)
wr save preview	倍長整数	1	ピクチャプレビュー作成設定: <u>wr no picture preview</u> (0), <u>wr picture preview creation</u> (1)
wr allow undo	倍長整数	2	アクションのバッファを設定: <u>wr no undo</u> (0) = アクションを記憶しない, <u>wr undo allowed</u> (1) = アクションを記憶する
wr modified	倍長整数	3	更新ビットステータス設定 — <i>area</i> = 0の場合を除く: <u>wr dirty bit status false</u> (0), <u>wr dirty bit status true</u> (1)
wr fixed print size	倍長整数	4	可変サイズのプリントステータス設定 — <i>area</i> = 0の場合を除く: <u>wr var size printing status</u> (0), <u>wr fixed size printing status</u> (1)
wr convert dialog	倍長整数	5	4D Write 6.0フィールド変換ダイアログの表示ステータス設定 — <i>area</i> = 0の場合: <u>wr no dialog</u> (0), <u>wr display dialog</u> (1)
wr minimized button title	倍長整数	6	<i>area</i> が最小化されているときのボタンタイトル設定: <u>wr area name</u> (0), <u>wr custom title</u> (1) は <i>stringValue</i> に渡された文字列
wr window title	倍長整数	7	フルスクリーンやプラグインウィンドウでの4D Writeウィンドウのタイトル (0=エリア名, 1= <i>stringValue</i> に渡されたカスタムタイトル)
wr minimum width	倍長整数	8	ボタンにスイッチする前の <i>area</i> の最小幅設定 (ピクセル単位)
wr minimum height	倍長整数	9	ボタンにスイッチする前の <i>area</i> の最小高さ設定 (ピクセル単位)
wr save template on server	倍長整数	10	クライアント/サーバーモードでのテンプレート保存先設定: <u>wr on client</u> (0), <u>wr on server</u> (1)
wr load template on server	倍長整数	11	クライアント/サーバーモードでのテンプレートの検索先設定: <u>wr on client</u> (0), <u>wr on server</u> (1)
wr convert by token	倍長整数	12	ドキュメント変換時のフィールド参照の解釈設定: <u>wr convert by names</u> (0), <u>wr convert by numbers</u> (1)
wr zoom factor	倍長整数	13	<i>area</i> のズーム設定 (value=25%から500%の間)
wr allow drag	倍長整数	14	<i>area</i> からのドラッグ許可設定 (0=ドラッグを許可しない, 1=ドラッグを許可する)
wr allow drop	倍長整数	15	<i>area</i> へのドロップ許可設定 (0=ドロップを許可しない, 1=ドロップを許可する)
wr on the fly spellchecking	倍長整数	16	タイプ時にスペルチェックを行うかどうかの設定 (0=行う, 1=行わない)
wr timer frequency	倍長整数	17	<u>wr on timer</u> イベントを生成する間隔設定 (value=tick単位の呼び出し間隔 — 1 tick = 1/60秒 — デフォルトは3600 tick)
wr use saved zoom value	倍長整数	18	エリアが最後に閉じられたときに保存される、エリアを開くときのズーム値: <u>wr use default zoom</u> (0) = 100 %, <u>wr use saved zoom</u> (1)

例題 1

エリアの自動的なピクチャプレビュー、確認ダイアログの表示、編集メニューの取り消しコマンドを無効にします:

```
WR SET AREA PROPERTY(Area;wr save preview;wr no picture preview)
WR SET AREA PROPERTY(Area;wr confirm dialog;wr no dialog)
WR SET AREA PROPERTY(Area;wr allow undo;wr no undo)
```

例題 2

フィールド名でなく、テーブル/フィールド番号を使用して4D Writeバージョン6.xドキュメントを開きます。それによりv6ドキュメントが保存された後にフィールド名が変更されていても、ドキュメントを開く際にエラーは発生しません。このようにするには以下のコードを実行します:

```
WR SET AREA PROPERTY(0;wr convert by token;wr convert by numbers)
```


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

引数	型		説明
area	倍長整数	⇒	4D Writeエリア
page	倍長整数	⇒	ページ番号
column	倍長整数	⇒	段組数
line	倍長整数	⇒	行番号
position	倍長整数	⇒	行内のカーソルの水平位置

説明

*WR SET CURSOR POSITION*は *page*、*column*、*line*、*position*で指定された新しい位置にカーソルポイントを移動します。

- *page*: 指定する値は、ドキュメントの最初のページ番号から最後のページ番号までの間の番号を指定する必要があります。このページ番号は「環境設定」ダイアログで定義したページ番号を考慮する必要があります。
- *column*: 指定する値は、1から全段組数までの間の数値を指定する必要があります。
- *line*: 指定する値は、1から段組内の全行数までの間の数値（1つの段組しかない場合はページ）を指定する必要があります。
- *position*: 指定する値は1から1行の文字数までの間の数値を指定する必要があります。

本文のエリアと別の場所にカーソルを配置したい場合、*WR SET CURSOR POSITION*を使用する前に*WR SET FRAME*を使用する必要があります。

例題

4ページ目の10行目の行頭に挿入ポイントを移動したいものとします:

```
 `ドキュメントのbodyエリアにいることを確認する
If (WR Get frame (Area) #0)
  `違うエリアにいたらbodyエリアに移動する
    WR SET FRAME (Area;wr_text_frame)
End if
  `カーソルの移動
WR SET CURSOR POSITION (Area;10;1;10;1)
  `挿入ポイントを表示するためにエリアをスクロールする
WR SCROLL TO SELECTION (Area)
```

WR SET FRAME (area ; frame)

引数	型		説明
area	倍長整数	⇒	4D Writeエリア
frame	整数	⇒	フレーム番号

説明

*WR SET FRAME*は、*area*で指定された4D Writeエリアにおいて、*frame*によって指示された位置にカーソルポイントを移動します。この位置は、あらかじめ4D Writeに記憶されています。通常の表示モードが選択されていて挿入ポイントがヘッダエリアまたはフッタエリアのいずれかに配置される場合、4D Writeは自動でページプレビューモードに移行します。

*frame*には、次の値または定数を渡すことができます：

定数	型	値
wr text frame	倍長整数	0
wr right header	倍長整数	1
wr right footer	倍長整数	2
wr left header	倍長整数	3
wr left footer	倍長整数	4
wr first header	倍長整数	5
wr first footer	倍長整数	6

左ページと右ページに異なったヘッダやフッタを使用する際には、3と4を使用します。

最初のページに異なったヘッダやフッタを使用する際には、5と6を使用します。

Note: 値の一覧は""定数テーマでもご覧いただけます。

例題

以下のコマンドの例題を参照してください: [WR GET CURSOR POSITION](#), [WR SET CURSOR POSITION](#), [WR INSERT PAGE NUMBER](#)

WR TEXT ACCESS (area ; mode)

引数	型		説明
area	倍長整数	→	4D Write area
mode	整数	→	0=Allow access 1=Restrict access

説明

WR TEXT ACCESSを使用して、エリア内のテキストへのアクセスを制御できます。エリアが読み込みのみモードで開かれているとき、メニューやルーラー、ズームボックスは表示されません。テキストは表示されスクロールできますが、更新できません。

modeには"WR Parameters"テーマの以下の定数を渡すことができます:

定数	型	値	コメント
wr allowed access	倍長整数	0	エリアにフルアクセスがあります。
wr restricted access	倍長整数	1	ユーザはエリア情報に読み込みのみでアクセスできます。

すでにアクセスが制限されているエリアへのアクセスを、modeにwr allowed accessを渡すことで変更する場合、WR SET DOC PROPERTY (Area;wr view menubar;wr displayed) とWR SET DOC PROPERTY (Area;wr view rulers;wr displayed) を呼び出してルーラーとメニューバーを表示しなければなりません。

ドラッグアンドドロップについて

このコマンドはキーボードを使用したデータ入力やコピー/ペーストを制御しますが、エリアへのドロップやエリアからのドラッグは制御しません。この動作は特定のインターフェースで便利です。しかしエリアへのすべての更新を拒否したい場合は、以下のコードを使用します:

```
WR TEXT ACCESS(TheArea;wr_restricted_access)
WR SET AREA PROPERTY(TheArea;wr_allow_drag;wr_drag_not_allowed)
WR SET AREA PROPERTY(TheArea;wr_allow_drop;wr_drop_not_allowed)
```

例題

以下の例題は4D Writeエリアを含むフォームのフォームメソッドで、フォームをロードする際に読み込みのみに設定します。

```
If (Form event=On_Load)
    WR TEXT ACCESS(area;wr_restricted_access)
End if
```

WRエリアコントロール

 エリアコントロールコマンドについて

-  WR EXECUTE COMMAND
-  WR GET COMMAND INFO
-  WR Get doc property
-  WR Get on command method
-  WR LOCK COMMAND
-  WR ON COMMAND
-  WR REDRAW
-  WR SCROLL TO SELECTION
-  WR SET DOC PROPERTY
-  WR UPDATE MODE

✦ エリアコントロールコマンドについて

"WRエリアコントロール"テーマのコマンドや関数を使用して、4D Writeエリアの表示や操作をコントロールできます。

WR SCROLL TO SELECTION, *WR UPDATE MODE*そして*WR REDRAW*コマンドを使用してスクリーンの更新をコントロールできます。

*WR ON COMMAND*と*WR Get on command method*コマンドを使用してエリアのメニュー項目の振る舞いをコントロールできます。

メニューステータスの情報や (*WR GET COMMAND INFO*)、メニュー項目の有効・ロック状態 (*WR EXECUTE COMMAND*, *WR LOCK COMMAND*) を取得できます。

また*WR SET DOC PROPERTY*や*WR Get doc property*コマンドは4D Writeエリアのインターフェースオブジェクトに関する情報とコントロールオプションを提供します。

参照の更新

4Dフィールドが割り当てられた、フォーム上に置かれた4D Writeエリアは、*On Load*などのフォームのコントロールイベントの前にロードされます。エリアに参照が含まれる場合、*WR REDRAW*コマンドを使用して再計算を行う必要があります。

WR EXECUTE COMMAND (area ; cmdNumber)

引数	型		説明
area	倍長整数	→	4D Writeエリア
cmdNumber	倍長整数	→	実行するコマンド番号

説明

*WR EXECUTE COMMAND*は、4D Writeのメニューやツールバーに割り当てられている処理を実行します。このコマンドの最も一般的な使い方としては、ユーザがメニューなどから4D Writeコマンドをコールした際に、*WR ON COMMAND*により割り込みをかけ、メソッドを実行した後に、本来ユーザが選択した4D Writeコマンドを実行することがあげられます。

Note: このコマンドリストとその説明については、[WR Commands](#)を参照してください。この引数には、定数と値のどちらでも渡すことができます。

例題

ボタンを使用してワープロの機能にアクセスします:

bNewボタンのオブジェクトメソッド:

```
WR EXECUTE COMMAND(theArea;wr_cmd_new)
`新規コマンドを実行
```

bOpenボタンのオブジェクトメソッド:

```
WR EXECUTE COMMAND(theArea;wr_cmd_open)
`開くコマンドを実行
```

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

引数	型	説明
area	倍長整数	⇒ 4D Writeエリア
commandNumber	倍長整数	⇒ 処理するコマンドの番号
applied	倍長整数	← 0=適用されない, 1=適用される, 2=部分的に適用される
stringValue	文字	← 選択されたテキスト値
name	文字	← コマンド名またはTipのテキスト
status	整数	← 0=無効 1=有効

説明

WR GET COMMAND INFOを使用して、*commandNumber*で指定された番号のメニューやツールバーのコマンドステータスを取得できます。

Note: コマンド一覧およびリファレンスはWR Commandsを参照してください。この引数には、定数と値のどちらでも渡すことができます。それぞれのコマンドについては付録B: メニュー項目番号参照してください。

*applied*はコマンドが現在選択されているテキストに適用されるかされないか、あるいは部分的に適用されるかどうかを返します。*applied*の値は、コマンドが使用不可の場合0、使用可能である場合は1、部分的に使用可能である場合には2となります。例えば**太字**コマンド (定数: *wr cmd bold* , 値: 502) を使用するとします。以下のコードが実行される時:

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

applied=0: 現在選択されているテキストが太字のとき

applied=1: 現在選択されているテキストが太字でないとき

applied=2: 現在選択されているテキストの一部が太字のとき

*stringValue*には各コマンドで変更または指定されるテキストが返されます。例えば**フォント**ドロップダウンリスト (定数: *wr cmd font dropdown*, 値: 1002) に関連して以下のコードを実行すると:

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

現在選択されているフォントがArialであれば、*stringValue*="Arial"が返されます。

*name*には、コマンド名が返されます。これは、メニュー項目のテキスト、またはそのコマンド用に表示されたTipsのテキストです。

*status*には、メニュー項目のステータスが返されます。メニュー項目が使用不可であれば0、使用可能な場合には1の値が<ステータス>に返されます。

例題

不可視文字を隠したり表示したりするボタンがフォームに配置されています。ボタンのタイトルは現在の設定によって変更されなければなりません:

```
WR GET COMMAND INFO(area;wr_cmd_view_invisibles;vApplied;vStringValue;vName;vStatus)
Case of
: (vApplied=1)
    BUTTON TEXT (bStatus; "不可視文字を隠す")
: (vApplied=0)
    BUTTON TEXT (bStatus; "不可視文字を表示")
End case
```

WR Get doc property

WR Get doc property (area ; property) -> 戻り値

引数	型		説明
area	倍長整数	→	4D Writeエリア
property	整数	→	読み取るプロパティの番号
戻り値	実数	↩	テストしたプロパティの値

説明

WR Get doc propertyを使用して、areaで指定された4D Writeエリア内で開かれているドキュメントの属性を取得できます。

property は“WR Document properties”テーマの定数を使用して設定できます。定数名あるいは値を使用できます。

いくつかのプロパティについて、WR Get doc propertyは1（真）または0（偽）を返します。例えばプロパティ2（wr view ruler）などです。

その他のプロパティについて、WR Get doc propertyは現在のデフォルト単位で表現されている番号を返します。例えばプロパティ37（wr paper width）などです。

“WR Document properties”テーマの定数に関する情報はWR SET DOC PROPERTYコマンドの説明を参照してください。

例題

WR SET DOC PROPERTY, WR INSERT PAGE NUMBER, WR GET CURSOR POSITION そして WR SET PICTURE IN PAGE INFO コマンドの例題を参照

WR Get on command method

WR Get on command method (area) -> 戻り値

引数	型		説明
area	倍長整数	→	4D Write area
戻り値	文字	↩	Name of installed on command method

説明

WR Get on command method コマンドは、*WR ON COMMAND*を使用して4D Write *area*にインストールされたメソッドの名前を返します。

メソッドがインストールされていない場合、空の文字列 ("") が返されます。

WR LOCK COMMAND (area ; cmdNumber ; locked)

引数	型		説明
area	倍長整数	→	4D Writeエリア
cmdNumber	倍長整数	→	処理するコマンドの番号
locked	整数	→	0=実行可能にする、1=実行できないようにする

説明

WR LOCK COMMANDを使用することで、cmdNumberに渡された番号で指定したコマンドを実行不能にできます。メニュー項目やパレット項目のコマンドが関連します。このコマンドはareaで参照された4D Writeエリア内の、指定されたコマンドへのアクセスに影響します。他の4D Writeエリア内のコマンドへのアクセスには影響を及ぼしません。

lockedには"WR Parameters"テーマの以下の定数を渡すことができます:

定数	型	値	コメント
wr enabled command	倍長整数	0	コマンドは呼び出されたときに実行されます。
wr locked command	倍長整数	1	コマンドは呼び出し時に実行されず、メニューは使用不可になります。

Notes:

- コマンドがロックされていても、WR EXECUTE COMMANDを利用すればコードを使用してコマンドを実行できます。
- 使用不可にされているコマンドを選択した場合、WR ON COMMANDは呼び出されません。
- cmdNumberにメニューやサブメニューを渡すと、そのメニューとメニュー内の全コマンドが使用不可（グレー表示される）になります。

実行不能にしたメニューのコマンドを選択することはできませんが、キーボードショートカットあるいはツールバーのボタンは使用できます。これらのコマンドを完全にロックしたい場合は、各メニュー項目に対しWR LOCK COMMANDを呼び出さなければなりません。

Note: メニュー、コマンド、そしてその参照は"WR Commands"定数テーマにあります。定数名あるいは値を渡すことができます。

例題 1

デザイナーにのみ、デザインモードへのアクセスを許可します:

```
If (Current user="Designer")
    WR LOCK COMMAND(Area;wr_cmd_insert_4D_expression;wr_enabled_command)
Else
    WR LOCK COMMAND(Area;wr_cmd_insert_4D_expression;wr_locked_command)
End if
```

例題 2

ユーザ名が"Guru"でなければ、新しいドキュメントを作成できません:

```
If (Form event=On_Load)
    If (Current user#"Guru")
        WR LOCK COMMAND(Area;wr_cmd_new;wr_locked_command)
    End if
End if
```

WR ON COMMAND (area ; 4DRepMethod)

引数	型	説明
area	倍長整数	→ 4D Writeエリア
4DRepMethod	文字	→ 代わりに実行するメソッド

説明

WR ON COMMANDは、メニューからコマンドを選択する、またはボタンをクリックすることにより、ユーザによって4D Writeコマンドが呼び出された際に、4DRepMethodで指定されたメソッドを実行します。areaが0の場合、4DRepMethodで指定されたメソッドはデータベースが閉じられるか、次のようにWR ON COMMANDが呼び出されるまで、各4D Writeエリアに適用されます

```
WR ON COMMAND(0; "")
```

4DRepMethodは2つの引数を受け取ります:

- \$1にはareaを表す倍長整数値
- \$2にはコマンド番号を表す倍長整数値

データベースをコンパイルする際には、\$1と\$2を倍長整数値として宣言しなければなりません。

選択されたコマンドを4DRepMethod内で実行するには、以下のコードを使用します:

```
WR EXECUTE COMMAND($1;$2)
```

例題

“Archive”フォルダにドキュメントを保存します:

```
C_LONGINT($1;$2)
Case of
: ($2=wr_cmd_save_as) `別名で保存が選択されたら...
  $DocName:=Request("ドキュメント名: ")
  If((OK=1) & ($DocName#""))
  `指定したフォルダにドキュメントを保存
    WR SAVE DOCUMENT($1;"HDisk:Archives:"+$DocName) `Mac
    WR SAVE DOCUMENT($1;"D:\Archives\"+$DocName) `Win
  Else
    BEEP `エラー
  End if
Else `他のメニューコマンド
  WR EXECUTE COMMAND($1;$2)
`通常のアクションを実行
End case
```

フォームメソッド:

```
If(Form event=On_Load)
  WR ON COMMAND(Area;"TheMethod")
End if
```

WR REDRAW (area)

引数	型		説明
area	倍長整数	⇒	4D Writeエリア

説明

*WR REDRAW*によって、*area*が再描画されます。このコマンドは、*WR UPDATE MODE*によってスクリーンの更新が行われないように設定されている場合に、実行されたコードによってエリアがどのように変更されたかを確認するために、4D Write エリアを再描画したい場合に便利です。

例題

次の例は、*area*で指定されたエリアのスクリーン更新をオフにして、エリアを再フォーマットする***Reformat***というプロジェクトメソッドを呼び出し、スクリーンの更新をオンに戻さないで*area*を再描画させます。

```
WR UPDATE MODE(area;0)
  `スクリーンの更新をオフにする
Reformat(area)
  `areaを引数としてメソッドに渡すことができる
WR REDRAW(area)
  `フォーマットを反映させるために再描画する
```

WR SCROLL TO SELECTION

WR SCROLL TO SELECTION (area)

引数	型	説明
area	倍長整数	4D Writeエリア

説明

*WR SCROLL TO SELECTION*は、選択されたテキストが表示されるまで*area*をスクロールします。このコマンドは、4D Writeコマンドを利用して修正を行い、その変更結果を見る必要がある場合に有効です。

Note: *WR UPDATE MODE*コマンドを使用して、事前にスクリーン更新を不可にしている場合、*WR SCROLL TO SELECTION*コマンドは無効になります。

例題

*WR Get font*と*WR SET CURSOR POSITION*コマンドの例題参照

WR SET DOC PROPERTY

WR SET DOC PROPERTY (area ; property ; value)

引数	型		説明
area	倍長整数	→	4D Writeエリア
property	整数	→	設定するプロパティの番号
value	倍長整数	→	プロパティの新しい値

説明

*WR SET DOC PROPERTY*を使用して、*area*で指定された4D Writeエリア内で開かれているドキュメントの属性を変更することができます。

*value*に与えられる意味は、使用される*property*の値によって変わります。*value*と*property*は定数を使用して設定されま

す。
"WR Document properties"テーマの定数は以下の表の通りです。

以下の定数は*WR SET DOC PROPERTY*と*WR Get doc property*コマンドで使用できます。また値を設定するために"**WR Parameters**"テーマの定数を使用できます：

定数	型	値	コメント
wr first page	倍長整数	0	最初のページ番号設定 (デフォルトで1)。例えば10に設定すると次のページは11になる。
wr view mode	倍長整数	1	ドキュメント表示モードの設定: <u>wr page mode</u> (0) または <u>wr normal mode</u> (1)
wr view rulers	倍長整数	2	ルーラの表示ステータス設定: <u>wr hidden</u> (0) または <u>wr displayed</u> (1)
wr view frames	倍長整数	3	テキストフレームの表示ステータス: <u>wr hidden</u> (0) または <u>wr displayed</u> (1)
wr view headers	倍長整数	4	ヘッダの表示ステータス: <u>wr hidden</u> (0) or <u>wr displayed</u> (1), 先頭ページが他のページと異なる場合は' <u>wr view first page header</u> 'を使用。
wr view footers	倍長整数	5	フッタの表示ステータス設定: <u>wr hidden</u> (0) または <u>wr displayed</u> (1), 先頭ページが他のページと異なる場合' <u>wr view first page footer</u> 'を使用。
wr view pictures	倍長整数	6	ピクチャの表示ステータス設定: <u>wr hidden</u> (0) または <u>wr displayed</u> (1)
wr view Hscrollbar	倍長整数	7	横スクロールバーの表示ステータス設定: <u>wr hidden</u> (0) または <u>wr displayed</u> (1)
wr view Vscrollbar	倍長整数	8	縦スクロールバーの表示ステータス設定: <u>wr hidden</u> (0) または <u>wr displayed</u> (1)
wr view statusbar	倍長整数	9	ステータスバーの表示ステータス設定: <u>wr hidden</u> (0) または <u>wr displayed</u> (1)
wr view menubar	倍長整数	10	メニューバーの表示ステータス設定: <u>wr hidden</u> (0) または <u>wr displayed</u> (1)
wr view standard palette	倍長整数	11	標準ツールパレットの表示ステータス設定: <u>wr hidden</u> (0) または <u>wr displayed</u> (1)
wr view format palette	倍長整数	12	フォーマットツールバーの表示ステータス: <u>wr hidden</u> (0) または <u>wr displayed</u> (1)
wr view style palette	倍長整数	13	スタイルツールバーの表示ステータス設定: <u>wr hidden</u> (0) または <u>wr displayed</u> (1)
wr view	倍		

wr view borders palette	長 整 数	14	罫線ツールバーの表示ステータス設定: <u>wr hidden</u> (0) or <u>wr displayed</u> (1)
wr view invisible chars	倍 長 整 数	15	非表示文字の表示ステータス設定: <u>wr hidden</u> (0) または <u>wr displayed</u> (1)
wr view references	倍 長 整 数	16	参照の表示ステータス設定: <u>wr hidden</u> (0) または <u>wr displayed</u> (1)
wr view column separators	倍 長 整 数	17	複数段組みモードに置いて、段組み間の縦分割線の表示を設定 - 段組みダイアログボックスの縦分割オプションに対応: <u>wr hidden</u> (表示しない) (0) または <u>wr displayed</u> (表示する) (1)
wr different on first page	倍 長 整 数	18	最初のページのヘッダ/フッタが異なるかを指定 - 環境設定ダイアログボックスの'左と右ページで異なる'オプションに対応: <u>wr similar</u> (0) または <u>wr different</u> (1)
wr different left right pages	倍 長 整 数	19	左と右のページでヘッダ/フッタが異なるかを指定 - 環境設定ダイアログボックスの'左と右ページで異なる'オプションに対応: <u>wr similar</u> (0) または <u>wr different</u> (1)
wr widow orphan	倍 長 整 数	20	Gets or sets whether widows and orphans are taken into account - corresponds to the 'Widow and Orphan Control' option in the Preferences dialog box: <u>wr ignored</u> (0) or <u>wr managed</u> (1)
wr unit	倍 長 整 数	21	ドキュメントの現在の単位を設定 - 環境設定ダイアログボックスの'単位'ポップアップメニューに対応: <u>wr centimeters</u> (0), <u>wr inches</u> (1) または <u>wr pixels</u> (2)
wr default tab	倍 長 整 数	22	現在のドキュメント単位で表されたデフォルトの"自動"タブスペース - 環境設定の'デフォルトタブスペース'エリアに対応 (デフォルトで0.5インチ、1.3cm、36ピルセル)
wr language	倍 長 整 数	23	ドキュメントに割り当てる言語設定 (American English = 1033, Australian English = 3081, English = 2057, Catalan = 1027, Danish = 1030, Dutch = 1043, Finnish = 1035, French = 1036, French Canadian = 3084, German = 1031, Italian = 1040, Norwegian Bokmal = 1044, Norwegian Nynorsk = 2068, Portuguese Brazil = 1046, Portuguese Iberian = 2070, Spanish = 1034, Swedish = 1053, Russian = 1049, Czech = 1029, Hungarian = 1038, Polish = 1045)
wr number of columns	倍 長 整 数	24	ドキュメントの段組み数設定 Gets or sets the number of columns of the document
wr column spacing	倍 長 整 数	25	現在のドキュメント単位で表した、段落間のスペース、段落ダイアログボックスの"スペース"に対応。
wr binding	倍 長 整 数	26	カレントのドキュメント単位で表したバインドサイズの設定、環境設定ダイアログボックスの"バインド"エリアに対応。
wr opposite pages	倍 長 整 数	27	ドキュメントの両面モード設定 - 環境設定ダイアログボックスの'両面ページ'オプションに対応: <u>wr single sided pages</u> (0) または <u>wr double sided pages</u> (1)

wr right first page	長 整 数	28	先頭ページが右か左かを設定 - デフォルトで右ページ: <code>wr left page (0)</code> または <code>wr right page (1)</code>
wr text inside margin	倍 長 整 数	29	テキストの左側と、右ページの場合用紙の左側間、左ページの場合右側間のマージンを、現在のドキュメントの単位で設定 (ページモードで使用)。
wr text left margin	倍 長 整 数	29	ページの左側と用紙の左側間のマージンを、現在のドキュメントの単位で設定 (通常モードで使用)。
wr text outside margin	倍 長 整 数	30	テキストの右側と、右ページの場合用紙の右側間、左ページの場合左側間のマージンを、現在のドキュメントの単位で設定 (ページモードで使用)。
wr text right margin	倍 長 整 数	30	ページの右側用紙の右側間のマージンを、現在のドキュメントの単位で設定 (通常モードで使用)。

環境設定ダイアログボックスで"先頭ページと異なる"オプションが選択されている場合、先頭ページを除き以下の定数をすべてのページで使用すべきです:

定数	型	値	コメント
wr text top margin	倍長整数	31	
wr text bottom margin	倍長整数	32	ページボディ下端と用紙下端のマーヅンを現在のドキュメントの単位で設定、最初のページだけに別の設定を行うには' <u>wrfirst page bottom margin</u> 'を使用。
wr header top margin	倍長整数	33	ページヘッダの上端と用紙上端間のマーヅンを現在の用紙の単位で設定、先頭ページが他のページと異なる場合' <u>wr header 1st page top margin</u> 'を使用。
wr header bottom margin	倍長整数	34	ページヘッダの下端と用紙上端間のマーヅンを現在の用紙の単位で設定、先頭ページが他のページと異なる場合' <u>wr header 1st page bottom mg</u> 'を使用。
wr footer top margin	倍長整数	35	ページフッタの上端と用紙下端間のマーヅンを現在の用紙の単位で設定、最初のページが他と異なる場合は' <u>wr footer 1st page top margin</u> 'を使用。
wr footer bottom margin	倍長整数	36	ページフッタの下端と用紙下端のマーヅンを現在のドキュメントの単位で設定、先頭ページ型のページと異なる場合' <u>wr footer 1st page bottom mg</u> 'を使用。
wr paper width	倍長整数	37	用紙の幅を現在のドキュメントの単位で設定 (*)
wr paper height	倍長整数	38	用紙の高さを現在のドキュメントの単位で設定 (*)
wr dead left margin	倍長整数	39	用紙左側の、プリンターによって予約されている非印刷領域のサイズを現在のドキュメント単位で取得 (この値は読み込みのみ可能です) (*)
wr dead top margin	倍長整数	40	用紙上側の、プリンターによって予約されている非印刷領域のサイズを現在のドキュメント単位で取得 (この値は読み込みのみ可能です) (*)
wr printable width	倍長整数	41	左上マーヅンから開始して、横方向の印刷可能領域を取得 (この値は読み込みのみ可能)。The right dead margin equals the paper width; the left dead margin-the printable width.
wr printable height	倍長整数	42	左上マーヅンから開始して、縦方向の印刷可能領域を取得 (この値は読み込みのみ可能)。The bottom dead margin equals the paper height; the top dead margin-the printable height.
wr data size	倍長整数	43	バイト単位でドキュメントのサイズを取得 (この値は読み込みのみ可能です)。
wr undo buffer size	倍長整数	44	取り消しバッファをバイト単位で設定 (この値は読み込みのみ可能)。
wr	倍		

horizontal splitter	長 整 数	45	水平スプリッターの表示ステータス: <u>wr hidden</u> (0) または <u>wr displayed</u> (1)
wr vertical splitter	倍 長 整 数	46	縦スプリッターの表示設定: <u>wr hidden</u> (0) または <u>wr displayed</u> (1)
wr links color	倍 長 整 数	47	未訪問のハイパーリンクのカラー設定
wr visited links color	倍 長 整 数	48	訪問済みハイパーリンクのカラー設定
wr view frame area	倍 長 整 数	49	フォーム中の領域の外周に表示されるフレームの設定: <u>wr hidden</u> (フレームなし) (0) または <u>wr displayed</u> (フレームあり)(1)

(*) プログラムで用紙サイズを設定している場合、4D Writeは"仮想"プリンタデバイスが使用されているものと見なします。プログラムはデッドマージンを0に設定し、印刷可能エリアは用紙サイズと等しくなります。この動作は印刷用でないドキュメントで便利です。

環境設定ダイアログボックスで"先頭ページと異なる"オプションが選択されている場合、以下の定数 (50 - 59) はドキュメントの先頭ページ用に使用されます。

定数	型	値	コメント
wr view first page header	倍長 整数	50	先頭ページのヘッダの表示ステータス設定: <u>wr hidden</u> (0) または <u>wr displayed</u> (1), 他のページには' <u>wr view headers</u> 'を使用。
wr view first page footer	倍長 整数	51	先頭ページのフッタの表示ステータス設定: <u>wr hidden</u> (0) または <u>wr displayed</u> (1), 他のページには' <u>wr view footers</u> 'を使用。
wr first page top margin	倍長 整数	52	最初のページのボディ上端と用紙上端のマージンを現在のドキュメントの単位で設定、他のページには' <u>wr text top margin</u> 'を使用。
wr first page bottom margin	倍長 整数	53	最初のページのボディと下端のマージンを現在のドキュメントの単位で設定、他のページには' <u>wr text bottom margin</u> 'を使用。
wr header 1st page top margin	倍長 整数	54	先頭ページのヘッダの上端とページの上端間のマージンを現在のドキュメントの単位で設定、他のページには' <u>wr header top margin</u> 'を使用。
wr header 1st page bottom mg	倍長 整数	55	先頭ページヘッダの下端と用紙上端間のマージンを現在の用紙の単位で設定、他のページには' <u>wr header bottom margin</u> 'を使用。
wr footer 1st page top margin	倍長 整数	56	最初のページのフッタと用紙下端のマージンを現在のドキュメントの単位で設定、他のページには' <u>wr footer top margin</u> 'を使用。
wr footer 1st page bottom mg	倍長 整数	57	最初のページの下端と用紙下端とのマージンを現在の用紙の単位で設定、他のページには' <u>wr footer bottom margin</u> 'を使用
wr draft mode	倍長 整数	58	ドキュメントのテキスト入力モード設定: <u>wr wysiwyg</u> (0) また <u>wr draft</u> (1)
wr column width	倍長 整数	59	現在のドキュメント単位で表した段落幅 (この値は読み込みのみ可能です)。

例題 1

4D Writeエリアをメニューとルーラーなしで表示します:

```
If (Form event=On Load)
  WR SET DOC PROPERTY(Area;wr_view_menubar;wr_hidden)
  WR SET DOC PROPERTY(Area;wr_view_rulers;wr_hidden)
End if
```

例題 2

このメソッドはスクロールバーを表示あるいは非表示にします:

```
C_LONGINT(ScrollStatus)
ScrollStatus:=WR Get doc property(Area;wr Hscrollbar) `定数=7
ScrollStatus:=ScrollStatus+WR Get doc property(Area;wr Vscrollbar) `定数=8
If(ScrollStatus>0)
  CONFIRM("スクロールバーが表示されています。隠しますか?")
  If(OK=1)
    WR SET DOC PROPERTY(Area;wr Hscrollbar;wr_hidden)
    WR SET DOC PROPERTY(Area;wr Vscrollbar;wr_hidden)
  End if
Else
  CONFIRM("スクロールバーが隠されています。表示しますか?")
  If(OK=1)
    WR SET DOC PROPERTY(Area;wr Hscrollbar;wr_displayed)
    WR SET DOC PROPERTY(Area;wr Vscrollbar;wr_displayed)
  End if
End if
```

WR UPDATE MODE (area ; mode)

引数	型		説明
area	倍長整数	→	4D Writeエリア
mode	整数	→	0=更新しない、1=更新する

説明

WR UPDATE MODEを使用して、areaで指定されたエリアにおけるスクリーンの更新を可または不可に設定することができます。このコマンドは4D Writeコマンドによるスクリーンの更新に対してのみ有効です。areaに対するユーザアクションはスクリーンに反映されます。

mode引数には"WR Parameters"テーマの以下の定数を渡します:

定数	型	値	コメント
wr screen updating off	倍長整数	0	スクリーンの更新を無効にする。
wr screen updating on	倍長整数	1	スクリーンの更新を有効にする。

WR UPDATE MODEのmode引数にwr screen updating on定数を渡して呼び出すと、エリアは再描画されるので、WR REDRAWコマンドを呼び出す必要はありません。

スクリーンの更新が無効になっていると、4D Writeコマンドの実行が速くなります。例えば4D Writeエリアに一連の更新を実行する場合、その前にスクリーンの更新を無効にし、処理が終了したら更新を有効に戻します。コマンドの実行は速くなり、スクリーンも再描画されます。

例題

次の例は、スクリーンの更新をオフにした後にいくつかの修正処理を行うプロジェクトメソッドReformatを呼び出し、スクリーンの更新をオンに戻します:

```
WR UPDATE MODE(area;wr screen updating off)
Reformat(Area)
WR UPDATE MODE(area;wr screen updating on)
```

WRスタイルシート

 スタイルシートコマンドについて

 WR ADD STYLESHEET TAB

 WR APPLY STYLESHEET

 WR Create stylesheet

 WR DELETE STYLESHEET

 WR DELETE STYLESHEET TAB

 WR Get stylesheet font

 WR GET STYLESHEET INFO

 WR GET STYLESHEET TAB

 WR Get stylesheet text prop

 WR SET STYLESHEET FONT

 WR SET STYLESHEET INFO

 WR SET STYLESHEET TAB

 WR SET STYLESHEET TEXT PROP

 WR UPDATE STYLESHEET

✦ スタイルシートコマンドについて

このテーマのコマンドを関数を使用して、選択したテキストに適用するスタイルシートをコントロールできます。

現在のスタイルシートを取得したり、異なるものを適用したりできます。この機能により太字やイタリックなどの適用、フォントサイズの変更などを行えます。

また既存のスタイルシートを削除することもできます。

WR ADD STYLESHEET TAB (area ; stylesheetNumber ; location ; justification ; fillCharacter)

引数	型		説明
area	倍長整数	→	4D Writeエリア
stylesheetNumber	倍長整数	→	スタイルシート番号
location	倍長整数	→	タブ位置
justification	整数	→	タブの文字揃え位置
fillCharacter	文字	→	選択された埋め込み文字

説明

WR ADD STYLESHEET TABを使用すれば、*styleSheetNumber*が参照するタブストップのリストに新規のタブストップを追加することができます。またWR ADD STYLESHEET TABは、タブ位置やそのタイプ、埋め込み文字を設定することもできます。

*position*によって指定された位置にすでにタブストップがある場合、定義したタブストップに置き換えられます。

Note: WR UPDATE STYLESHEETを呼び出して、スタイルシートを使用しているテキストを更新しない限り、変更したいスタイルシートを使用しているテキストは更新されません。

*position*は（ドキュメントのデフォルト単位で表されている）左余白からの距離です。

オプションの*justification*引数は、タブストップのタイプを指定します。“WR Tabs”テーマの定数を使用できます：

定数	型	値	コメント
wr left tab	倍長整数	1	左揃え
wr centered tab	倍長整数	2	中央合わせ
wr right tab	倍長整数	3	右揃え
wr decimal tab	倍長整数	4	小数点
wr vertical separator tab	倍長整数	5	縦分割

*justification*が省略されている場合は、左揃えのタブが作成されます。

オプションの*fillCharacter*引数は、ASCIIコードの33から127までの間の文字を指定することができます。この文字はタブストップと同じフォントを使用して追加されます。*fillCharacter*が省略されている場合や空の文字例が指定されている場合、埋め込み文字は挿入されません。

例題

WR UPDATE STYLESHEETコマンドの例題参照

WR APPLY STYLESHEET

WR APPLY STYLESHEET (area ; stylesheetNumber)

引数	型		説明
area	倍長整数	→	4D Write area
stylesheetNumber	倍長整数	→	Stylesheet number

説明

*WR APPLY STYLESHEET*は、*area*の現在選択されている部分に *styleSheetNumber*で指定されたスタイルシートを適用します。スタイルシートの書式は選択部分に適用され、選択部分はスタイルシートを使用して表示されます（カーソルがテキスト内にある場合、スタイルツールバーのスタイルシートドロップダウンリストにスタイルが表示されます）。

*styleSheetNumber*がスタイルシートに対応していない場合、エラー1078 (存在しないスタイルシート) が返されます。

例題

*WR Create stylesheet*コマンドの例題参照

WR Create stylesheet (area ; name ; applyTo ; shortcut) -> 戻り値

引数	型		説明
area	倍長整数	→	4D Writeエリア
name	文字	→	スタイルシート名
applyTo	倍長整数	→	0=文字、1=段落
shortcut	文字	→	1つの文字
戻り値	倍長整数	↻	スタイルシート参照番号

説明

WR Create stylesheetは、新規スタイルシートを作成し、そのスタイルシートに割り当てられた番号を返します。この新規スタイルシートの内容はname、applyTo、shortCutで設定されます。WR SET STYLESHEET TEXT PROP、WR SET STYLESHEET FONT、WR SET STYLESHEET TAB及びスタイルシート参照番号を使用して、スタイルシートを修正することができます。

name: スタイルシートの名前は、最大31バイトです。

applyTo引数にはWR Parameters引数の以下の定数を渡すことができます:

定数	型	値	コメント
wr apply to characters	倍長整数	0	文字にスタイルシートを適用する。
wr apply to paragraphs	倍長整数	1	段落にスタイルシートを適用する。

段落に適用される場合、選択したテキストに含まれる最初の段落から始まり、選択された最後の段落の終わりまで適用されます。applyToが省略されると、スタイルシートは文字に対して適用されます

オプションのshortCut引数を使用して、スタイルシートにキーボードショートカットを割り当てることができます。ショートカットを使用する際は**Ctrl** (Windows) あるいは**Command** (Mac OS) とともに使用します。標準の4D Writeキーボードショートカットとの衝突をさせるために、番号を使用することをお勧めします。

shortCutを省略するか空の文字列を指定すると、スタイルシートにショートカットキーは割り当てられません。

例題

文字のスタイルシートをカスタマイズして各ドキュメントに追加し、選択範囲に適用したいものとします。スタイルシートは、Mac OS上では「**Command+1**」キー、Windows上では「**Ctrl+1**」キーのショートカットキーを適用します。フォント属性はComic Sans MS、12ポイントにします。

```
$NumSheet:=WR Create stylesheet(Area;"MyOwnStyle";wr apply to characters;"1")
WR SET STYLESHEET FONT(Area;$NumSheet;"Comic Sans MS")
WR SET STYLESHEET TEXT PROP(Area;$NumSheet;wr font size;12;1)
WR EXECUTE COMMAND(Area;wr cmd select_all)
WR APPLY STYLESHEET(Area;$NumSheet)
```

WR DELETE STYLESHEET

WR DELETE STYLESHEET (area ; stylesheetNum)

引数	型		説明
area	倍長整数	→	4D Writeエリア
stylesheetNum	倍長整数	→	スタイルシート番号

説明

*WR DELETE STYLESHEET*は、*area*で参照される4D Writeエリアから*styleSheetNum*で渡された番号のスタイルシートを削除します。

警告: システムのスタイルシートは削除できません。*WR GET STYLESHEET INFO*を使うことにより、そのスタイルシートが削除から保護されているかどうかを確認することができます。

例題

ドキュメント内の保護されていないスタイルシートを削除します:

```
C_LONGINT (Area)
C_LONGINT (NbStyleSheet; $SheetNum)
//スタイルシートの数を取得
NbStyleSheet:=WR_Count(Area;wr_nb_stylesheets)
$SheetNum:=1
For ($i;1;NbStyleSheet)
  WR_GET_STYLESHEET_INFO(Area;$SheetNum;$Name;$ApplyTo;$Protected;$Shortcut)
  If ($Protected=0) //スタイルシートが保護されていなければ...
    WR_DELETE_STYLESHEET(Area;$SheetNum)
  Else
    $SheetNum:=$SheetNum+1
  End if
End for
```

WR DELETE STYLESHEET TAB

WR DELETE STYLESHEET TAB (area ; stylesheetNumber ; tabNumber)

引数	型		説明
area	倍長整数	⇒	4D Writeエリア
stylesheetNumber	倍長整数	⇒	スタイルシート番号
tabNumber	倍長整数	⇒	削除するタブの番号

説明

*WR DELETE STYLESHEET TAB*は、*area*で参照される4D Writeエリア内にある*styleSheetNumber*を持つスタイルシートから*tabNumber*で渡された番号のタブストップを削除します。スタイルシートダイアログに一覧表示される通りに、スタイルシートは先頭から末端に向かって順に番号が振られます。このコマンドは*styleSheetNumber*のスタイルシートが使用されている場合でも、現在選択されているテキストに影響を与えません。

修正されたスタイルシートを使用するテキストを更新するには、*WR UPDATE STYLESHEET*を使用する必要があります。

例題

[WR UPDATE STYLESHEET](#)コマンドの例題参照

WR Get stylesheet font

WR Get stylesheet font (area ; stylesheetNumber) -> 戻り値

引数	型	説明
area	倍長整数	→ 4D Writeエリア
stylesheetNumber	倍長整数	→ スタイルシート番号
戻り値	文字	→ フォント名、フォントが割り当てられていない場合は空の文字列

説明

*WR Get stylesheet font*は、*area*で参照される4D Writeエリア内にある*styleSheetNumber*を持つスタイルシートに割り当てられたフォントの名前を返します。スタイルシートは先頭から末端に番号が振られ、スタイルシートダイアログに一覧表示されます。このスタイルシート用に定義されたフォントがない場合は、空の文字列が返されます。

例題

システムに入っていない特定のフォントが使われているスタイルシートからフォント属性をなくしたいものとします。:

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

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

引数	型	説明
area	倍長整数	⇒ 4D Writeエリア
stylesheetNumber	倍長整数	⇒ スタイルシート番号
name	文字	⇒ スタイルシート名
applyTo	整数	⇒ 0=文字、1=段落
protected	整数	⇒ 0=保護されていない、1=保護されている
shortcut	文字	⇒ 1つの文字、ショートカットが割り当てられていない場合は空の文字列

説明

*WR GET STYLESHEET INFO*は、*area*で参照される4D Writeエリア内にある*stylesheetNumber*を持つスタイルシートに関する情報を取り出します。

*name*は、スタイルシートのタイトルを返します。

applyTo:

- *applyTo*が0の場合、スタイルシートは文字列のみが対象となります。
- *applyTo*が1の場合、スタイルシートは段落のみが対象となります。

protected:

- *protected*が0の場合、スタイルシートは保護されません。つまり、これはシステムに用意されたスタイルシートではありません。
- *protected*が1の場合、スタイルシートは保護されます。これはシステムが用意するスタイルシートなので削除できません。

*shortcut*はスタイルシートに割り当てられたショートカットを返します。ただし1つの文字しかショートカットに利用することはできません。ショートカットを利用するには、Windows上では**Ctrl**キー、Mac OS上では**command**キーと共に、この引数に渡されるキーを押下する必要があります。

*shortcut*が空の文字列の場合、*stylesheetNumber*に割り当てられるショートカットはありません。

例題

WR SET STYLESHEET INFO, *WR DELETE STYLESHEET* そして *WR UPDATE STYLESHEET* コマンドの例題参照

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

引数	型		説明
area	倍長整数	→	4D Writeエリア
stylesheetNum	倍長整数	→	スタイルシート番号
tabNumber	倍長整数	→	タブ番号
position	倍長整数	←	タブ位置
justification	整数	←	タブの揃え位置
fillCharacter	文字	←	選択された埋め込み文字

説明

*WR GET STYLESHEET INFO*は、*tabNumber*で渡された番号のタブ設定を取り出します。このタブ設定は*area*で参照される4D Writeエリア内にある*stylesheetNumber*を持つスタイルシートに属します。

スタイルシート内のタブ数を知るには、タブストップの数を返す*WR Get stylesheet text prop*(*area*;*stylesheetNumber*;*wr tab*;*applyTo*)を用います。

*position*はドキュメントの左マージンからタブストップまでの距離です。これは、現在そのドキュメントのデフォルト単位となっているもので表されます。

*alignment*はタブの行揃えタイプです:

値 テキスト揃え

- 1 左揃え
- 2 中央揃え
- 3 右揃え
- 4 小数点揃え
- 5 縦区切り

*fillCharacter*はASCIIコード33から127までの文字を使用することができます。*fillCharacter*が空の文字列の場合、タブ設定に用いられる埋め込み文字はありません。

例題

各スタイルシートのタブストップに対する埋め込み文字を変更し、ドキュメントに反映させたいものとします。

```

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

```

WR Get stylesheet text prop (area ; stylesheetNumber ; property ; applyTo) -> 戻り値

引数	型	説明
area	倍長整数	➡ 4D Writeエリア
stylesheetNumber	倍長整数	➡ スタイルシート番号
property	整数	➡ 読み込むテキストプロパティの番号
applyTo	整数	➡ 0=プロパティは適用されない、1=プロパティが適用される
戻り値	実数	➡ property引数の値により異なる

説明

WR Get stylesheet text propは、propertyで渡されたプロパティがareaの4D Writeエリア内の選択した範囲に適用されているかどうかを調べることができます。

property:

- property=7 (wr font number定数)の場合、返される値は内部的な番号です。4D Writeは使用されるフォントにフォント番号を順番に割り振ります。このフォント番号は、WR SET STYLESHEET TEXT PROPでのみ使用されます。フォント名に関する基本的な操作については、WR Get stylesheet fontやWR SET STYLESHEET FONTを使用することをお勧めします。
- property 15 (wr stylesheet number 定数) は、この関数では機能しません。
- property = 64 (wr tab 定数) の場合、WR Get stylesheet text propはスタイルシートのタブ設定を返します。

カラーに関するプロパティの戻り値は、(4Dや4D Writeの旧バージョンと同じように) 次のようなフォーマットで表されます: 0x00RRGGBB。RGBの値を分割するにはWR COLOR TO RGBを使用します。

-1がプロパティ11(wr strikethrough color定数)、12(wr underline color定数)または13(wr shadow color定数)に対して返されると、これらの要素はテキスト内では同じカラーになります。

-1がプロパティ10 (wr text back color定数)に対して返されると、選択されたテキストの背景色はありません。

Note: propertyは定数を使用して設定できます。

テキストプロパティ定数の一覧は"[WR Text properties](#)"を参照してください。

applyToが1の場合、スタイルシートはプロパティを適用します。

applyToが0の場合、スタイルシートはプロパティを適用しません。

例題

WR UPDATE STYLESHEET, WR GET STYLESHEET TAB コマンドの例題参照。

WR SET STYLESHEET FONT

WR SET STYLESHEET FONT (*area* ; *stylesheetNumber* ; *font*)

引数	型		説明
<i>area</i>	倍長整数	⇒	4D Writeエリア
<i>stylesheetNumber</i>	倍長整数	⇒	スタイルシート番号
<i>font</i>	文字	⇒	フォント名

説明

*WR SET STYLESHEET FONT*は、*area*で参照される4D Writeドキュメント内の*styleSheetNumber*で渡されたスタイルシートの文字フォントを変更します。

*font*には使用したいフォントの名前を渡します。選択した範囲にスタイルシートを適用したい場合は、*font*に空の文字列を渡します。

*font*がシステムにインストールされていない場合、エラー1077 (Font not in system) が生成されます。

例題

[WR SET STYLESHEET INFO](#)の例題参照

WR SET STYLESHEET INFO

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

引数	型	説明
area	倍長整数	⇒ 4D Writeエリア
stylesheetNumber	倍長整数	⇒ スタイルシート番号
name	文字	⇒ スタイルシートの名前
applyTo	整数	⇒ 0=文字、1=段落
shortCut	文字	⇒ ショートカットに使用する文字、ショートカットを割り当てない場合空の文字列

説明

WR SET STYLESHEET INFOを使用して、*styleSheetNumber*で渡される参照番号のスタイルシートのプロパティを修正できます。このスタイルシートは*area*の参照番号を持つ4D Writeドキュメント内に含まれています。スタイルシート番号は、スタイルシートがスタイルシートドロップダウンリストまたはスタイルシートダイアログ内のリストに表示される際の順序に対応しています。

name: *name*が空の文字列の場合、オリジナルのスタイルシート名は変更されません。名前は31文字以下でなければなりません。

警告: 2つのスタイルシートが同じ名前を持つことはできますが、これらは常に異なる参照番号を持っています。

applyTo: *applyTo*-1の場合、現在の値がそのまま保持されます。"**WR Parameters**"テーマの以下の定数を渡すこともできます:

定数	型	値	コメント
wr apply to characters	倍長整数	0	文字にスタイルシートを適用する。
wr apply to paragraphs	倍長整数	1	段落にスタイルシートを適用する。

段落のスタイルシートは、先頭または最終の段落だけを選択した場合でも常に選択範囲内のすべての段落に適用されます。デフォルトでは、新規に作成されるスタイルシートは文字用のスタイルシートとなります。

shortCut: オプションの*shortCut*引数を使用して、スタイルシートにキーボードショートカットを割り当てることができます。ただし1つの文字しかショートカットに利用することはできません。ショートカットを利用するには、Windows上ではCtrlキー、Mac OS上ではcommandキーと共に、この引数に渡されるキーを押下する必要があります。4D Writeに標準で備わっているキーボードショートカットとコンフリクトを起こさないように数値を用いることをお勧めします。

*shortCut*が省略または空の文字列の場合、スタイルシートに割り当てられるショートカットはありません。

styleSheetNumber: スタイルシート番号を同じものにしておくには、WR GET STYLESHEET INFOから返される参照番号を使用しなければなりません。

例題

"Title"スタイルシートの定義を次のように修正したいものとします: 名前は"Title14"に変更し、フォントはTimes、14ポイント、太字、青に設定します。

```
NbStyles:=WR Count(Area;12)
For($i;1;NbStyles)
  WR GET STYLESHEET INFO(Area;$i;$Name;$ApplyTo;$Protected;$Shortcut)
  If($Name="Title")
    WR SET STYLESHEET INFO(Area;$i;"Title 14";$ApplyTo;$Shortcut)
    WR SET STYLESHEET FONT(Area;$i;"Times")
    WR SET STYLESHEET TEXT PROP(Area;$i;wr_font_size;14;1)
    WR SET STYLESHEET TEXT PROP(Area;$i;wr_bold;1;1)
    WR SET STYLESHEET TEXT PROP(Area;$i;wr_text_color;212;1)
  End if
End for
```

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

引数	型		説明
area	倍長整数	→	4D Writeエリア
stylesheetNumber	倍長整数	→	スタイルシート番号
tabNumber	倍長整数	→	タブ番号
position	倍長整数	→	新しいタブ位置
alignment	整数	→	新しいタブ揃えの値
fillChar	文字	→	選択された埋め込み文字

説明

WR SET STYLESHEET TABを使用して、*styleSheetNumber*で渡された番号のスタイルシートに属している、*tabNumber*で渡された番号を持つタブストップのパラメータを変更できます（タブは左から右にカウントされ、スタイルシートはスタイルシートダイアログに示されるように上から下にカウントされます）。WR SET STYLESHEET TABは、*position*にタブを移動し、タブの行揃えだけでなく埋め込み文字の設定も行います。

このコマンドは、選択されたテキストがそのスタイルシートを使用している場合、影響を与えません。

- スタイルシートを用いるテキストを更新したい場合は、スタイルシート定義内容を修正した後にWR UPDATE STYLESHEETを呼び出します。
- スタイルシートと減じ選択されているテキスト両方に即座にスタイルシートの新しいタブプロパティを適用するには、WR APPLY STYLESHEETを用います。

あるタブストップがすでにスタイルシート内の新しい場所に存在している場合、それはこのコマンドで設定されるタブストップによって置き換えられます。

*position*は、ドキュメントの左マージンからタブストップまでの距離です。*position*は現在のドキュメントのデフォルト単位で表されます。タブストップの位置を変更しない場合は*position*に-1を渡します。

*alignment*には、タブストップで選択したい行揃えのタイプを指定します。この場合下記の値または"WR Tabs"テーマの定数どちらかを使用することができます：

定数	型	値	コメント
wr left tab	倍長整数	1	左揃え
wr centered tab	倍長整数	2	中央合わせ
wr right tab	倍長整数	3	右揃え
wr decimal tab	倍長整数	4	小数点
wr vertical separator tab	倍長整数	5	縦分割

*fillCharacter*には、ASCIIコード33から127までの文字を使用することができます。この文字は修正されたタブストップと同じフォントで表示されます。

例題

WR GET STYLESHEET TABコマンドの例題参照

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

引数	型	説明
area	倍長整数	⇒ 4D Writeエリア
stylesheetNumber	倍長整数	⇒ スタイルシート番号
property	倍長整数	⇒ 設定するプロパティ
value	倍長整数	⇒ 選択したプロパティの値
apply	整数	⇒ 1=プロパティに値を適用、0=プロパティに値を適用しない

説明

WR SET STYLESHEET TEXT PROPを使用して、stylesheetNumberに渡されたスタイルシートのテキストプロパティを修正することができます。

- このスタイルシートを現在使用しているすべてのテキストを更新したい場合は、スタイルシート定義内容を修正した後にWR UPDATE STYLESHEETを呼び出します。
- このコマンドを使用して即座にスタイルシートと現在選択されているエリアの両方に新しく設定したスタイルシートのテキストプロパティを適用したい場合は、WR APPLY STYLESHEETを用います。
- valueに与えられる意味は、使用されるpropertyに依存します。
propertyがwr boldまたは0の場合、valueの値は1 (True) または0 (False) のいずれかになります。
propertyがwr font sizeまたは8の場合、valueの値は9, 10, 12...なおですが255を超えてはなりません。

Note: propertyとvalue は定数を使用して設定できます。

テキストプロパティとその値の定数一覧は"**WR Text properties**"と"**WR Text properties values**"定数テーマを参照してください。"**WR Text properties**"定数に関する説明はWR SET TEXT PROPERTYコマンドの説明を参照してください。

- プロパティへの変更を行いたい場合は、applyに1を渡します。これを行うとvalueはプロパティの新しい値を定義します。
- プロパティへの変更を行いたくない場合は、applyに0を渡します。これを行うとvalueは何も効果もありません。

例題

WR SET STYLESHEET INFOの例題参照

WR UPDATE STYLESHEET (area ; stylesheetNumber)

引数	型	説明
area	倍長整数	⇒ 4D Writeエリア
stylesheetNumber	倍長整数	⇒ スタイルシート番号

説明

WR UPDATE STYLESHEETは、areaで参照される4D Writeエリア内のstylesheetNumberで参照されたスタイルシートを使用して、すべてのテキストの表示フォーマットを更新します。このコマンドを実行すると、参照されたスタイルシート上で書式設定されているすべてのテキストはこのスタイルシートで現在設定されている内容に従って新しくフォーマットされます。

例題

"LayoutPar"スタイルシート内のタブストップを置き換え、スタイルシートが適用されているテキストエリアを更新したいものとしてします:

```
  `スタイルシート番号を探す
$StyleSheetNb:=WR Count(Area;wr_nb_stylesheets)
For($i;1;$StyleSheetNb)
  WR GET STYLESHEET INFO(Area;$i;$Name;$ApplyTo;$Prot;$Shortcut)
  If($Name="LayoutPar")
    SheetNumber:=$i
  End if
End for
  `スタイルシート中のタブストップ数を取得
$NbTab:=WR Get stylesheet text prop(Area;SheetNumber;wr_tab;Apply)
  `すべてのタブストップを削除
For($i;1;$NbTab)
  WR DELETE STYLESHEET TAB(Area;SheetNumber;1)
End for
  `新しいタブを挿入
WR ADD STYLESHEET TAB(Area;SheetNumber;10;wr_left_tab;Char(126))
...
  `スタイルシートが適用されている段落を更新
WR UPDATE STYLESHEET(Area;SheetNumber)
```

WRタブ

 タブコマンドについて

 WR ADD TAB

 WR DELETE TAB

 WR GET TAB

 WR SET TAB

タブコマンドについて

このテーマのコマンドを使用して4D Writeエリア内のタブストップの位置とプロパティをコントロールできます。
タブストップのプロパティの読み書きや既存のタブの削除、新規作成ができます。

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

引数	型		説明
area	倍長整数	⇒	4D Writeエリア
position	倍長整数	⇒	タブ位置
justification	整数	⇒	揃え値
fillCharacter	文字	⇒	選択された埋め込み文字

説明

WR ADD TABを使用すれば、ドキュメントの左余白から`position`に指定された位置に新規のタブを追加することができます。またこのコマンドを使用すれば、埋め込み文字および新規のタブストップの行揃えを設定することもできます。

このタブストップは、選択されているすべての段落に追加されます。この位置にすでにタブストップがある場合、作成したタブストップに置き換えられます。

`position`は、（ドキュメントのデフォルト単位で表されている）左余白からの距離です。

オプションの`justification`はタブストップのタイプを指定します。"WR Tabs"テーマで定義された定数を指定することができます:

定数	型	値	コメント
wr left tab	倍長整数	1	左揃え
wr centered tab	倍長整数	2	中央合わせ
wr right tab	倍長整数	3	右揃え
wr decimal tab	倍長整数	4	小数点
wr vertical separator tab	倍長整数	5	縦分割

`justification`が省略されている場合は、左揃えのタブが作成されます。

オプションの`fillCharacter`は、ASCIIコードの33から127までの間の文字を指定することができます。この文字は、タブストップと同じフォントを使用して追加されます。

`fillCharacter`が省略されている場合や、空の文字列が指定されている場合は、埋め込み文字は挿入されません。

例題

以下の例は、左余白からドットを埋め込み文字として、50単位離れた位置に左タブストップを作成します。

```
WR ADD TAB(area;50;wr_left_tab;".")
```

WR DELETE TAB

WR DELETE TAB (area ; tabNum)

引数	型		説明
area	倍長整数	⇒	4D Writeエリア
tabNum	倍長整数	⇒	タブ番号

説明

*WR DELETE TAB*は、*area*で指定された4D Writeエリアから、*tabNum*で指定（左から右へ数えて）されたタブを削除します。他のタブが同じ位置に配置された場合、それらのタブも削除されます。

Note: 選択範囲にいくつかの段落が含まれている場合、タブの番号付けは最後に選択された段落に適用されます。

例題

ドキュメントからすべてのタブを削除します:

```
C_LONGINT (Area; $i; $TabNum; $uniform)
  `エリアの先頭にカーソルを挿入
WR SET SELECTION(Area; 0; 0)
  `ドキュメント内の段落数をカウント
NbParag:=WR Count (Area; wr_nb_paragraphs)
  `段落ごとに
For ($i; 1; NbParag)
  `段落の位置を取得
  WR GET PARAGRAPHS (Area; START; Pos)
  `段落内部に入る
  WR SET SELECTION(Area; START+1; START+1)
  `タブ数を取得
  $TabNum:=WR Get text property(Area; wr_tab; $uniform)
  While ($TabNum#0)
    WR DELETE TAB(Area; 1)
    $TabNum:=$TabNum-1
  End while
  `最後に処理した段落のすぐ後に移動
  WR GET SELECTION(Area; Pos; Pos)
End for
```

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

引数	型		説明
area	倍長整数	⇒	4D Writeエリア
tabNumber	倍長整数	⇒	タブ番号
position	倍長整数	⇐	タブ位置
alignment	整数	⇐	タブの文字揃え
fillCharacter	文字	⇐	埋め込み文字

説明

WR GET TABは、areaの現在のルーラー内でtabNumberによって指定されたタブの位置と行揃え、埋め込み文字を返します。現在のルーラーはカーソルポイントが表示されているルーラーもしくはいくつかの段落が選択された最後のルーラーです。

- *tabNumber*: 段落内のタブの数を知るには、タブストップの数を返すWR Get text property (Area;45;1)を使用します。タブ番号を順にくり返し、処理することで、現在のルーラーの引数すべてを取得することができます。
- *position*: *position*は、ドキュメントの左マージンからタブストップまでの距離です。これは現在そのドキュメントのデフォルト単位となっているもので表されます。
- *alignment*: *alignment*は、タブの行揃えのタイプです。

定数	型	値	コメント
wr left tab	倍長整数	1	左揃え
wr centered tab	倍長整数	2	中央合わせ
wr right tab	倍長整数	3	右揃え
wr decimal tab	倍長整数	4	小数点
wr vertical separator tab	倍長整数	5	縦分割

- *fillCharacter* は、ASCIIコードの33から127までの間の文字を指定することができます。fillCharacterが空の文字列の場合、タブ設定に用いられる埋め込み文字はありません。

例題

WR SET TABとWR DELETE TABコマンドの例題参照

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

引数	型		説明
area	倍長整数	→	4D Writeエリア
tabNumber	倍長整数	→	タブ番号
position	倍長整数	→	新しいタブ位置
alignment	整数	→	タブのテキスト揃えの新しい値
fillCharacter	文字	→	新しい埋め込み文字

説明

WR SET TABを使用して、tabNumberで指定されたタブストップに関する引数を設定することができます（タブは左から右へ数えます）。WR SET TABは、positionで指定された位置にタブストップを移動し、さらに行揃え、埋め込み文字を設定します。

選択されたタブストップは、現在選択されている範囲内にあるすべての段落について修正されます。新しく指定された位置にすでにタブストップがある場合、定義したタブストップに置き換えられます。

positionは左余白からの距離です。positionはドキュメントの現在のデフォルト単位で表されます。

alignmentはタブストップの行揃えを指定します。値または定数を指定することができます。テキスト揃えを変更したくない場合は-1を渡します。そうでない場合、"WR Tabs"テーマの以下の定数を使用できます：

定数	型	値	コメント
wr left tab	倍長整数	1	左揃え
wr centered tab	倍長整数	2	中央合わせ
wr right tab	倍長整数	3	右揃え
wr decimal tab	倍長整数	4	小数点
wr vertical separator tab	倍長整数	5	縦分割

fillCharacterはASCIIコードの33から127までの間の文字を指定することができます。この文字は、変更されたタブストップとしてフォントで表示されます。

例題

選択された範囲において、168ポイントの位置にあるタブを削除し、252ポイントの位置のタブを280ポイントに移動して"\$"文字を埋め込み文字とします：

```

C_LONGINT (Area; $i; $Nbtab; $Unit; $uniform; $Justif)
C_REAL ($Pos)
C_TEXT ($fill)
$Nbtab:=WR Get text property(Area;wr_tab;$uniform)
`現在の単位を記憶
$Unit:=WR Get doc property(Area;wr_unit)
If ($Unit#2)
`単位をポイントに設定する
WR SET DOC PROPERTY(Area;wr_unit;2)
End if
$i:=1
Repeat
WR GET TAB(Area;$i;$pos;$Justif;$fill)
Case of
: ($Pos=168)
`168ポイントに位置するタブを削除
WR DELETE TAB(Area;$i)
$Nbtab:=$Nbtab-1
: ($Pos=252)
`252ポイントに位置するタブを280ポイントに移動
WR SET TAB(Area;$i;280;$Justif;"$")
$i:=$i+1
End case
Until ($i=$Nbtab)

```

元の単位に戻す

```
WR SET DOC PROPERTY(Area;wr_unit;$Unit)
```

WRデータベースオブジェクト

 データベースオブジェクトコマンドについて

 WR GET DATE AND TIME FORMAT

 WR Get HTML expression

 WR GET HYPERLINK

 WR GET PAGE NUMBER FORMAT

 WR GET REFERENCE

 WR Get RTF expression

 WR INSERT DATE AND TIME

 WR INSERT EXPRESSION

 WR INSERT FIELD

 WR INSERT HTML EXPRESSION

 WR INSERT HYPERLINK

 WR INSERT PAGE NUMBER

 WR Insert picture area

 WR INSERT RTF EXPRESSION

データベースオブジェクトコマンドについて

このテーマのコマンドと関数を使用して、4Dオブジェクト (メソッド、変数、関数、フィールド、ページ番号、4D Writeピクチャエリアなど) にアクセスできます。

またこれらのオブジェクトが4D Writeエリア上に置かれているとき、*WR GET REFERENCE*コマンドを使用してオブジェクトの情報を取得できます。

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

引数	型		説明
area	倍長整数	→	4D Write エリア
dateFormat	整数	←	日付フォーマットの数値
timeFormat	整数	←	時間フォーマットの数値

説明

WR GET DATE AND TIME FORMATは、選択された動的な日付や時間の表示フォーマットを知るために使用します。

dateFormat 引数は、挿入された参照の日付フォーマットの数値を返します。受け取った値は4Dの"**Date Display Formats**"テーマの定数および4D Writeの"**WR Parameters**"テーマの定数と比較できます:

定数	型	値	コメント
Internal date abbreviated	倍長整数	6	Dec 29, 2006
Internal date long	倍長整数	5	December 29, 2006
Internal date short	倍長整数	7	2006/12/29
Internal date short special	倍長整数	4	06/12/29 (しかし 1986/12/29 または 2096/12/29)
System date abbreviated	倍長整数	2	
System date long	倍長整数	3	
System date short	倍長整数	1	

定数	型	値	コメント
wr no date format	倍長整数	0	日付フォーマットなし

timeFormat引数は、挿入された参照の時間フォーマットの数値を返します。。受け取った値は4Dの"**Time Display Formats**"テーマの定数および4D Writeの"**WR Parameters**"テーマの定数と比較できます:

定数	型	値	コメント
HH MM	倍長整数	2	01:02
HH MM AM PM	倍長整数	5	1:02 AM
HH MM SS	倍長整数	1	01:02:03
Hour min	倍長整数	4	1時2分
Hour min sec	倍長整数	3	1時2分3秒

定数	型	値	コメント
wr no time format	倍長整数	0	時間フォーマットなし

WR Get HTML expression

WR Get HTML expression (area) -> 戻り値

引数	型		説明
area	倍長整数	→	4D Writeエリア
戻り値	テキスト	↺	HTML式の内容

説明

WR Get HTML expression コマンドを使用して、*area*内で現在選択されているHTML式をテキストとして取り出します。
4D Writeドキュメントに含まれるHTML式を選択するには、*WR Count*(Area;wr nb HTML expressions) を使用し、そして *WR SELECT*(Area;13;\$loop)をループで使用します。

例題

4D Writeドキュメントに含まれるHTML式を取り出します:

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

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

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

引数	型	説明
area	倍長整数	⇒ 4D Write エリア
linkType	整数	⇒ ハイパーリンクタイプ: 0= メソッド、 1=URL、 2=4D Write ドキュメント
urlStyle	整数	⇒ URL の外観: 1= デフォルトスタイル、 0= カスタムスタイル
linkLabel	テキスト	⇒ リンクの表示テキスト (表示/値モード)
linkContent	テキスト	⇒ ハイパーリンクの値
methodRef	倍長整数	⇒ \$3の値、メソッドの第3引数 (リンクタイプがメソッドの場合)

説明

WR GET HYPERLINK は、*area*に含まれる、選択されたハイパーリンクのプロパティを返します。

linkType:

- メソッドタイプリンクの場合、*linkType* は0 を返します。
- URLタイプリンクの場合、*linkType* は1 を返します。
- ドキュメントタイプリンクの場合、*linkType* は2 を返します。

urlStyle

- リンクスタイルがデフォルトに設定されている場合、*urlStyle* は1を返します。
- リンクスタイルがカスタマイズされている場合、*urlStyle* は0を返します。この場合WR GET TEXT PROPERTYコマンドを使用してスタイル情報を取得できます。

linkLabel:

linkLabel は (表示/値モードのときに) リンクに表示されるテキストを返します。

linkContent:

linkContent はハイパーリンクの値を返します。つまり:

- 4Dメソッドタイプのリンクの場合、メソッド名
- URLタイプリンクの場合、完全URL
- ドキュメントタイプリンクの場合、完全ドキュメントパス

methodRef:

methodRef は (4Dメソッドタイプのリンクの場合) 呼び出されたメソッドに渡される値を返します。

4D Writeドキュメントに含まれるハイパーリンクを選択するには、WR Count(Area;wr nb hyperlinks) コマンドの後にループ中でWR SELECT(Area;12;\$loop)を使用します。

WR GET PAGE NUMBER FORMAT

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

引数	型	説明
area	倍長整数	⇒ 4D Writeエリア
format	整数	⇒ フォーマットのタイプ
numType	整数	⇒ ページ番号のタイプ。0= ページ番号、1= 総ページ数

説明

WR GET PAGE NUMBER FORMATコマンドは、挿入したページ番号参照に使用される表示フォーマットおよびナンバリングのタイプを取得するために使用します。

format 引数は参照の表示フォーマット番号を返します。受け取った値は"WR Page number formats"テーマの定数と比較できます:

定数	型	値	コメント
wr 123	倍長整数	0	1, 2, 3...
wr abc	倍長整数	1	a, b, c...
wr ABC	倍長整数	2	A, B, C...
wr i ii iii	倍長整数	3	i, ii, iii...
wr I II III	倍長整数	4	I, II, III...

numType 引数は参照がページ番号であるときに1を、総ページ数であるときに1を返します。

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

引数	型		説明
area	倍長整数	→	4D Write エリア
info1	整数	←	参照に関する最初の情報
info2	整数	←	参照に関する2 番目の情報
name	文字	←	参照名を取得
type	整数	←	参照タイプを取得
numFormat	文字	←	数値のフォーマット
dateFormat	整数	←	日付フォーマットの数値
timeFormat	整数	←	時間フォーマットの数値

説明

WR GET REFERENCEコマンドは、4D Write area内で選択された参照に関する情報を返します。

選択された参照に関する情報はinfo1、info2、name および type に返されます。また数値、日付、時間の表示フォーマットも知ることができます。

info1、info2およびname に返される値はtypeによって変わります。選択されたオブジェクトが参照でない場合、type には0が返されます。

- type=1の場合、参照はフィールドです。info1 はテーブル番号を指定します。info2 はフィールド番号を示します。name は空です。
- type=2の場合、参照は数値式を指定します。info1 およびinfo2 は0になります。name には変数名、あるいは数値式が返されます。

numFormat には、数値フィールド/数値式（実数、倍長整数、整数）に対して設定された表示フォーマットが返ります。表示フォーマットが設定されていない場合や数値でないフィールド/数値式でない場合には、ヌルストリングが返ります。

日付タイプの場合、dateFormat には選択されたフィールド/数値式に割り当てられた日付フォーマットの数値が返されます。それ以外の場合は0 を返します。

日付フォーマットが返されたら、"**Date Display Formats**"の4D定数と比較できます：

定数	型	値	コメント
System date short	倍長整数	1	
System date abbreviated	倍長整数	2	
System date long	倍長整数	3	
Internal date short special	倍長整数	4	06/12/29 (しかし 1986/12/29 または 2096/12/29)
Internal date long	倍長整数	5	December 29, 2006
Internal date abbreviated	倍長整数	6	Dec 29, 2006
Internal date short	倍長整数	7	2006/12/29

時間タイプの場合、timeFormatには選択されたフィールド/数値式に割り当てられた時間フォーマットの数値が返されます。それ以外の場合は0 を返します。

時間フォーマットが返されたら、"**Time Display Formats**"の4D定数と比較できます：

定数	型	値	コメント
HH MM SS	倍長整数	1	01:02:03
HH MM	倍長整数	2	01:02
Hour min sec	倍長整数	3	1時2分3秒
Hour min	倍長整数	4	1時2分
HH MM AM PM	倍長整数	5	1:02 AM

例題 1

WR SELECTの例題参照

例題 2

この例題ではユーザが選択したオブジェクトが参照かを判定します。また選択したオブジェクトがフィールドか式かをユーザに表示します。

```
WR GET REFERENCE(Letter;$Table;$Field;$Name;$Type)
Case of
:($Type=0) `テキストまたは何も選択されていない
    ALERT("Selected text or nothing")
:($Type=1)
    ALERT("Selected the field "+Field name($Table;$Field))
:($Type=2)
    ALERT("Selected the expression named "+$Name)
End case
```

WR Get RTF expression

WR Get RTF expression (area) -> 戻り値

引数	型		説明
area	倍長整数	→	4D Writeエリア
戻り値	テキスト	↩	RTF式の内容

説明

WR Get RTF expression コマンドは、*area* 内で現在選択されている RTF 式のテキストを取得するために使用します。

4D Write ドキュメントに含まれる RTF 式を選択するには、*WR Count*(Area;wr nb RTF expressions) コマンドの後にループの中で *WR SELECT*(Area;14;\$loop) を使用します。

例題

4D Write ドキュメント中の RTF 式を取得します:

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

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

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

引数	型	説明
area	倍長整数 →	4D Writeエリア
dateFormat	整数 →	日付フォーマットの数値
timeFormat	整数 →	時間フォーマットの数値

説明

WR INSERT DATE AND TIMEコマンドは、ダイナミックな日付あるいは/または時間を表示する参照にカーソルの位置を挿入します。ドキュメント中で選択された現在のテキストがある場合は、挿入される参照に置き換えられます。

dateFormatは、日付の参照に対する表示フォーマットを設定します。

"Date Display Formats"テーマの4D定数または"WR Parameters"テーマの4D Write定数を使用しなければなりません:

定数	型	値	コメント
System date short	倍長整数	1	
System date abbreviated	倍長整数	2	
System date long	倍長整数	3	
Internal date short special	倍長整数	4	06/12/29 (しかし 1986/12/29 または 2096/12/29)
Internal date long	倍長整数	5	December 29, 2006
Internal date abbreviated	倍長整数	6	Dec 29, 2006
Internal date short	倍長整数	7	2006/12/29

定数	型	値	コメント
wr no date format	倍長整数	0	日付フォーマットなし

timeFormatは、時間の参照に対する表示フォーマットを設定します。

"Time Display Formats"テーマの4D定数または"WR Parameters"テーマの4D Write定数を使用しなければなりません:

定数	型	値	コメント
HH MM SS	倍長整数	1	01:02:03
HH MM	倍長整数	2	01:02
Hour min sec	倍長整数	3	1時2分3秒
Hour min	倍長整数	4	1時2分
HH MM AM PM	倍長整数	5	1:02 AM

定数	型	値	コメント
wr no time format	倍長整数	0	時間フォーマットなし

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

引数	型	説明
area	倍長整数	→ 4D Write エリア
expression	文字	→ 挿入する式
numFormat	文字	→ 数値フォーマット
dateFormat	整数	→ 日付フォーマットの番号
timeFormat	整数	→ 時間フォーマットの番号
destination	倍長整数	→ ピクチャーを配置する場所
size	倍長整数	→ 0=サイズを固定、1=サイズを調整

説明

WR INSERT EXPRESSION コマンドは *area* に *expression* の参照を挿入し、現在選択されているテキストと置き換えます。

expression は、値を返す有効な4D式でなければなりません。*expression* は、4Dの変数や関数、値を返すステートメント、外部関数やユーザー定義の関数 (プロジェクトメソッド) です。*expression* が変数の場合、ダブルクォーテーション (") の間に名前を渡します。

expression がキャリッジリターンやタブを含む値を返す場合、4D Write はその *expression* がある段落のルーラーに従ってテキストをフォーマットします。

オプション引数 *numFormat* は、数値フィールド (実数、倍長整数、整数) に対するフォーマットを指定します。これは、すべての表示フォーマットを含むことができます。また指定しても指定しなくてもかまいません (例: "###,##")。この引数が不要な場合には空の文字列を指定します。また、後の引数をすべて省略する場合には、この引数も省略することができます。

オプション引数 *dateFormat* は、日付タイプの式のフォーマットを指定します。存在する日付フォーマットを指定する数を含んでいる必要があります。この引数を指定したくない場合0を渡すか、後の引数を省略する場合はこの引数も省略できます。

日付フォーマットを指定する場合、**Date Display Formats** テーマの 4D定数あるいは **WR ParametersWR**

Parameters テーマの 4D Write定数を渡します:

定数	型	値	コメント
System date short	倍長整数	1	
System date abbreviated	倍長整数	2	
System date long	倍長整数	3	
Internal date short special	倍長整数	4	06/12/29 (しかし 1986/12/29 または 2096/12/29)
Internal date long	倍長整数	5	December 29, 2006
Internal date abbreviated	倍長整数	6	Dec 29, 2006
Internal date short	倍長整数	7	2006/12/29

定数	型	値	コメント
wr no date format	倍長整数	0	日付フォーマットなし

オプション引数 *timeFormat* は、時間タイプの式のフォーマットを指定します。存在する時間フォーマットを指定する数を含んでいる必要があります。この引数を指定したくない場合0を渡すか、省略します。

時間フォーマットを指定する場合、**Time Display Formats** テーマの 4D定数あるいは **WR Parameters** テーマの 4D Write定数を渡します:

定数	型	値	コメント
HH MM SS	倍長整数	1	01:02:03
HH MM	倍長整数	2	01:02
Hour min sec	倍長整数	3	1時2分3秒
Hour min	倍長整数	4	1時2分
HH MM AM PM	倍長整数	5	1:02 AM

定数	型	値	コメント
wr no time format	倍長整数	0	時間フォーマットなし

ピクチャー式を挿入する場合、任意の *destination* 引数を使用してドキュメントのどこにピクチャーを挿入するかを指定でき

まず、0より大きい値または**WR Parameters** テーマの4D Write定数を渡します:

定数	型	値	コメント
wr on left hand pages	倍数 長整数	-12	ピクチャはページ上に挿入され、偶数と奇数ページのヘッダが異なる場合、左側のページにのみ表示されます。
wr on right hand pages	倍数 長整数	-11	ピクチャはページ上に挿入され、偶数と奇数ページのヘッダが異なる場合、右側のページにのみ表示されます。そうでなければすべてのページに表示されます。
wr on current page	倍数 長整数	-4	ピクチャはページ上に挿入され、現在のページ (挿入ポイントあるいは現在選択されている箇所) に表示されます。
wr into the text flow	倍数 長整数	0	ピクチャはテキストフロー中に挿入されます。この場合他の引数は使用されず、ピクチャは挿入ポイントに挿入されるか現在選択されている箇所を置き換えます。
0より大きな値			ピクチャーは <i>destination</i> に渡されたページ番号に表示される。値は開始番号を考慮しなければなりません。

ピクチャー式を追加する際、オプションの *size* 引数で表示領域を保持するか調整するかを指定できます:

- *size*に 1を渡すと、式が評価される時、表示領域は新しいピクチャーサイズに調整されます。
- *size*に 0を渡すと、式が評価される時、新しいピクチャーのサイズに関わらず表示領域はそのまま保持されます。

例題

以下の2つの部分に別れた例は、4D Write エリアに4Dのプロセスメソッドへの参照を挿入する例です。前半に示すプロジェクトメソッドは、顧客に関連したインボイスを探し、インボイス番号と金額を連結します。

```

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

```

後半に示した例は、プロジェクトメソッド**SHOW INVOICES**をエリアに挿入します。4D Writeがエリアを表示または印刷する際に、各インボイスは各行に表示されます。

```
WR INSERT EXPRESSION(area;"SHOW INVOICES")
```

```
WR INSERT FIELD ( area ; numTable ; numField {; numFormat {; dateFormat {; timeFormat {; destination {; size}}}} )
```

引数	型		説明
area	倍長整数	→	4D Write エリア
numTable	整数	→	テーブル番号
numField	整数	→	フィールド番号
numFormat	文字	→	数値フォーマット
dateFormat	整数	→	日付フォーマットの番号
timeFormat	整数	→	時間フォーマットの番号
destination	倍長整数	→	ピクチャーを配置する場所
size	倍長整数	→	0=サイズを固定、1=サイズを調整

説明

WR INSERT FIELD コマンドは *area* にフィールドへの参照を挿入し、現在選択されているテキストと置き換えます。挿入されるフィールドは、*numTable* と *numField* で指定します。数値、日付、時間フィールドの場合は、表示フォーマットを設定できます。

オプション引数 *numFormat* は、数値フィールド（実数、倍長整数、整数）に対するフォーマットを指定します。任意の数値表示フォーマットを指定できます。また指定しても指定しなくてもかまいません（例："###,##"）。この引数が必要ない場合空文字を渡します。残りの引数を渡さない場合はこの引数も省略できます。

オプション引数 *dateFormat* は、日付タイプの式のフォーマットを指定します。存在する日付フォーマットを指定する数を含んでいる必要があります。この引数を指定したくない場合0を渡すか、後の引数を省略する場合はこの引数も省略できます。日付フォーマットを指定する場合、[Date Display Formats](#) テーマの4D定数あるいは [WR Parameters](#) テーマの4D Write 定数を渡します：

定数	型	値	コメント
System date short	倍長整数	1	
System date abbreviated	倍長整数	2	
System date long	倍長整数	3	
Internal date short special	倍長整数	4	06/12/29 (しかし 1986/12/29 または 2096/12/29)
Internal date long	倍長整数	5	December 29, 2006
Internal date abbreviated	倍長整数	6	Dec 29, 2006
Internal date short	倍長整数	7	2006/12/29

定数	型	値	コメント
wr no date format	倍長整数	0	日付フォーマットなし

オプション引数 *timeFormat* は、時間タイプの式のフォーマットを指定します。存在する時間フォーマットを指定する数を含んでいる必要があります。この引数を指定したくない場合0を渡すか、省略します。

時間フォーマットを指定する場合、[Time Display Formats](#) テーマの4D定数あるいは [WR Parameters](#) テーマの4D Write 定数を渡します：

定数	型	値	コメント
HH MM SS	倍長整数	1	01:02:03
HH MM	倍長整数	2	01:02
Hour min sec	倍長整数	3	1時2分3秒
Hour min	倍長整数	4	1時2分
HH MM AM PM	倍長整数	5	1:02 AM

定数	型	値	コメント
wr no time format	倍長整数	0	時間フォーマットなし

ピクチャー式を挿入する場合、オプションの *destination* 引数を使用してドキュメントのどこにピクチャーを挿入するか指定できます。0より大きい値または [WR Parameters](#) テーマの4D Write 定数を渡します：

定数	型	値	コメント
wr on left hand pages	倍 長 整 数	- 12	ピクチャはページ上に挿入され、偶数と奇数ページのヘッダが異なる場合、左側のページにのみ表示されます。
wr on right hand pages	倍 長 整 数	- 11	ピクチャはページ上に挿入され、偶数と奇数ページのヘッダが異なる場合、右側のページにのみ表示されます。そうでなければすべてのページに表示されます。
wr on current page	倍 長 整 数	-4	ピクチャはページ上に挿入され、現在のページ (挿入ポイントあるいは現在選択されている箇所) に表示されます。
wr into the text flow	倍 長 整 数	0	ピクチャはテキストフロー中に挿入されます。この場合他の引数は使用されず、ピクチャは挿入ポイントに挿入されるか現在選択されている箇所を置き換えます。
0より大きな 値	ピクチャーは	<i>destination</i> に渡されたページ番号に表示される。値は開始番号を考慮しなければなりません。	

ピクチャー式を追加する際、オプションの*size*引数で表示領域を保持するか調整するかを指定できます:

- *size*に1を渡すと、式が評価される時、表示領域は新しいピクチャーサイズに調整されます。
- *size*に0を渡すと、式が評価される時、新しいピクチャーのサイズに関わらず表示領域はそのまま保持されます。

WR INSERT HTML EXPRESSION (area ; htmlExpression)

引数	型		説明
area	倍長整数	→	4D Writeエリア
htmlExpression	テキスト	→	HTML式

説明

*WR INSERT HTML EXPRESSION*は、*htmlExpression* に格納されたHTML式を*area* に挿入します。式はカーソルの位置に挿入されます。このときテキストが選択されていればテキストが式に置換されます。

HTML式はオリジナルの4D Write ドキュメントには表示されませんが、ドキュメントをHTML形式で保存するとHTML式として挿入されます。HTMLテキストはWebブラウザを通して直接解析されます。したがって、あらゆるHTMLタグ（URL、スタイルマーカ、イメージ等）を含むことができます。

4D WriteドキュメントをHTMLとしてエクスポートすると、式は生成されたHTMLドキュメントに保存されます。

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

引数	型	説明
area	倍長整数	⇒ 4D Write エリア
linkType	整数	⇒ ハイパーリンクタイプ: 0= メソッド、 1=URL、 2=4D Write ドキュメント
urlStyle	整数	⇒ URL の外観: 1= デフォルトスタイル、 0= カスタムスタイル
linkLabel	テキスト	⇒ リンクの表示テキスト (表示/値モード)
linkContent	テキスト	⇒ ハイパーリンクの値
methodRef	倍長整数	⇒ \$3の値、メソッドの3番目の値 (リンクタイプがメソッドの場合)

説明

WR INSERT HYPERLINKコマンドは、area内の現在のカーソル位置またはテキストの現在選択している範囲にハイパーリンク参照を挿入します。

linkType

linkType は、挿入するハイパーテキストリンクのタイプを指定します。4D Writeには、URL タイプリンク、ドキュメントタイプリンク、メソッドタイプリンクの3つのハイパーテキストリンクを使用できます。

- **メソッドタイプリンク**は、参照をクリックすることによって4Dメソッドを実行します。メソッドに関数は使用できません。また引数を渡すこともできません。しかし、\$1、\$2、オプションの\$3を通じて2つまたは3つの値を受け渡すことができます:
 - \$1 (倍長整数) 4D Write エリア参照番号を含む
 - \$2 (テキスト) リンクラベルの文字列を含む
 - \$3 (倍長整数) 任意の数値を含む。methodRef 引数によってリンクを関連付け4D Write のユーザモードで使用することができる。
データベースをコンパイルする場合には、これらの引き数を使わなくても、\$1と\$2を倍長整数、\$3をテキストとして宣言する必要があります。
メソッドタイプリンクの場合linkTypeに0を挿入します。
- **URL タイプリンク**はlinkContent で指定したURL をデフォルトブラウザでオープンします。URL タイプリンクの場合は、linkTypeに1を挿入します。
- **ドキュメントタイプリンク**は、リンクをクリックするとlinkContent に格納されたパスによって現在のドキュメントを他のドキュメントに置換します。もちろんドキュメントを開くためにはドキュメントのフォーマットが4D_Write によって認識可能なものでなければなりません。ドキュメントタイプリンクの場合はlinkTypeに2を挿入します。

linkType引数には"WR Parameters"テーマの以下のいずれかの定数を渡します:

定数	型	値	コメント
wr method type link	倍長整数	0	メソッドタイプのリンクを挿入。
wr URL type link	倍長整数	1	URLタイプのリンクを挿入
wr document type link	倍長整数	2	ドキュメントタイプのリンクを挿入

urlStyle:

urlStyle を使用して、挿入したハイパーテキストリンクの外観を設定できます。この引数には"WR Parameters"テーマの以下のいずれかの 定数を渡します:

定数	型	値	コメント
wr custom link appearance	倍長整数	0	カスタマイズされたアピアランスの利用を許可します。この場合リンクを選択し、WR SET TEXT PROPERTYコマンドを使用してスタイルを定義できます。
wr default link appearance	倍長整数	1	デフォルトのハイパーリンクアピアランス (青色および下線) を保持します。デフォルトのカラーはWR SET DOC PROPERTYコマンドを使用してプログラムで変更できます。

定数wr custom link appearanceを使用してかつリンクスタイルを指定しない場合、リンクはカレントテキストとして表示されます (表示がリンクになりません)。

linkLabel:

linkLabel は、(表示/値モードでの) リンクの表示テキストを設定します。

linkContent:

linkContent は、ハイパーテキストリンクの値を設定します。値の性質は、リンクタイプに依存します:

- 4Dメソッドタイプリンクの場合、メソッド名を指定します。例 "Order_Clients"
- URLタイプリンクの場合、完全URLを設定します。例 "http://www.4d-japan.com"
- ドキュメントタイプリンクの場合、ドキュメントへのフルパスを設定します。例 " ("C:¥MyFolder¥MyDoc.4w7" Windows の場合) 、 ("HardDrive:MyFolder:MyDoc" MacOSの場合) "

methodRef:

methodRef は、リンクにメソッドタイプを指定したときに、呼び出すメソッドに受け渡す値を設定します。メソッドは、引数\$3 (倍長整数) 中の格納された値を受け取ることができます。

例題 1

4D Write エリアにWeb サイトのURL を挿入します:

```
WR INSERT HYPERLINK(area;wr URL type link;wr default link appearance;"Visit that great site";"http://www.MySite.com/")
```

例題 2

4D_Write ドキュメントに、ドキュメントタイプリンクによるプラットフォームに依存しないハイパーテキストのナビゲーション機能をあたえます。次のメソッドは、ドキュメントのパスを動的に管理します。:

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

例題 3

メソッドタイプリンクの使用例です。この例は、ユーザに対してドキュメントの特定の場所に名字と名前を入力を要求します。\$3 に設定した値 (メソッド名) によってメソッドを起動します。そのメソッドがユーザに名前を入力を要求し、リンクを入力された名前に置き換えます:

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

4D Writeエリアにメソッドタイプのリンクを挿入する:

```
$title:="Click to enter"
$method:="Hyperlink_Method"
WR INSERT TEXT (Area;"Last name: ")
WR INSERT HYPERLINK (Area;wr method type link;wr default link appearance;$title;$method;1)
```

```
WR INSERT TEXT(Area;Char(Carriage_return)+"First name: ")  
WR INSERT HYPERLINK(Area;wr method type link;wr default link appearance;"Click to  
enter";"Hyperlink_Method";2)
```

WR INSERT PAGE NUMBER (area ; format ; typeNum)

引数	型	説明
area	倍長整数	⇒ 4D Writeエリア
format	整数	⇒ フォーマットタイプ
typeNum	整数	⇒ 挿入する番号 0 = ページ番号, 1 = 総ページ数

説明

WR INSERT PAGE NUMBERコマンドを使用して、カーソル位置に、現在のページ番号または総ページ数の参照を挿入できます。子の参照はメインテキスト、フッター、またはヘッダーエリアに配置できます。WR SET FRAMEコマンドを使用して選択したエリアにカーソルを置くことができます。

format引数を使用して挿入する参照のフォーマットを指定します。この引数にはWR Page number formatsテーマの定数を渡せます:

定数	型	値	コメント
wr 123	倍長整数	0	1, 2, 3...
wr abc	倍長整数	1	a, b, c...
wr ABC	倍長整数	2	A, B, C...
wr i ii iii	倍長整数	3	i, ii, iii...
wr I II III	倍長整数	4	I, II, III...

typeNumオプション引数を使用して現在のページ番号を挿入するか、現在のドキュメントの総ページ数を挿入するか指定できます。定数wr page number (値0) を渡すかこの引数を省略した場合、現在のページ番号が挿入されます。定数wr total number of pages (値1)を渡すと、ドキュメントの総ページ数が挿入されます。

例題

以下のメソッド (OddPages) はカレントドキュメントのフッターに挿入される変数に書かれます:

```

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

```

WR Insert picture area

WR Insert picture area (area ; picture ; where) -> 戻り値

引数	型		説明
area	倍長整数	→	4D Write エリア
picture	ピクチャー	→	挿入する4D Write エリアピクチャ
where	整数	→	0= 挿入ポイント 1= ドキュメントの最後
戻り値	倍長整数	↪	エラーコード

説明

WR Insert picture areaコマンドは、ピクチャ中の4D Writeドキュメントをareaに挿入します。

where は、テキストがどこに挿入されるかを示します。

where には"[WR Parameters](#)"テーマの以下の定数を渡せます:

定数	型	値	コメント
wr at insertion point	倍長整数	0	テキストは現在の挿入位置に挿入される。
wr at end of document	倍長整数	1	テキストはドキュメントの最後に挿入される。

WR Insert picture areaは倍長整数型のエラーコードを返します。

挿入に成功すると0が返されます。[付録C: エラーコード](#)を参照してください。

例題

次の例では、送り主の署名をドキュメントの最後に挿入します。

```
QUERY ([Sender]; [Sender]Name=[Letter]Sender)
ErrorNum:=WR Insert picture area(area;[Sender]Signature_;wr at end of document)
```

WR INSERT RTF EXPRESSION (area ; rtfExpression)

引数	型		説明
area	倍長整数	→	4D Writeエリア
rtfExpression	テキスト	→	RTF式

説明

WR INSERT RTF EXPRESSION コマンドは、*area* に *rtfExpression* に渡されたRTF 式挿入します。式はカーソルが置かれた位置に挿入されます。テキストが挿入される時に選択されていたテキストは、式によって置き換えられます。

4D Write ドキュメントがRTFに書き出される場合、式は生成されたRTFドキュメントに保存されます。

RTF(Rich Text Format)は、異なるワープロソフト間でも、ドキュメント内のサイズ、スタイル、文字の色、マージン等ほとんどのフォーマット属性を保存するファイル フォーマットに変換されます。このフォーマットはRTF に書き出す際に変換される得別なマーカーの使用に基づいています。

WRテキスト操作

 テキスト操作コマンドについて

 WR BACKSPACE

 WR DELETE SELECTION

 WR Direct find

 WR Find

 WR Get font

 WR GET PARAGRAPHS

 WR Get selected text

 WR GET SELECTION

 WR Get styled text

 WR Get text

 WR Get text property

 WR GET WORDS

 WR INSERT STYLED TEXT

 WR INSERT TEXT

 WR Mouse to selection

 WR Replace

 WR SELECT

 WR SET FONT

 WR SET SELECTION

 WR SET TEXT PROPERTY

テキスト操作コマンドについて

このテーマのコマンドや関数を使用してテキストを処理できます。これらのコマンドは4D Writeエリアにテキストを挿入したり、取り出したりするために使用できます。

このテーマのコマンドには標準の検索や置換などが含まれます。

WR BACKSPACE (area)

引数	型		説明
area	倍長整数	⇒	4D Writeエリア

説明

WR BACKSPACE コマンドは、delete キーまたはbackspace キーの押下をシミュレートします。

*area*内で文字が選択されていると、その文字は削除されます。

文字が選択されていない場合、*WR BACKSPACE*はdelete キーまたはbackspace キーの押下と同じ動作を行います。1 回につき1 文字が削除され、挿入ポインタ（カーソル）が1 文字分、左に移動します。このような動作を行わせたくない場合（文字が選択された場合のみ削除させる）は、*WR DELETE SELECTION*を使用します。

WR DELETE SELECTION

WR DELETE SELECTION (area)

引数	型	説明
area	倍長整数 	4D Write エリア

説明

WR DELETE SELECTION コマンドは、*area* で参照される 4D Write エリアから現在選択されているテキストエリアを削除します。

次のステートメントを実行すると、*WR DELETE SELECTION* を使用した場合と同じ結果になります。

WR EXECUTE COMMAND (Area;wr cmd clear)

Note : 定数 *wr cmd clear* の値は 6 です。

現在何も選択されていない場合、カーソルの前にある文字を削除する *WR BACKSPACE* とは異なり、このコマンドは何も行いません。

例題

ドキュメント中のすべてのソフトハイフンを削除します:

```
 `オカレンス数をカウント
HyphenNb:=WR Count(Area;wr_nb_soft_hyphens)
For ($i;1;HyphenNb)
  `先頭のソフトハイフンを選択する
  WR SELECT(Area;9;1)
  `それを削除
  WR DELETE SELECTION(Area)
End for
```

WR Direct find (BLOB ; charString ; wholeWord ; upperCase) -> 戻り値

引数	型	説明
BLOB	BLOB	→ 4D Writeエリアを含むBLOB
charString	文字	→ 検索する文字列
wholeWord	整数	→ 0= 部分一致 1= 完全一致
upperCase	整数	→ 0= 大文字小文字を無視 1= 大文字小文字を区別
戻り値	倍長整数	↻ 検索ステータス

説明

WR Direct findコマンドは任意の4D Write エリアを含むBLOB 内で文字列を直接検索することができます。このコマンドの使用の際にBLOB を4D Writeエリア内であらかじめオープンしておく必要はありません。つまり、このコマンドはとても高速に実行されることを意味しています。

検索文字列が見つかったら、WR Direct find はテキスト内の検索文字列が見つかった位置を返します。

検索が失敗した場合は、WR Direct find は-1 を返します。

blob が4D Write エリアのコンテンツでない場合、WR Direct find は-2 を返します。

wholeWord とupperCase によって、検索オプションを選択できます：

wholeWord にはWR Parametersテーマの以下の定数を渡すことができます：

定数	型	値	コメント
wr partial match	倍長整数	0	文字列は単語全体あるいはより長い単語の一部です。
wr whole word	倍長整数	1	単語は区切り文字 (スペース、句読点等) の間になければなりません。

upperCase にはWR Parametersテーマの以下の定数を渡す ことができます：

定数	型	値	コメント
wr ignore uppercase	倍長整数	0	検索は大文字と小文字を区別せずに行われます。
wr case sensitive	倍長整数	1	検索は大文字と小文字を区別して行われます。

例題

この例題は、レコードの選択範囲内で検索するキーワードからの検索メソッドを示しています。データベースの内容は料理レシピです。4D Write エリアはBLOB フィールドに保存されています。特別な素材を使用したレシピをすべて検索したいものとして、ここに、大変高速なメソッドを示します。

```
ToFind:=Request("Enter the ingredient(s) to find:")
` 検索されたレコードを配置する空のセットを作成
CREATE EMPTY SET([MyRecipes];"FoundRecords")
ALL RECORDS([MyRecipes]) ` 全テーブルの選択範囲をブラウズ
While(Not(End selection([MyRecipes])))
  If(WR Direct find([MyRecipes]BlobRecipe_ ;ToFind;wr whole word;wr case sensitive)>0)

    ADD TO SET([MyRecipes];"FoundRecords")
  End if
NEXT RECORD([MyRecipes])
End while
USE SET("FoundRecords")
OUTPUT FORM([MyRecipes];"Output")
MODIFY SELECTION([MyRecipes];*)
```

WR Find (area ; charString ; wholeWord ; upperCase ; wrap) -> 戻り値

引数	型	説明
area	倍長整数	→ 4D Write エリア
charString	文字	→ 検索する文字列
wholeWord	整数	→ 0= 部分一致 1= 完全一致
upperCase	整数	→ 0= 大文字/小文字の区別なし 1= 大文字/小文字の区別あり
wrap	整数	→ 0= カーソル以降を検索 1= ドキュメント全体を検索
戻り値	倍長整数	↻ 検索ステータス

説明

WR Findコマンドで4D Write エリア内の文字を検索することができます。WR GET WORDSを使うと、検索した単語の位置を取り出すことができます。また、WR GET SELECTIONを使うと、検索され選択された位置を取り出すことができます。検索文字列が見つかったら、WR Findは1を返し、最初に見つかった文字列を選択します。

検索が失敗すると、WR Find は0を返し、現在選択されている部分は修正されません。area が存在しない場合、WR Findは-1を返します。

wholeWord とupperCase によって検索オプションを選択できます。

wholeWord 引数にはWR Parametersテーマの以下の引数を渡すことができます:

定数	型	値	コメント
wr partial match	倍長整数	0	文字列は単語全体あるいはより長い単語の一部です。
wr whole word	倍長整数	1	単語は区切り文字 (スペース、句読点等) の間になければなりません。

upperCase 引数にはWR Parametersテーマの以下の引数を渡すことができます:

定数	型	値	コメント
wr ignore uppercase	倍長整数	0	検索は大文字と小文字を区別せずに行われます。
wr case sensitive	倍長整数	1	検索は大文字と小文字を区別して行われます。

wrapは、検索がドキュメント全体に適用されるかどうかを定義することができます。

この引数にはWR Parametersテーマの以下の引数を渡すことができます:

定数	型	値	コメント
wr after insertion point	倍長整数	0	検索は挿入位置からドキュメントの最後まで行われます。
wr whole document	倍長整数	1	検索は挿入ポイントからドキュメントの終わりまで実行され、さらに先頭から挿入ポイントまで検索されます。

例題 1

ユーザに検索したい文字列を入力させ、検索を行う:

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

例題 2

この例題は、レコードの選択範囲内で検索するキーワードからの検索メソッドを示しています。検索は、ピクチャエリアで実行されます。

重要: 4D Write エリアがBLOBフィールドとして保存された場合は、*WR Direct find*の例題を参照してください。その例題は、より高速なものです。

データベースの内容は料理レシピです。4D Writeエリアはピクチャフィールドに保存されています。特別な素材が使われているレシピをすべて検索したい場合のメソッドを示します:

```
ToFind:=Request("Enter the ingredient(s) to find:")
`検索されたレコードを格納する空のセットを作成
CREATE EMPTY SET([MyRecipes];"FoundRecords")
ALL RECORDS([MyRecipes]) `全テーブルの選択範囲をブラウズ
OffscreenArea:=WR New offscreen area
While(Not(End selection([MyRecipes])))
  WR PICTURE TO AREA(OffscreenArea;[MyRecipes]PictRecipe_)
  If(WR Find(OffscreenArea;ToFind;1;1;1)=1)
    `素材が見つかったら、そのレコードをセットに追加
    ADD TO SET([MyRecipes];"FoundRecords")
  End if
  NEXT RECORD([MyRecipes])
End while
WR DELETE OFFSCREEN AREA(OffscreenArea)
USE SET("FoundRecords")
OUTPUT FORM([MyRecipes];"Output")
MODIFY SELECTION([MyRecipes];*)
```

WR Get font (area ; sameFont) -> 戻り値

引数	型	説明
area	倍長整数	→ 4D Write エリア
sameFont	倍長整数	← 1= 選択範囲内で同じフォントを適用 0= 選択範囲内で異なるフォントを適用
戻り値	文字	↻ 選択された文字列の最後の文字のフォント名

説明

WR Get fontコマンドはareaで参照される4D Writeエリア内で選択されている箇所の最終文字に適用されているフォントの名前を返します。

- sameFont = 1の場合、選択されている全範囲に同じフォントが適用されてます。
- sameFont = 0の場合、選択されている範囲に異なるフォントが適用されてます。

例題

現在の選択範囲のフォントをドキュメント全体に適用します:

```
vFont:=WR Get font(Area;vUniform)
If (vUniform=0) `現在の選択範囲に複数のフォントがある場合
  CONFIRM ("選択範囲内に複数のフォントが存在します。最後の文字に使われたフォントは"
    +vFont+"です。このフォントをドキュメント全体に適用しますか?")
Else
  CONFIRM ("選択範囲のフォントは"+vFont+"です。このフォントをドキュメント全体に適用しますか?")
End if
If (OK=1)
  WR EXECUTE COMMAND(Area;wr_cmd_select_all) `ドキュメント全体を選択
  WR SET FONT(Area;vFont) `新しいフォントを適用
  `ドキュメントの開始位置に挿入ポイントを移動
  WR SET SELECTION(Area;0;0)
  WR SCROLL TO SELECTION(Area) `現在のテキストの選択範囲を表示
End i
```

WR GET PARAGRAPHS (area ; beginPara ; endPara)

引数	型		説明
area	倍長整数	→	4D Write エリア
beginPara	倍長整数	←	返される段落の開始位置
endPara	倍長整数	←	返される段落の終了位置

説明

WR GET PARAGRAPHSコマンドは、areaで参照される4D Write エリア内で選択されている、最初の段落の開始文字位置および最後の段落のキャリッジリターンの位置を返します。

例題

次の例はドキュメントをスキャンし、各段落の開始文字位置と終了文字位置を取り出します。

```

`エリアの先頭にカーソルを置く
WR SET SELECTION(area;0;0)
`ドキュメント内の段落数を数える
nbPara:=WR Count(Zone;wr_nb_paragraphs)
`1 つずつ段落を処理する
For ($i;1;nbPara)
`開始文字位置と終了文字位置を取り出す
  WR GET PARAGRAPHS(area;begin;Pos)
`最後に処理された段落の後にエリアを再配置する
  WR SET SELECTION(area;Pos;Pos)
End for

```

WR Get selected text

WR Get selected text (area) -> 戻り値

引数	型		説明
area	倍長整数	→	4D Write エリア
戻り値	テキスト	↩	エリア内の選択テキスト

説明

WR Get selected text コマンドは area 内の選択テキストを返します。

データベースが非Unicodeモードで実行されている場合、返されるテキストは最初の32,000文字が上限となります。

例題 1

次の例は、vText 変数の中に area 内の選択テキストを代入します。

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

例題 2

非Unicodeモードで実行されているデータベースで、32,000文字以上を選択したかどうかのテスト:

```
C_LONGINT($start;$end)
C_TEXT($text)

WR GET SELECTION(WritePicture;$start;$end) `選択部分の先頭と最後の位置を取得
If($end-$start>=32000) `差が32,000以上の場合、選択部分は切り取られる
    ALERT("Only the first 32,000 characters will be recovered.")
End if
$text:=WR Get selected text(WritePicture)
```

WR GET SELECTION (area ; first ; last)

引数	型		説明
area	倍長整数	⇒	4D Write エリア
first	倍長整数	⇐	先頭文字位置を受け取る
last	倍長整数	⇐	最終文字位置を受け取る

説明

WR GET SELECTIONコマンドはareaで参照される4D Write エリア内の選択テキストの先頭文字位置と最終文字位置をfirstとlast に返します。

first は、常に選択された先頭文字位置よりも1 つ前です。last は、常に選択された最終文字位置と同じです。first とlast が等しい場合、選択されているテキストはありません。また、first の位置にある文字の後ろにカーソルポイントが置かれます。

例題

次の例はドキュメント全体のマージンを設定し、選択範囲を元に戻します:

```
WR GET SELECTION(area;StartSel;EndSel) `現在選択されている部分を読み込む
WR EXECUTE COMMAND(area;wr_cmd_select_all) `すべてを選択
`マージンを設定
WR SET TEXT PROPERTY(area;wr_left_margin;49)
WR SET TEXT PROPERTY(area;wr_first_indent;49)
WR SET TEXT PROPERTY(area;wr_right_margin;504)
WR SET SELECTION(area;StartSel;EndSel) `選択範囲を元に戻す
```

WR Get styled text

WR Get styled text (area) -> 戻り値

引数	型		説明
area	倍長整数	→	4D Write エリア
戻り値	BLOB	↩	フォーマットされたテキスト

説明

WR Get styled text コマンドは、*area* で指定された4D Write エリア内で選択されたテキストを、BLOB フィールドまたは変数に返します。返されたBLOB の構造は、スタイルシートなしで、文字および段落のフォーマットを含んでいます。

WR Get styled text を用いて返されたテキストは、*WR INSERT STYLED TEXT* を使用して他の4D Write ドキュメントに配置することができます。スタイルを付加されたテキストが挿入されても、4D Write ドキュメントのページレイアウトは挿入の影響を受けません。

WR Get styled text と *WR INSERT STYLED TEXT* を使用することにより、クリップボードの代わりにBLOB をバッファとして使用し、コピー&ペースト操作をシミュレートすることができます。

警告： *WR Get styled text* によって返されたBLOBは、4D Write エリアのすべての要素を含んでいるものではないので、*WR BLOB TO AREA* では使用できません。

例題

WR INSERT STYLED TEXT コマンドの例題参照

WR Get text (area ; first ; last) -> 戻り値

引数	型		説明
area	倍長整数	→	4D Write エリア
first	倍長整数	→	テキストの先頭文字位置
last	倍長整数	→	テキストの最終文字位置
戻り値	テキスト	↻	先頭文字と最終文字の間のテキスト

説明

WR Get textコマンドは、*area* 内の*first* の位置にある文字と*last*の位置にある文字の間にあるテキストを返します。4Dのフィールドおよび変数が格納できる最大文字数は2GBです。WR Get textはデータベースがUnicodeモードで実行されている場合、扱える文字の最大値は2GBで、非Unicodeモードの場合は32,000文字です。

もし...

last - first > 32 000でデータベースが非Unicodeモード
last < first
last > *area*の長さ

WR Get text...

空の文字が返され、エラー1024が生成される
空の文字が返され、エラー1013が生成される
*area*に含まれるテキストが返される

WR Get textは*area*で選択されている文字を変更しません。

例題

次の例は、vText 変数の中にエリア内の先頭100 文字を格納します。

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

WR Get text property

WR Get text property (area ; property ; sameProperty) -> 戻り値

引数	型	説明
area	倍長整数	→ 4D Write エリア
property	整数	→ プロパティ番号
sameProperty	整数	← 0= 選択範囲の一部または全体にプロパティがない場合 1= 選択範囲全体にプロパティがある場合
戻り値	実数	↻ プロパティによる

説明

WR Get text propertyコマンドで、*property* で渡されたプロパティが*area*の4D Write エリア内で現在選択されている範囲内で使用されているかどうかを調べることができます。

- *sameProperty* が1 の場合、そのプロパティは選択されている範囲全体に適用されています。
- *sameProperty* が0 の場合、そのプロパティは選択されている範囲内の一部のみに適用されています。
このとき、返される値は選択範囲の最終文字のステータスに対応します。

*property*引数でテストするプロパティを指定します。詳細はWR SET TEXT PROPERTYコマンドの説明を参照してください。

無効なプロパティ番号を渡すと、エラー1075が返されます。

例題 1

マージンが指定値を超えていないことを確かめます:

```
LEFT:=WR Get text property(Area;wr_left_margin;$Uniform)
If (LEFT<3) `左マージンを3に設定
  WR SET TEXT PROPERTY(Area;wr_left_margin;3)
End if
RIGHT:=WR Get text property(Area;wr_right_margin;$Uniform)
If (RIGHT>43) `右マージンを43に設定
  WR SET TEXT PROPERTY(Area;wr_right_margin;43)
End if
```

例題 2

ユーザーにメニューやルーラーにはアクセスさせずに、行間と行揃えを設定させたいものとします。入力フォームに情報とラベル付けされたボタンと、2つの変数 (LineSpacing及びAlignment) を置き、メソッドを設定します。

- 情報ボタンのオブジェクトメソッドで、現在のカーソル位置の情報を取得します:

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

- LineSpacingオブジェクトメソッドは、ユーザが行間を選択した場合に設定を行います:

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

- Alignmentオブジェクトメソッドは、ユーザが行揃えを選択した場合の設定を行います:

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

- On Load フォームイベントにおいて、メニューとルーラーを隠します:

```
If (Form event=On Load)
  WR SET DOC PROPERTY(Area;wr_view_menubar;0)
  WR SET DOC PROPERTY(Area;wr_view_rulers;0)
End if
```

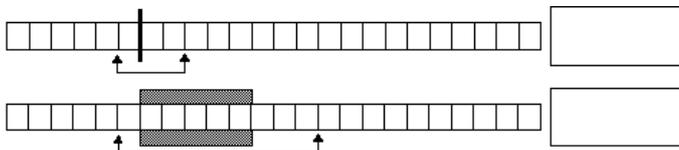
WR GET WORDS (area ; beginSel ; endSel ; smartCutPaste)

引数	型	説明
area	倍長整数	⇒ 4D Write エリア
beginSel	倍長整数	← 返される単語の開始位置
endSel	倍長整数	← 返される単語の終了位置
smartCutPaste	整数	← 0= 最終文字が空白でない場合 1= 最終文字が空白の場合

説明

WR GET WORDSコマンドは選択範囲内の最初の単語の開始文字位置と最後の単語の終了文字位置を返します。また、選択範囲の終了文字が空白かどうかを示します。何もテキストが選択されていない場合、beginSel とendSel はカーソルが置かれている単語の開始文字位置と終了文字位置を返します。

このコマンドは、現在選択されている部分に何の影響も与えません。



選択された範囲が単語の途中（または単語の最終文字の次に来る文字が空白である場合）から開始されている場合、beginSel はその単語の開始文字位置を返します。

選択範囲が単語の途中で終わっている場合は、次の2つのケースが考えられます：

- 単語の次に来る文字が空白の場合、endSel は空白を含んだ位置を返し、smartCutPaste は1を返します。
- 単語の次に来る文字が空白でない場合、endSel はその単語の終了文字位置を返し、smartCutPaste は0を返します。

例題

次の例はドキュメントをスキャンし、各単語の開始文字位置と終了文字位置を取り出します。

```

`エリアの先頭にカーソルを置く
WR SET SELECTION(area;0;0)
`ドキュメント内の単語数を数える
nbWords:=WR Count(area;wr_nb_words)
`1語ずつ単語を処理する
For($i;1;nbWords)
  `開始文字位置と終了文字位置を取り出す
  WR GET WORDS(area;beginning;pos)
  `最後に処理された単語の後にエリアを再配置する
  WR SET SELECTION(area;Pos;Pos)
End for

```

WR INSERT STYLED TEXT (area ; BLOB)

引数	型		説明
area	倍長整数	→	4D Write エリア
BLOB	BLOB	→	変数またはフィールド

説明

WR INSERT STYLED TEXT コマンドは、*area* で参照される4D Write エリアの中に*blob*の内容を挿入します。この挿入処理はカーソルの位置または*blob*の内容と現在選択されている部分を置き換えるかのどちらかです。*blob* は、BLOB 変数または BLOB フィールドのどちらかです。ただし*blob* が*WR Get styled text* を使用して最初に作成されていることが必須条件となります。

blob 内で書式化されたテキストを表すために用いられる内部フォーマットは、プラットフォームに依存しません。そのため、Mac OS上で作成した*blob* コンテンツを、後でWindows ドキュメントの中に挿入することができます。また、その逆も可能です。

*blob*はスタイルシートを除く4D Writeエリアの属性(カラーやスタイル等)付きテキストおよび段落属性(マージン、タブ、フォーマット)を格納しています。

例題

この例は、最も頻繁に用いられるビジネスレターのテンプレートをテーブル[Letters]の中に格納し、ハードディスクの空きに保存したい場合を示します。'Templates'という名前のBLOB フィールドをテーブル中に作成します。テーブルの入力フィールドには、'Area'という名前の4D Write エリアを挿入します。つまり、以下のメソッドをフォームに埋め込みます：

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

WR INSERT TEXT

WR INSERT TEXT (area ; text)

引数	型		説明
area	倍長整数	⇒	4D Write エリア
text	文字	⇒	挿入されるテキスト

説明

WR INSERT TEXT コマンドは *text* を *area* に挿入します。文字が選択されている場合、それが置き換えられます。文字が選択されていない場合、*text* はカーソルポイントに置かれます。このコマンドは、自動参照を行いたくない場合に *WR INSERT EXPRESSION* や *WR INSERT FIELD* の代わりに使用されます。

例題

次の例は *area* に *vText* 変数内のテキストを挿入します。

```
WR INSERT TEXT(Area;vText)
```

WR Mouse to selection

WR Mouse to selection (area ; posHoriz ; posVert ; beginSel ; endSel) -> 戻り値

引数	型		説明
area	倍長整数	→	4D Write エリア
posHoriz	整数	→	エリア内のマウスの横位置
posVert	整数	→	エリア内のマウスの縦位置
beginSel	倍長整数	←	選択範囲の開始位置
endSel	倍長整数	←	選択範囲の終了位置
戻り値	整数	↻	カーソル位置に一致する選択範囲

説明

WR Mouse to selection コマンドは、カーソル位置と一致するセレクションを返します。カーソル位置がテキストならば0を、カーソル位置がピクチャならば1を返します。

WR Mouse to selection は、マウスが解放されてオブジェクトがペーストされたときのカーソルと位置を知るために、ドラッグとドロップマネージャーとともに使用します。

参照上でマウスボタンを解放したとき、*beginSel* と *endSel* には特定の値が返されます。

警告: この場合リファレンスに含まれる文字数にかかわらず、 $endSel = beginSel + 1$ を返します。つまり、リファレンス=1文字です。

posHoriz と *posVert* はデフォルトとして0000を返します。これらの値を返させるためには、4D コマンドの **GET MOUSE** を事前使用する必要があります。詳細は、このコマンドのドキュメントを参照してください。

WR Replace (area ; searchedFor ; replaceWith ; wholeWord ; upperCase ; replaceAll ; wrap) -> 戻り値

引数	型	説明
area	倍長整数	→ 4D Write エリア
searchedFor	文字	→ 検索する文字列
replaceWith	文字	→ 置き換え文字列
wholeWord	整数	→ 0=部分一致、1=完全一致
upperCase	整数	→ 0=大文字小文字を区別しない、1=大文字小文字を区別する
replaceAll	整数	→ 0=次を置換、1=すべて置換
wrap	整数	→ 0=選択位置以降を検索、1=ドキュメント全体を検索
戻り値	倍長整数	→ 置換されたオカレンス数

説明

WR Replaceコマンドは編集メニューの置換コマンドと同じ働きをします。

wholeWord引数にはWR Parametersテーマの以下の定数を渡します:

定数	型	値	コメント
wr partial match	倍長整数	0	文字列は単語全体あるいはより長い単語の一部です。
wr whole word	倍長整数	1	単語は区切り文字 (スペース、句読点等) の間になければなりません。

upperCase引数にはWR Parametersテーマの以下の定数を渡します:

定数	型	値	コメント
wr ignore uppercase	倍長整数	0	検索は大文字と小文字を区別せずに行われます。
wr case sensitive	倍長整数	1	検索は大文字と小文字を区別して行われます。

replaceAll引数にはWR Parametersテーマの以下の定数を渡します:

定数	型	値	コメント
wr replace next	倍長整数	0	単語の次のオカレンスのみが置換されます。
wr replace all	倍長整数	1	単語のすべてのオカレンスが置換されます。

wrap引数にはWR Parametersテーマの以下の定数を渡します:

定数	型	値	コメント
wr after insertion point	倍長整数	0	検索は挿入位置からドキュメントの最後まで行われます。
wr whole document	倍長整数	1	検索は挿入ポイントからドキュメントの終わりまで実行され、さらに先頭から挿入ポイントまで検索されます。

WR Replaceは置換したオカレンス数を返します。

例題

ドキュメント内の不必要な連続した2つのスペース文字を取り除きます:

```

`連続した2つのスペース文字を含む変数
ToFind:=" "
`オカレンスが見つかる間
While (WR Find(Area;ToFind;wr partial match;wr ignore uppercase;wr whole document)=1)
`連続した2つのスペース文字を1つのスペース文字に置き換える
  $n:=WR Replace(Area;ToFind;" ";wr partial match;wr ignore uppercase;wr replace all;wr whole
document)
End while

```

WR SELECT (area ; type ; begin ; end)

引数	型		説明
area	倍長整数	→	4D Write エリア
type	整数	→	選択タイプ
begin	倍長整数	→	開始文字位置
end	倍長整数	→	終了文字位置、typeによっては省略可能

説明

WR SELECT コマンドは *type*、*begin*、*end* によって設定されたテキストを選択します。*WR SELECT* は検索する値が存在しない場合、現在の選択を変更しません。

type 引数は **WR Select type** テーマの以下の定数から指定します:

定数	型	値	コメント
wr select characters	倍長整数	0	Selects the characters located between <i>begin</i> と <i>end</i> の間に位置する文字を選択。この場合 <i>WR SET SELECTION</i> を使用するのと同じ。
wr select expression	倍長整数	1	ドキュメント中で <i>begin</i> により指定されるランクの参照を選択。 <i>end</i> は省略しなければなりません。
wr select paragraphs	倍長整数	2	<i>begin</i> と <i>end</i> の間に位置する段落を選択します。
wr select ruler	倍長整数	3	(ドキュメント中でドキュメントの先頭から始まるランクの) X番目のルーラを使用する段落を選択。 <i>end</i> は省略しなければなりません。
wr select picture	倍長整数	4	ドキュメント中で <i>begin</i> により指定されるランクのピクチャを選択。 <i>end</i> は省略しなければなりません。
wr select style	倍長整数	5	(ドキュメント中でドキュメントの先頭から始まるランクの) X番目のスタイルを使用する単語を選択。 <i>end</i> は省略しなければなりません。
wr select word	倍長整数	6	挿入ポイントの位置の単語を選択。
wr select page break	倍長整数	7	ドキュメント中で <i>begin</i> により指定されるランクの改ページを選択。 <i>end</i> は省略しなければなりません。
wr select column break	倍長整数	8	ドキュメント中で <i>begin</i> で指定されるランクである段組みブレイクを選択。 <i>end</i> は省略しなければなりません。
wr select hyphen	倍長整数	9	ドキュメント中で <i>begin</i> により指定されるランクのハイフンを選択。 <i>end</i> は省略しなければなりません。
wr select page number	倍長整数	10	ドキュメント中で <i>begin</i> により指定されるランクのページ番号を選択。 <i>end</i> は省略しなければなりません。選択はテキストのボディに挿入されるページ番号にのみ持ち越されます。
wr select date and time	倍長整数	11	ドキュメント中で <i>begin</i> により指定されるランクの日付および時間変数を選択。 <i>end</i> は省略しなければなりません。選択は自動でボディテキストに更新・挿入される日付と時刻にのみ繰り越されます。
wr select hyperlink	倍長整数	12	ドキュメント中で <i>begin</i> により指定されるランクのハイパーリンクを選択。 <i>end</i> は省略しなければなりません。
wr select HTML expression	倍長整数	13	ドキュメント中で <i>begin</i> により指定されるランクのHTML式を選択。 <i>end</i> は省略しなければなりません。
wr select	倍		

wr select RTF expression	長 整 数	14	ドキュメント中でbeginにより指定されるランクのRTF式を選択。endは省略しなければなりません。
--------------------------------	-------------	----	--

例題 1

次の例は、あるページブレークの有無によって異なる関数を実行します：

```
  `選択範囲の設定
WR SET SELECTION(area;0;0)
  `最初のページブレークを選択
WR SELECT(area;wr_select_page_break;1)
  `新しい選択範囲を取り出す
WR GET SELECTION(area;$v1begin;$v1lend)
If (($v1begin=0) & ($v1lend=0))
  `ページブレークなし
Else
  `ページブレークを使用して何らかの処理を行う
End if
```

例題 2

次の例はareaで参照される4D Write エリア内のリファレンスを選択し、その参照を見つけやすいようにスタイルを適用します：

```
NbObjects:=WR Count(area;4)
  `参照数をカウントする
For (i;1;NbObjects)
  WR SELECT(area;wr_select_expression;i)
  `各参照を選択する
  WR GET REFERENCE(area;TableNo;FieldNo;vName;vType)
  WR SET TEXT PROPERTY(area;wr_bold;1)
  WR SET TEXT PROPERTY(area;wr_text_color;wr_blue)
  `選択範囲に青色とボールドを適用する
End for
```

WR SET FONT (area ; font)

引数	型		説明
area	倍長整数	⇒	4D Write エリア
font	文字	⇒	フォントの名前

説明

WR SET FONT コマンドを使用して、*area*によって参照される4D Write エリア内の現在選択されている範囲にフォントを指定することができます。

font には、使用したいフォントの名前を渡します。*font*がインストールされていない場合、システムエラー1077が返されます。

例題

[WR Get fontの例題参照](#)

WR SET SELECTION (area ; first ; last)

引数	型		説明
area	倍長整数	⇒	4D Write エリア
first	倍長整数	⇒	先頭文字位置
last	倍長整数	⇒	最終文字位置

説明

WR SET SELECTION コマンドは、*area* 内の *first* と *last* で指定されるテキストを選択します。選択されるテキストは、*first* に 1 プラスした文字から *last* 位置にある文字までです。

first と *last* が同じ場合、*WR SET SELECTION* は *first* の位置にある文字の後ろにカーソルポイントを置きます。*last* の値がエリア内のテキストの長さよりも大きいと、*WR SET SELECTION* はそのドキュメントの終わりまでテキストを選択します。*last* が *first* より小さい場合、*WR SET SELECTION* は何も行いません。

例題 1

次の例は、vText 変数の中にエリア内の先頭から 10 文字を選択します：

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

例題 2

テキストの最後に挿入ポイントを置きます：

```
WR SET SELECTION(area;10000000;10000000)
```

WR SET TEXT PROPERTY (area ; property ; value)

引数	型		説明
area	倍長整数	⇒	4D Write エリア
property	整数	⇒	設定するプロパティ番号
value	倍長整数	⇒	選択されたプロパティの値

説明

*WR SET TEXT PROPERTY*コマンドを使用して、*area*で参照される4D Write エリア内で現在選択されている範囲のテキストプロパティを修正できます。

*property*と*value*はペアで使用します。

Tip: *WR SET TEXT PROPERTY* (Area;*wr font number*;Value)ではなく*WR SET FONT*コマンドの利用を推奨します。フォント番号の管理は動的であり、OSやマシンごとに異なることがあります。

無効なプロパティ番号を渡すとエラー-1075が生成されます。

選択したプロパティに対して無効な値を渡すとエラー-1076が生成されます。

Notes:

- *property*と*value*は定数を使用して指定できます。テキストプロパティのリストおよびその値のリストは**WR Text properties**と**WR Text properties values**定数テーマにあります。
- エラーコードのリストは**付録C: エラーコード**を参照してください。

以下の定数と値を*WR SET TEXT PROPERTY*と*WR Get text property*コマンドで使用できます:

property (WR Text properties)	設定または取得で使用 (valueまたはWR Text properties values)
<u>wr bold</u> (0)	テキストのボールドスタイル (false=0, true=1)
<u>wr italic</u> (1)	テキストのイタリックスタイル (false=0, true=1)
<u>wr shadow</u> (2)	テキストの影付きスタイル (false=0, true=1)
<u>wr strikethrough</u> (3)	テキストの取り消し線スタイル (false=0, true=1)
<u>wr underline</u> (4)	テキストの下線スタイル: 下線なし=0, <u>wr single underline</u> (1), <u>wr word underline</u> (2), <u>wr double underline</u> (3), <u>wr hatched underline</u> (4)
<u>wr superscript or subscript</u> (5)	上付きあるいは下付きテキスト: なし=0, <u>wr superscript</u> (1), <u>wr subscript</u> (2)
<u>wr capital case</u> (6)	小型大文字、大文字、または小文字: 小文字=0, <u>wr capitals</u> (1), <u>wr small capitals</u> (2)
<u>wr font number</u> (7)	渡される番号は内部的なものです。4D Writeは使用されるごとに番号を割り当てます。フォント名を使用する WR Get font と WR SET FONT コマンドの利用をお勧めします。
<u>wr font size</u> (8)	テキストのサイズ (9から255)
<u>wr text color</u> (9)	0x00RRGGBB形式の値
<u>wr text back color</u> (10)	4Dと同様 (または以前のバージョンの4D Write).
<u>wr strikethrough color</u> (11)	WR Standard colors テーマの定数を使用できます。 .
<u>wr underline color</u> (12)	
<u>wr shadow color</u> (13)	
<u>wr links appearance</u> (14)	リンクのアピアランス: <u>wr no links appearance</u> (0), <u>wr unvisited links appearance</u> (1), <u>wr visited links appearance</u> (2)
<u>wr stylesheet number</u> (15)	リスト中のスタイルシートのインデックスを渡します。スタイルシートインデックスを渡した場合、テキストにスタイルシートが割り当てられますが、スタイルシートのプロパティは適用されません。 WR APPLY STYLESHEET コマンドはスタイルシートのプロパティを設定し、それを適用します。
<u>wr user property</u> (16)	値は自由に設定できます。このプロパティのカスタマイズした値を設定および取得できます。例えば階層リストとテキストを同期させたいときに、このプロパティを使用して階層リストの要素参照を格納できます。テキストをクリックするごとに、プロパティを取得して階層リスト中の対応する要素を選択できます。
<u>wr justification</u> (32)	テキスト揃え: <u>wr left justified</u> (0), <u>wr centered</u> (1), <u>wr right justified</u> (2), <u>wr full justified</u> (3)
<u>wr line spacing</u> (33)	行間、値は1から10まで0.5毎に設定: 1= 1行, 1.5=1.5行, 2=2行
<u>wr bullet</u> (34)	箇条書きスタイル: <u>wr black square bullet</u> (110), <u>wr white square bullet</u> (111), <u>wr black circle bullet</u> (108), <u>wr white circle bullet</u> (109), <u>wr diamonds bullet</u> (117), <u>wr clubs bullet</u> (118), <u>wr no bullet</u> (0)
<u>wr left margin</u> (35)	左デッドマージンを考慮した距離。ドキュメントのカレントの単位を使用。
<u>wr first indent</u> (36)	右マージンを考慮した距離。 <0 = 左への右マージン, >0 = 右への右マージン。ドキュメントのカレントの単位を使用。
<u>wr right margin</u> (37)	右デッドマージンを考慮した距離。ドキュメントのカレントの単位を使用。

wr border	
back color (38)	0x00RRGGBB形式の値を使用
wr border line color (39)	4Dと同様 (または以前のバージョンの4D Write)。 WR Standard colors テーマの定数を使用できます。 .
wr border line style (40)	枠線のスタイルとサイズ: wr 1 pt line (0), wr 2 pts line (1), wr 3 pts line (2), wr dotted line (3), wr double dotted line (4), wr triple dotted line (5), wr double 1 pt line (6), wr double inside 2 pts line (7), wr triple center 2 pts line (8), wr double outside 2 pts line (9), wr half pt line (10), wr quarter pt line (11)。 枠線スタイルの設定を行うと、選択部の枠線に直接効果があります。 または事前に枠線のタイプを設定してから配置することもできます。 まず枠線タイプを設定し、後に枠線を配置することをお勧めします。 そのようにすれば再描画を避けられます。 枠線スタイルはすべての側 (上下と左右) で同じであることに留意してください。
wr left border (41)	枠線の設定 (false=0, true=1)
wr right border (42)	枠線の設定 (false=0, true=1)
wr inside top border (43)	枠線の内側の設定 (false=0, true=1)。 段落の上下にスペースが追加されます。
wr inside bottom border (44)	枠線の内側の設定 (false=0, true=1)。 段落の上下にスペースが追加されます。
wr border spacing (45)	枠線とテキストの間隔。 ドキュメントのカレントの単位を使用。
wr top border (46)	枠線の設定 (false=0, true=1)。 段落の上にスペースが追加されます。
wr bottom border (47)	枠線の設定 (false=0, true=1)。 段落の下にスペースが追加されます。
wr tab (64)	選択された最後の段落のタブ数。 プロパティは <i>WR SET TEXT PROPERTY</i> では使用できません — <i>WR Get text property</i> で使用します。

例題 1

選択部に以下のプロパティを適用します: Times フォント, 12 ポイント, スミレ色, イタリックなし, ボールド

```
Violet:=WR RGB to color(61952;2048;33792)
WR SET FONT(Area;"Times")
WR SET TEXT PROPERTY(Area;wr_font_size;12)
WR SET TEXT PROPERTY(Area;wr_text_color;wr violet)
WR SET TEXT PROPERTY(Area;wr_bold;1)
WR SET TEXT PROPERTY(Area;wr_italic;0)
```

例題 2

マージンを定義済みの値に設定します:

```
WR GET SELECTION(Area;StartSel;EndSel) `現在の選択テキストを取得
WR UPDATE MODE(Area;0) `スクリーンの描画を無効に
WR EXECUTE COMMAND(Area;wr_cmd_select_all) `すべて選択
`ドキュメントの単位をcmに設定
WR SET DOC PROPERTY(Area;wr_unit;0)
`ドキュメントのマージンを設定
WR SET TEXT PROPERTY(Area;wr_right_margin;1,8)
WR SET TEXT PROPERTY(Area;wr_left_margin;1,3)
WR SET SELECTION(Area;StartSel;EndSel) `選択を復元
WR UPDATE MODE(Area;1) `スクリーン描画を有効にする
```

WRドキュメント

 ドキュメントコマンドについて

 WR GET DOCUMENT INFO

 WR LOCK DOCUMENT

 WR OPEN DOCUMENT

 WR SAVE DOCUMENT

 WR SET DOCUMENT INFO

ドキュメントコマンドについて

このテーマのコマンドや関数を使用してディスクに保存された4D Writeドキュメントを操作できます。

これらのコマンドを使用することで、4D Writeドキュメントをメソッドで保存したり開いたり、ロックしたりできます。

また件名や著者などの情報を設定したり取得したりもできます。

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

引数	型		説明
area	倍長整数	→	4D Write エリア
string	文字	←	ドキュメントのタイトル
subject	文字	←	ドキュメントの件名
author	文字	←	ドキュメントの著者
company	文字	←	会社名
notes	文字	←	ドキュメントの注記
createDate	日付	←	作成した日
createTime	時間	←	作成した時刻
modifDate	日付	←	修正した日
modifTime	時間	←	修正した時刻
lock	整数	←	0= ロックされていない 1= ロックされている

説明

WR GET DOCUMENT INFOコマンドを使うことで、ドキュメント情報ダイアログに表示されるドキュメント情報を取得することができます。ドキュメント情報ダイアログはツールメニューから**ドキュメント情報**を選択することで表示されます。

ドキュメントの件名、著者名、会社名、注記などいくつかのドキュメント情報は、WR SET DOCUMENT INFOを使用して設定することができます。

lock は、WR LOCK DOCUMENTで設定できます。これは、ユーザによるドキュメントの変更を防止する論理的なロックです。ロックはペーストやカット、テキスト入力、修正または属性の変更といったユーザによる操作に影響を及ぼします。ドキュメントの読み取りやテキストのコピー、文字の検索、ドキュメントの印刷といった操作は行うことができます。

createDate、createTime、modifDate、modifTime はドキュメントが保存された際に、4D Write によって自動的に更新されます。

例題

WR SET DOCUMENT INFOコマンドの例題参照

WR LOCK DOCUMENT (area ; status)

引数	型		説明
area	倍長整数	→	4D Write エリア
status	整数	→	0= ロックされていない 1= ロックされている

説明

WR LOCK DOCUMENT コマンドは、*area*によって参照される4D Write エリアのユーザによる変更を禁止します。ドキュメントが一旦ロックされると、ユーザはテキストのペースト、カット、更新ができなくなります。テキストのスクロール、コピー、検索、印刷は引き続き可能です。

現在のドキュメントのロック状態を知るには、*WR GET DOCUMENT INFO*を使用します。この情報は**ツールメニューからドキュメント情報**を選択することで表示されるドキュメント情報ダイアログでも表示されます。

*status*引数には、**WR Parameters**テーマの以下の定数を渡すことができます：

定数	型	値	コメント
wr unlocked document	倍長整数	0	ドキュメントはアンロックされます。
wr locked document	倍長整数	1	ドキュメントはロックされます。

例題

最終的なレコードを閉じ、その後ユーザが編集するのを禁止したいものとします。

```

`ドキュメントの編集は不可
WR LOCK DOCUMENT(Area;wr_locked_document)
`ユーザーがメニューコマンドツール>ドキュメント情報を使用してダイアログを開き、
`オプションを使用することができないようにする
WR LOCK COMMAND(Area;wr_cmd_doc_information;wr_locked_command)

```

WR OPEN DOCUMENT (area ; document ; type)

引数	型	説明
area	倍長整数	→ 4D Write エリア
document	文字	→ 開くドキュメントのパス
		← 開いたドキュメントのパス
type	文字	→ 開くドキュメントのタイプ (4 文字)
		← 開かれたドキュメントのタイプ (4 文字)

説明

WR OPEN DOCUMENT コマンドは、*document* で指定されたドキュメントを開き、*area* で参照された 4D Write エリアに配置します。

document はドキュメントファイルの名前または完全アクセスパスです。

例:

- Windows ではディレクトリ区切り文字として “¥” を使用します。
“D:¥directory1¥directory2¥file.4W7”.
 - Mac OS ではディレクトリ区切り文字として “:” を使用します。
“MacintoshHD:Folder:Document”.
- ドキュメント名に拡張子が指定されていない場合 (Macintosh ドキュメント)、4D Write は最適なタイプで開こうとします。

document がファイル名のみの場合、**WR OPEN DOCUMENT** はデータベースストラクチャファイルと同階層でファイルを探します。

document が空文字の場合、**WR OPEN DOCUMENT** は標準のファイルを開くダイアログボックスを表示します。ファイルを開くダイアログボックスの開くボタンがクリックされると、OK システム変数に 1 が設定され、*document* 変数にユーザーが選択したドキュメントの完全パス名が代入されます。

キャンセルボタンがクリックされると、*document* 変数は空文字となり、OK システム変数に 0 が設定されます。

オプションの *type* 引数を使用して標準のファイルを開くダイアログボックスにデフォルトで表示されるドキュメントのタイプをフィルターできます (HTML ドキュメントを除く)。HTML ドキュメントの場合、*type* 引数は HTML ソースコード (*type*="TEXT" の場合) または HTML ページ (*type* が "HTM3"、"HTML" を含むが省略された場合) のいずれかを表示するために使用されます。(4D Write は HTML 3 のみをサポートしていることに留意してください。)

type は **WR Document types** テーマの定数を使用して指定できます:

定数	型	値	コメント
wr 4D Write document	文字列	4WR7	4D Write カレントバージョンフォーマットのドキュメント
wr 4D Write template	文字列	4WT7	4D Write テンプレートフォーマットドキュメント
wr HTML 3 document	文字列	HTM3	HTML 3.0 フォーマットテキスト
wr Macintosh text document	文字列	ASCM	Mac OS フォーマットテキスト
wr RTF document	文字列	RTF	RTF フォーマットテキスト
wr unicode document UTF16	文字列	ASCU	Unicode 16-byte フォーマットテキスト
wr unicode document UTF8	文字列	ASC8	Unicode 8-byte フォーマットテキスト
wr Windows text document	文字列	ASCW	Windows フォーマットテキスト

互換性に関する注意: 以前のバージョンとの互換性を保つため、**4WR6** (4D Write 6.0 ドキュメント) と **DOC6** (Word 6 ドキュメント) タイプもサポートされています。

すべてのケースで、コマンド実行後、*type* に開かれたドキュメントの実際のタイプが代入されます。

例題

以下の例はデータベースディレクトリにあるファイルを開きます:

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

WR SAVE DOCUMENT (area ; document ; type)

引数	型	説明
area	倍長整数	→ 4D Write エリア
document	文字	→ 作成するドキュメントのパス名
		← 作成されたドキュメントのパス名
type	文字	→ 作成するドキュメントのタイプ (4文字)
		← 作成されたドキュメントのタイプ (4文字)

説明

WR SAVE DOCUMENTコマンドは、*document*に指定されたアクセスパスを使用して、*area*で指定される4D Write エリアドキュメントを保存します。

document はドキュメント名またはパス名です。Windows 上では、ファイルの拡張子を含める必要があります。

例:

- Windowsまたはクロスプラットフォームにする場合、ディレクトリ区切り文字に“¥”を使用します:
"D:¥directory1¥directory2¥file.4W7"
- Mac OSではディレクトリ区切り文字に“:”を使用します: "MacintoshHD:Folder:Document"

*document*にファイル名のみが指定されている場合、WR SAVE DOCUMENTはデータベースストラクチャと同階層にドキュメントを保存します。

*document*が空文字の場合、WR SAVE DOCUMENTは標準のファイル保存ダイアログボックスを表示します。

ファイルを保存ダイアログボックスで**保存** (Mac OS) あるいは **OK** (Windows) ボタンをクリックされると、OKシステム変数に1が設定され、*document*変数にユーザーが選択したファイルの完全パス名が代入されます。

この場合、*type*引数にはユーザーがタイプドロップダウンリストで選択したタイプ、あるいはユーザーがタイプを選択しなかった場合はドキュメントタイプが返されます。

ユーザーが**キャンセル**ボタンをクリックすると、*document*には空文字が返され、OKシステム変数に0が設定されます。

ファイルフォーマットはファイルを保存ダイアログボックス中の、タイプドロップダウンリスト (Windows) あるいはタイプポップアップメニュー (MacOS) で選択できます。

デフォルトでドキュメントは4D Writeフォーマットで保存されます。異なるタイプを指定したい場合、そのタイプを*type*引数に渡します。タイプは4文字で構成されます。**WR Document types**テーマの以下の定数を使用できます:

定数	型	値	コメント
wr 4D Write document	文字列	4WR7	4D Writeカレントバージョンフォーマットのドキュメント
wr 4D Write template	文字列	4WT7	4D Writeテンプレートフォーマットドキュメント
wr HTML 3 document	文字列	HTM3	HTML 3.0フォーマットテキスト
wr HTML 4 document	文字列	HTML	HTML 4.0フォーマットテキスト
wr Macintosh text document	文字列	ASCM	Mac OSフォーマットテキスト
wr RTF document	文字列	RTF	RTFフォーマットテキスト
wr unicode document UTF16	文字列	ASCU	Unicode 16-byteフォーマットテキスト
wr unicode document UTF8	文字列	ASC8	Unicode 8-byteフォーマットテキスト
wr Windows text document	文字列	ASCW	Windowsフォーマットテキスト

Notes:

- "RTF"の場合、4文字にするために最後にスペースを追加します。
- 4D Write中でHTMLとしてドキュメントを表示したい場合は、HTML 3.2書き出しを使用します (4D WriteはHTML 3の読み込みのみをサポートしています)。

互換性に関する注意: 以前のバージョンとの互換性を保持するため、**DOC8** (Word 8ドキュメント) タイプもサポートされています。

*type*引数はドキュメントのエンコーディングにのみ使用されます。Mac OSのファイルタイプにもWindowsの拡張子にも対応しません。

しかし、この引数は4D Writeが適切なWindowsのファイル拡張子、あるいはMac OSのファイルクリエーター/タイプを判断

するために使用されます:

- **Windows**

4D Writeフォーマット	拡張子
4D Write ドキュメント	.4W7
4D Write テンプレート	.4WT
RTF	.RTF
HTML 3.2または4	.HTM
ASCII PC/Mac	.TXT
ASCII unicode 8または16バイト	.TXT
Word	.DOC

名前に拡張子が付けられている場合でも、ファイル拡張子はtype引数の値に基づいて決定されます。例えばdocumentに"Report.RTF"を渡しても、typeに"HTML"を指定すれば、ファイル名は"Report.HTM"となります。

- **Mac OS**

4D Write フォーマット	クリエーター	タイプ
4D Write ドキュメント	4DW7	4WR7
4D Writeテンプレート	4DW7	4WT7
RTF	4DW7	RTF
HTML 3.2または4	MOSS	TEXT
ASCII PC/Mac	4DW7	TEXT
ASCII unicode 8または16 bytes	4DW7	TEXT
Word	MSWD	W8BN

例題 1

4D Write ファイルタイプの'LetterClient'というファイル名でドキュメントを保存したい場合の例を示します。このドキュメントはデータベースストラクチャファイルと同じ階層に位置する"WriteDocuments"フォルダに保存されます:

```
`データベースストラクチャファイルのフルパス名を取得
$Doc:=Structure file
Doc:=$Doc
$Long:=0
`フルパス名からストラクチャ名を取り除くために最後の区切り文字の位置を取得
While ((Position(":";$Doc)#0)
  $Doc:=Substring($Doc;1+Position(":";$Doc);Length($Doc))
  $Long:=Length($Doc)
End while
`ドキュメントのフルパス名を作成するために名前を連結する
`ドキュメント名に拡張子を追加しクロスプラットフォームで使用できるようにする
Doc:=Substring(Doc;1;Length(Doc)-$Long)+"WriteDocuments:LetterClient.4W7"
WR SAVE DOCUMENT(Area;doc;wr 4D Write document)
```

例題 2

保存するドキュメントの名前とタイプをユーザーに選択させ、その値を取得します:

```
DocName:=""
DocType:=""
WR SAVE DOCUMENT(Area;DocName;DocType)
If (OK=1)
  ... `DocNameとDocTypeの値を使用
End if
```

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

引数	型		説明
area	倍長整数	⇒	4D Write エリア
title	文字	⇒	ドキュメントのタイトル
subject	文字	⇒	ドキュメント件名
author	文字	⇒	ドキュメントの著者
company	文字	⇒	会社名
comment	テキスト	⇒	コメント

説明

WR SET DOCUMENT INFOコマンドは、引数に渡された情報をドキュメントに保存します。ユーザはこれらの情報をドキュメント情報ダイアログで参照することができます。このダイアログは**ツールメニュー**から**ドキュメント情報**を選択することで表示できます。

ドキュメントのロック状態を管理するには、[WR LOCK DOCUMENT](#)コマンドの説明を参照してください。

例題

ドキュメント情報でユーザが編集できる箇所をタイトル、件名、注記に限定したい場合、メニュー項目のセレクションに割り込むメソッドを実行し、ユーザが**ツールメニュー**から**ドキュメント情報**を選択した時にカスタマイズされたフォームを表示します。

1. 4D Write エリアを含むフォームのフォームメソッドで、メニュー項目に割り込むためのコード:

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

2. z65OnCmd メソッドは以下の通り:

```
C_LONGINT ($1; $2; $3)
Case of
  : ($2=wr_cmd_doc_information) `ユーザーがツール/ドキュメント情報を選択したら=801
    DIALOG ([TheTable]; "InfoArea") `カスタマイズした情報フォーム
Else
  WR EXECUTE COMMAND ($1; $2) `他のメニューコマンドが選択された場合
End case
```

3. "InfoArea"という名前のカスタマイズされた情報フォームに、変数vTitle、vSubject、vCommentsだけを編集可能エリアとして配置します。以下はこのフォームのメソッドです:

```
Case of
  : (Form event=On_Load)
    WR GET DOCUMENT INFO (WArea; vTitle; vSubject; vAuthor; vCy; vComments; DCreat; HCreat; DModif;
    HModif; Lock)
    `必要であれば空要素を代入
    If (vCy="")
      vCy:="A.C.I."
      vAuthor:=Current user
      vCreation:=String (DCreat)+" at "+Time string (HCreat)
      vModification:=String (DModif)+" at "+Time string (HModif)
    End if
  : (Form event=On_Unload) `フォームが閉じられるとき
    WR SET DOCUMENT INFO (WArea; vTitle; vSubject; vAuthor; vCy; vComments)
End case
```

WRドラッグアンドドロップ

 ドラッグアンドドロップコマンドについて

 WR GET DRAG SOURCE

 WR GET DROP INFO

 WR GET DROP TARGET

🌿 ドラッグアンドドロップコマンドについて

4D Writeでは同じ4D Writeエリア内、異なる2つの4D Writeエリア間、また4D Writeと4D間でドラッグ&ドロップ操作を行います。

ドラッグ&ドロップはデフォルト (標準モード) またはプログラムで使用できます。

デフォルトドラッグ&ドロップ

デフォルトで4D Writeは、選択されたテキストやピクチャーをマウスで動かす、移動やコピーを行う標準の自動ドラッグ&ドロップを提供します。

ドラッグ&ドロップを使用してピクチャーが4D Writeエリアに挿入されると、自動でテキスト中にペーストされます。データは同じあるいは2つの4D Writeエリア間でドラッグ&ドロップされると、ドラッグ元のエリアから削除され、移動されます。データをコピーしたい場合、**Ctrl** (Windows) や **Command** (Mac OS) を操作の間押したままにします。

このタイプのドラッグ&ドロップでは特別なプログラミングを必要としません。4Dフォーム内でドラッグ&ドロップを行う場合は単に“ドラッグ可”や“ドロップ可”プロパティを設定するだけです (後述)。

4Dオブジェクトでドラッグ&ドロップを設定する

4D Writeエリアと4Dオブジェクト間でデータをドラッグ&ドロップできます。

BLOBを除き、すべての型の4Dフィールドと変数を4D Writeエリアにドロップしたり、あるいはその逆が行えます。4D Writeエリアには元の型に基づき、自動でテキストまたはピクチャーとして挿入されます。

警告: 4Dフィールドや変数から4D Writeエリアにテキストデータをドラッグするには、操作の間**Alt** (Windows) または **Option** (Mac OS) キーを押す必要があります。

4Dエリアから4D Writeエリアに、テキストの一部を選択してドラッグ&ドロップすることはできません。オブジェクトの内容全体がコピーできます。階層リストの場合、リスト参照だけがコピーされます。リストの内容を扱うには、4Dのドラッグ&ドロップコマンドを使用します。

- 4D Writeエリアと4Dオブジェクトでドラッグ&ドロップを行う場合、4D側でドラッグ&ドロップされるオブジェクトごとに“ドラッグ可”プロパティが選択されていなければなりません。
- 4D Writeエリアがフォーム上に置かれている場合、他の4D Writeエリアからの4Dオブジェクトや要素を受け取るためには、エリアのプロパティで“ドロップ可”プロパティが選択されていなければなりません。エリアの要素をドラッグするためには、“ドラッグ可”プロパティが選択されていなければなりません。
- 外部ウィンドウで開かれた4D Writeの場合、ドラッグ&ドロップはデフォルトで有効です。ドラッグ&ドロップを制御するには **WR SET AREA PROPERTY** コマンドを使用します。

ドラッグ&ドロップをプログラムで管理する

4D Writeのデフォルトのドラッグ&ドロップにより直感的で使いやすい、より汎用的なインターフェースを作成できます。しかし特定のケースでは、このメカニズムをカスタマイズしたくなることがあります。例えば以下のようなケースです:

- 他のフォームオブジェクト (階層リスト、スクロールエリア等) からのドラッグ&ドロップを使用する
- ドラッグ&ドロップの効果を制御する、例えばドラッグしたデータを複数の場所にコピーするなど。

この場合、4Dのドラッグ&ドロップ管理コマンドと4D Writeのコマンドを組み合わせる必要があります。

まず、**On Drag Over**および**On Drop**を、使用するオブジェクトで有効にします。

4D Writeエリアのドラッグ&ドロップのプロパティは**WR GET AREA PROPERTY**と**WR SET AREA PROPERTY**コマンドで設定できます。

4D Writeエリアがフォームに含まれている場合、組み込まれたエリアオブジェクトで**On Drag Over**と**On Drop**フォームイベントで使用できます。外部ウィンドウの場合、**WR ON EVENT**コマンドを使用して特別にイベントを管理しなければなりません。

動かされたオブジェクトのタイプを正確に制御したい場合、4Dの**DRAG AND DROP PROPERTIES**コマンドを使用しなければなりません。4Dコマンドはドラッグ&ドロップの結果としていかなるアクションも実行できます。

2つの4D Writeエリア間のドラッグ&ドロップの場合、ドラッグ元のエリアは**WR GET DRAG SOURCE**コマンドで知ることができます。

オブジェクトがドロップされた4Dのエリアは`WR GET DROP TARGET`コマンドで、およびオブジェクトがドロップされた正確な位置 (ヘッダー、ボディー、フッターなどのエリアとカーソルの位置) は`WR GET DROP INFO`コマンドで知ることができます。

WR GET DRAG SOURCE

WR GET DRAG SOURCE (area ; source)

引数	型		説明
area	倍長整数	→	4D Writeエリア
source	ポインター	←	ドラッグ&ドロップのソースオブジェクトへのポインター

説明

WR GET DRAG SOURCE コマンドはドラッグ&ドロップのソースである4Dフィールド、変数、あるいは4D Writeエリア参照へのポインタを返します。

このコマンドは `wr_on_drag` 内で呼び出さなければなりません。ドラッグ&ドロップが4Dオブジェクトからであった場合、**DRAG AND DROP PROPERTIES** コマンドを使用して移動されたオブジェクトの追加の情報を取得できます。

WR GET DROP INFO

WR GET DROP INFO (area ; frame ; cursor)

引数	型		説明
area	倍長整数	→	4D Writeエリア
frame	倍長整数	←	ドキュメントのエリア
cursor	倍長整数	←	テキストの位置

説明

WR GET DROP INFO コマンドはドラッグされたデータがドロップされた位置を特定する情報を返します。このコマンドは `wr on drop` イベント内で呼び出さなければなりません。

frame 引数にはデータがドロップされた、ドキュメント中のエリアが返されます。受け取った値は **WR Frames** テーマの定数と比較できます。

cursor 引数には、*area* 内の文字列中での挿入カーソルの位置を返されます。

WR GET DROP TARGET

WR GET DROP TARGET (area ; target)

引数	型	説明
area	倍長整数	→ 4D Writeエリア
target	ポインター	← ドラッグ&ドロップのドロップ先オブジェクトのポインター

説明

WR GET DROP TARGET コマンドはドロップが行われた4Dフィールド、変数、あるいは4D Writeエリア参照へのポインタを返します。

このコマンドは `wr_on_drop` イベント内で呼び出さなければなりません。ドロップが4D Writeエリアに行われた場合、*WR GET DROP INFO* コマンドを使用してエリアやドロップが行われた位置に関する追加の情報を取得できます。ドロップが4Dオブジェクトに行われた場合、**Drop position**などの4Dコマンドを使用して処理を管理します。

WRピクチャコントロール

 ピクチャコントロールコマンドについて

 WR DELETE PICTURE IN PAGE

 WR GET PICTURE IN PAGE INFO

 WR GET PICTURE SIZE

 WR Get selected picture

 WR INSERT PICTURE

 WR SELECT PICTURE IN PAGE

 WR SET PICTURE IN PAGE INFO

 WR SET PICTURE SIZE

ピクチャコントロールコマンドについて

このテーマの4D Writeコマンドを使用して4D Writeエリアのピクチャを管理できます。これらのコマンドを使用して4D Writeエリア内のピクチャを挿入、配置、削除できます。

ピクチャ参照について

ピクチャ参照には2つの情報が格納されます。1つは参照自身でもうひとつはピクチャに関する情報です。ドキュメントがロードされ参照が計算されると、結果がピクチャであれば、ピクチャに関する情報が使用されます。参照が何も返さない場合 (例えば式がピクチャを参照しているのにレコードがない)、ピクチャ情報は失われます。ピクチャ参照を使用する場合、ドキュメントをロードする前にピクチャにアクセスできることを確認することが重要です。

WR DELETE PICTURE IN PAGE

WR DELETE PICTURE IN PAGE (area ; pictureNumber)

引数	型		説明
area	倍長整数	⇒	4D Writeエリア
pictureNumber	倍長整数	⇒	ピクチャの番号

説明

WR DELETE PICTURE IN PAGE コマンドは、*area* で指定された 4D Write エリア内で、*pictureNumber* で指定された番号のピクチャを削除します。*WR DELETE PICTURE IN PAGE* を正しく動作させるためには、ピクチャがテキスト内ではなくページ内に配置されている必要があります。テキスト内のピクチャを削除するためには、目的のピクチャを選択して *WR DELETE SELECTION* を呼び出してください。

WR count(Area;13) を使用して、エリア内のピクチャのタイプ番号を取得できます。

ピクチャを削除するとき、後に続くピクチャの番号からそれぞれ 1 が引かれます。

WR Get selected picture を使用して、ピクチャ番号を取得することもできます。

例題

以下の例は、指定されたエリアのページに配置されたすべてのピクチャを削除します。

```
$NbOccurrence:=WR Count(area;13)
For ($i;1;$NbOccurrence)
  `常に最初のピクチャを削除する
  WR DELETE PICTURE IN PAGE(area;1)
End for
```

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

引数	型	説明
area	倍長整数	⇒ 4D Writeエリア
pictureNumber	倍長整数	⇒ ピクチャ番号
page	倍長整数	⇒ ピクチャの場所
behind	整数	← 0 =ピクチャはテキストの前面 1 =ピクチャはテキストの背面
firstPage	整数	← ***廃止、使用しないでください***
horizPos	倍長整数	← ページ内の水平位置
verticalPos	倍長整数	← ページ内の垂直位置
width	倍長整数	← ピクチャの現在の幅
height	倍長整数	← ピクチャの現在の高さ
origWidth	倍長整数	← ピクチャの元の幅
origHeight	倍長整数	← ピクチャの元の高さ

説明

WR GET PICTURE IN PAGE INFOコマンドは、*area*で指定された4D Writeエリア内に現在表示されているピクチャについて、*pictureNumber*で指定したピクチャの情報を返します。

警告: このコマンドは、テキストフローの一部であるピクチャには使用しないでください。

*page*にはピクチャが表示されているページが返されます。

- -1より大きい場合、返された数値のページにピクチャが表示されます。この値は、現在定義されているページ番号付けも考慮します。
- -11の場合、奇数偶数ページでヘッダが異なる場合、ピクチャは右ページに表示されます。
- -12の場合、奇数偶数ページでヘッダが異なる場合、ピクチャは左ページに表示されます。

behind:

- 0の場合、ピクチャはテキストの前面にあります。
- 1の場合、ピクチャはテキストの背面にあります。

firstPage: この引数は互換性の目的でのみ保持されています。2004以降この引数は使用しないでください。

*horizPos*および*vertPos*は、ページの左上からの相対で、ピクチャの左上の座標を返します。値はドキュメントに設定されたカレントの単位で表現されます。

*width*と*height*には現在のピクチャの幅と高さが返されます。

*origWidth*および*origHeight*には、変更される前のピクチャの元のサイズが返されます。ピクチャのサイズが変更されていない場合は、*origWidth*および*origHeight*は、*width*および*height*と同じ値を返します。値はドキュメントに設定されたカレントの単位で表現されます。

Note: 使用する単位をピクセルに変更すると計算用に便利です。

例題

WR SET PICTURE IN PAGE INFOコマンドの例題参照

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

引数	型		説明
area	倍長整数	→	4D Writeエリア
width	倍長整数	←	ピクチャの現在の幅
height	倍長整数	←	ピクチャの現在の高さ
origWidth	倍長整数	←	ピクチャの元の幅
origHeight	倍長整数	←	ピクチャの元の高さ

説明

*WR GET PICTURE SIZE*コマンドを利用すれば、選択されたピクチャのサイズに関する情報を取得できます。ピクチャはテキストフロー内に配置されている必要があります。ページ内に埋め込まれているピクチャのサイズ情報を取得するには、*WR GET PICTURE IN PAGE INFO*を使用してください。

*WR GET PICTURE SIZE*が正しく動作するには、ピクチャだけが選択されている必要があります。

*height*はピクチャの高さです。これはドキュメントに設定された単位で表されます。

*width*はピクチャの幅です。これはドキュメントに設定された単位で表されます。

*origHeight*と*origWidth*はそれぞれピクチャのサイズが変更される前の元の高さと同幅を返します。*origHeight*と*origWidth*が、*height*と*width*とそれぞれ

一致している場合、ピクチャはサイズ変更されていません。*origHeight*と*origWidth*は、ドキュメントに設定された単位で表されます。

Note: ピクチャを選択する際には、*WR SELECT*を利用できます。

例題

*WR INSERT PICTURE*コマンドと*WR GET CURSOR POSITION*コマンドの例題を参照してください。

WR Get selected picture

WR Get selected picture (area ; status) -> 戻り値

引数	型		説明
area	倍長整数	→	4D Write area
status	整数	←	Picture status
戻り値	ピクチャー	↻	Selected picture

説明

*WR Get selected picture*は、*area*で指定された4D Writeエリア内で現在選択されているピクチャーのコピーを返します。

*status*は以下の値を返します：

- -1の場合、ピクチャーは選択されていません。
- 0の場合、選択されたピクチャーはテキストフロー内にあります。
- 0より大きい場合、選択されたピクチャーはページ内にあります。

*status*は、*WR GET PICTURE IN PAGE INFO*や*WR SET PICTURE IN PAGE INFO*、*WR DELETE PICTURE IN PAGE*を使用する際にピクチャーの確認に有効です。

例題

*WR SET PICTURE IN PAGE INFO*コマンドの例題参照

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

引数	型	説明
area	倍長整数	⇒ 4D Writeエリア
picture	ピクチャー	⇒ 挿入するピクチャ
destination	倍長整数	⇒ 挿入する位置
horizPos	倍長整数	⇒ ページ内の水平位置
verticalPos	倍長整数	⇒ ページ内の垂直位置
behind	整数	⇒ 0 =ピクチャはテキストの前面 1 =ピクチャはテキストの背面
firstPage	整数	⇒ ***廃止、使用しないでください***

説明

WR INSERT PICTUREコマンドは、areaで指定された4D Writeエリア内の、destination、horizPos、verticalPosで指定された位置にピクチャを挿入します。

pictureはピクチャフィールドまたはピクチャ変数を指定可能です。引数の内容がピクチャでない場合、エラー1065が生成されます。

オプションのdestination引数を使用して、ピクチャの挿入先を指定できます。WR Parametersテーマの以下の定数、または0より大きい任意の値を指定できます：

定数	型	値	コメント
wr into the text flow	倍長整数	0	ピクチャはテキストフロー中に挿入されます。この場合他の引数は使用されず、ピクチャは挿入ポイントに挿入されるか現在選択されている箇所を置き換えます。
wr on current page	倍長整数	-4	ピクチャはページ上に挿入され、現在のページ (挿入ポイントあるいは現在選択されている箇所) に表示されます。
wr on left hand pages	倍長整数	-12	ピクチャはページ上に挿入され、偶数と奇数ページのヘッダが異なる場合、左側のページにのみ表示されます。
wr on right hand pages	倍長整数	-11	ピクチャはページ上に挿入され、偶数と奇数ページのヘッダが異なる場合、右側のページにのみ表示されます。そうでなければすべてのページに表示されます。

0より大きい任意の値 ピクチャはページ番号がdestinationであるページに表示されます。値はページ番号の開始設定を考慮します。

horizPosとverticalPosオプション引数はドキュメントに設定されたカレントのデフォルト単位で指定します。これら2つの引数でピクチャの左上の座標を指定します (ページの左上隅を起点とする)。

オプションのbehind引数を使用して、ピクチャをテキストの前に配置するか後ろに配置するかを指定します。この引数には、WR Parametersテーマの以下の定数を指定できます：

定数	型	値	コメント
wr above text	倍長整数	0	ピクチャをテキストの上に挿入する
wr behind text	倍長整数	1	ピクチャはテキストの後ろに挿入される。この場合、テキストと背景の属性に注意を払う必要があります。"なし"を選択するとテキストの後ろのピクチャを表示することができます。

オプションのfirstPageは互換性の目的でのみ保持されています。今後は省略してください。

例題 1

以下の例題はボタンのオブジェクトメソッドです。4D Writeエリアに4Dピクチャを挿入し、50%縮小します。

```
WR INSERT PICTURE(Area;Logo) `Logoフィールドからピクチャを挿入
WR SELECT(Area;wr_select_picture;1) `ピクチャを選択
WR GET PICTURE SIZE(Area;Vert;Horiz;pictPosition) `ピクチャサイズを取得
WR SET PICTURE SIZE(Area;Vert*1/2;Horiz*1/2) `ピクチャをリサイズ
```

例題 2

ページにピクチャを挿入する例については[WR SET PICTURE IN PAGE INFO](#)コマンドの例題を参照してください。

WR SELECT PICTURE IN PAGE

WR SELECT PICTURE IN PAGE (area ; pictureNum)

引数	型		説明
area	倍長整数	⇒	4D Writeエリア
pictureNum	倍長整数	⇒	ピクチャ番号

説明

*WR SELECT PICTURE IN PAGE*コマンドを使用して、*pictureNum*で渡された番号のピクチャを選択できます。コマンドが正しく動作するために、ピクチャは（テキストフロー中ではなく）ページ内に配置されている必要があります。テキストフロー中に配置されているピクチャを選択したい場合は*WR SELECT(Area;4;XthPosition)*を使用します。*WR SELECT*コマンドのドキュメントを参照してください。

例題

*WR SET PICTURE IN PAGE INFO*コマンドの例題参照

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

引数	型	説明
area	倍長整数	⇒ 4D Writeエリア
pictureNumber	倍長整数	⇒ ピクチャ番号
page	倍長整数	⇒ ピクチャの場所
behind	整数	⇒ 0 =ピクチャはテキストの上 1 =ピクチャはテキストの後
firstPage	整数	⇒ ***廃止、使用しないでください***
horizPos	倍長整数	⇒ ページ内の水平位置
verticalPos	倍長整数	⇒ ページ内の垂直位置
width	倍長整数	⇒ 現在のピクチャの幅
height	倍長整数	⇒ 現在のピクチャの高さ

説明

WR SET PICTURE IN PAGE INFOコマンドは、*pictureNumber*で指定された番号のピクチャの属性を変更できます。

警告: このコマンドは、テキストフロー内に挿入されたピクチャには使用できません。

*page*でピクチャをどのページに表示するかを指定できます。そのためには*page*にページ番号を渡します。このページ番号は環境設定ダイアログで設定されているページ番号付けを考慮する必要があります。

- *page*が-11の場合、奇数ページと偶数ページでヘッダが異なる場合、ピクチャはページの右側に表示されます。
- *page*が-12の場合、奇数ページと偶数ページでヘッダが異なる場合、ピクチャはページの左側に表示されます。
- *page*が-4の場合、設定は変更されません。

behind: この引数にはWR Parametersテーマの以下の定数を渡すことができます。:

定数	型	値	コメント
wr above text	倍長整数	0	ピクチャをテキストの上に挿入する
wr behind text	倍長整数	1	ピクチャはテキストの後ろに挿入される。この場合、テキストと背景の属性に注意を払う必要があります。"なし"を選択するとテキストの後ろのピクチャを表示することができます。

firstPage: この引数は互換性のためにのみ保持されています。バージョン2004以降は使用せず、-1を渡してください。

*horizPos*と*verticalPos*で、物理的ページの左上隅を基点として、ピクチャの左上隅の水平および垂直座標を指定できます。*horizPos*の値は0からページの全幅までの間で設定できます。この場合、プリンタのマージンは考慮されず、ピクチャはページ中の印刷可能領域の外に配置されることもあります。

Note: ユーザーモードでピクチャをペーストする場合、プリンタのマージンは考慮されます。

*width*と*height*はピクチャの新しいサイズを指定できます。値はドキュメントのカレントデフォルト単位で指定します。

Note: *behind*, *firstPage*, *horizPos*, *verticalPos*, *width* そして *height* 引数に-1を渡すと、コマンド実行時に設定されていた値は変更されません。

例題

同じピクチャを異なるドキュメントに挿入します:

```
C_REAL($PosHoriz;$PosVert;$PictWidth;$PictHeight;$OrigWidth;$OrigHeight;$TxtMgTop;$HeadMgBottom)
WR SET DOC PROPERTY(Area;wr_view_mode;0)
$PosHoriz:=WR Get doc property(Area;wr_text_left_margin)
$PosVert:=WR Get doc property(Area;wr_header_top_margin)
ALL RECORDS([Interface])
  `ピクチャを挿入
WR INSERT PICTURE(Area;[Interface]Logo;-1;$PosHoriz;$PosVert;1;0) `ピクチャはLogoフィールドに格納されている
WR SELECT PICTURE IN PAGE(Area;1) `ピクチャを選択
  `ピクチャ属性を取得
```

```
MyPict:=WR Get selected picture(Area;$NumPict)
WR GET PICTURE IN PAGE
INFO(Area;$NumPict;$Page;$Behind;$PageOne;$PosHoriz;$PosVert;$PictWidth;$PictHeight;$OrigWidth;$OrigHeight)
  `ピクチャサイズを50%縮小
$PictHeight:=$PictHeight*1/2
$PictWidth:=$PictWidth*1/2
WR SET PICTURE IN PAGE
INFO(Area;$NumPict;$Page;$Behind;$PageOne;$PosHoriz;$PosVert;$PictWidth;$PictHeight)
  `ヘッダにロゴを挿入できるかチェック
$TxtMgTop:=WR Get doc property(Area;wr_text_top_margin)
$HeadMgBottom:=WR Get doc property(Area;wr_header_bottom_margin)
WR SET DOC PROPERTY(Area;wr_text_top_margin;$PosVert+$PictHeight+$TxtMgTop+$HeadMgBottom)
WR SET DOC PROPERTY(Area;wr_header_bottom_margin;$PosVert+$PictHeight)
```

WR SET PICTURE SIZE

WR SET PICTURE SIZE (area ; width ; height)

引数	型		説明
area	倍長整数	→	4D Writeエリア
width	倍長整数	→	ピクチャの新しい幅
height	倍長整数	→	ピクチャの新しい高さ

説明

*WR SET PICTURE SIZE*コマンドを使用することにより、*area*で指定した4D Writeエリア内の、選択されたピクチャのサイズを変更できます。

このコマンドは、バックグラウンドピクチャには効果がありません。バックグラウンドピクチャのサイズを変更するには*WR SET PICTURE IN PAGE INFO*を使用します。

*width*および*height*は、ドキュメントのカレントデフォルト単位で表されます。これらの値はページの範囲内、複数の段組を使用する場合は段組の範囲内で指定します。

単位としてピクセルを使用するには、ドキュメントのカレントデフォルト単位を一時的に変更し、*WR SET PICTURE SIZE*を実行した後に元の設定に戻します。

例題

*WR INSERT PICTURE*コマンドの例題を参照

WRプリント

-  プリントコマンドについて
-  WR BLOB TO PRINT SETTINGS
-  WR GET PRINT OPTION
-  WR PRINT
-  WR PRINT MERGE
-  WR Print settings to BLOB
-  WR SET PRINT OPTION

✚ プリントコマンドについて

このテーマの4D Writeコマンドや関数を使用して、4D Writeエリアのプリントを制御できます。

これらのコマンドで、ユーザーに**ファイル**メニューから**プリント**を選択させることなしに、レポートやレターを印刷させることができます。

Note: 4Dコマンドを使用してカレントプリンターを取得したり設定したりできます。プリンターを変更しても印刷オプションは変更されません (新しく指定されたプリンターで使用できないオプションを除く)。

WR BLOB TO PRINT SETTINGS

WR BLOB TO PRINT SETTINGS (*area* ; *printSettings* ; *paramType*)

引数	型	説明
<i>area</i>	倍長整数	→ 4D Writeエリア
<i>printSettings</i>	BLOB	→ 印刷設定を格納したBLOB
<i>paramType</i>	倍長整数	→ 0 = 用紙設定および印刷設定、1 = 印刷設定

説明

WR BLOB TO PRINT SETTINGSコマンドは4D Writeの*area*エリアの現在の印刷設定を、*printSettings* BLOBの内容で置き換えます。

*area*は外部ウィンドウ、フォーム上のエリア、あるいはオフスクリーンエリアを指定できます。しかし4D Writeの印刷設定を管理するメカニズムのため、*area*に0を渡すことですべてのエリアにコマンドを適用することはできません。

printSettings BLOBはWR Print settings to BLOBコマンドで生成されたものでなければなりません。

printSettings には2タイプの設定が含まれます:

- 用紙設定 (用紙サイズ, 向き, 倍率);
- 印刷設定 (枚数, トレイ, etc.).

Note: Windowsでは、BLOBに格納された設定にプリンターを含みます。

paramType 引数には、WR Parametersテーマの以下の定数を渡します:

定数	型	値	コメント
wr layout and print settings	倍長整数	0	プリントおよび用紙設定が使用されます。
wr print settings only	倍長整数	1	プリント設定のみが使用されます。

新しい印刷設定は*area*で指定されたドキュメントに適用されます。

Note: 印刷設定はMac OSとWindowsでフォーマットが異なります。従って、両OS間での*printSettings* BLOBの互換性は保証されません。

システム変数およびセット

BLOBが正しくロードされるとシステム変数OKに1が、そうでなければ0が設定されます。

エラー管理

プリンターが選択されていない場合、エラー1014が生成されます。*printSettings* BLOBが有効な印刷設定を含んでいない場合、1074が生成されます。

WR GET PRINT OPTION (area ; option ; value1 ; value2 ; value3)

引数	型	説明
area	倍長整数	4D Writeエリア
option	倍長整数	オプション番号
value1	倍長整数	オプションの値1
value2	倍長整数	オプションの値2
value3	文字	オプションの値3

説明

WR GET PRINT OPTIONコマンドはoptionで指定したプリントオプションの現在値を返します。

option引数で、取得するオプションを指定します。値あるいはWR Print optionsテーマの以下の定義済み定数を渡せます:

定数	型	値
wr color option	倍長整数	8
wr destination option	倍長整数	9
wr double sided option	倍長整数	11
wr number of copies option	倍長整数	4
wr orientation option	倍長整数	2
wr pages from option	倍長整数	6
wr pages to option	倍長整数	7
wr paper option	倍長整数	1
wr paper source option	倍長整数	5
wr scale option	倍長整数	3
wr spooler document name option	倍長整数	12

コマンドはoptionで指定したオプションの現在値をvalue1および(オプションで) value2 と value3 引数に返します。オプションおよびその値についてはWR SET PRINT OPTIONコマンドの説明を参照してください。WR GET PRINT OPTIONコマンド特有の動作については以下を参照してください:

- option = 1 (wr paper option): value2とvalue3が省略されていれば、現在の用紙名をvalue1に返します。value3のみが省略されている場合、コマンドは用紙の高さと幅をそれぞれvalue1とvalue2に返します。プリンターで使用できるすべての名前、高さ、幅を取得するには、PRINT OPTION VALUESコマンドを使用します。
- option = 2 (wr orientation option): 1 (縦) または 2 (横) を返します。異なる方向オプションが使用されていると、value1は0に設定されます (value2とvalue3は省略されなければなりません)。
- option = 5 (wr paper source option): PRINT OPTION VALUESコマンドから返されるトレイ配列において、使用されている用紙トレイのインデックスがvalue1に返されます (value2とvalue3は省略されなければなりません)。
Note: このオプションはWindowsでのみ使用できます。
- option = 6とoption = 7 (wr pages from optionとwr pages to option): すべてのページが印刷される場合、コマンドはwr pages from optionのときvalue1に1を、wr pages to optionのときvalue1に-1を返します (value2とvalue3は省略されなければなりません)。
- option = 8 (wr color option): カラーを処理するモードを特定するコードをvalue1に返します: 1=白黒 (モノクロ), 2=カラー (value2とvalue3は省略されなければなりません)。
Note: このオプションはWindowsでのみ使用できます。
- option = 9 (wr destination option): 現在値が定義済みのリストにない場合、value1に-1が返され、システム変数OKに1が設定されます。エラーが発生するとvalue1とシステム変数OKに0が設定されます。value1に1以外の定義済みの値が返される場合、value3には印刷されたファイルのアクセスパスが返されます。value2には常に0が返されます。
- option = 11 (wr double sided option): value1には0 (標準または一面、デフォルト値) または1 (両面) が返されます。value1が1の場合、value2には以下のいずれかの値が返されます: 0=左綴じ (デフォルト), 1=上綴じ (value3は省略する)。
Note: このオプションはWindowsでのみ使用できます。
- option = 12 (wr spooler document name option): 事前に定義されていれば、value3に現在のプリントドキュメント名が返されます (value1 と value2 には0が返されます)。そうでない場合、空の文字列が返されます。

コマンドが正しく実行されるとシステム変数OKに1が、そうでない場合0が設定されます。

WR PRINT (area ; mode ; nbCopies)

引数	型		説明
area	倍長整数	⇒	4D Write エリア
mode	整数	⇒	0 = 値 1 = 参照
nbCopies	整数	⇒	印刷枚数

説明

WR PRINTコマンドは、*area*で指定されたエリア内のドキュメントを印刷します。このコマンドは**ファイルメニューからプリント**を選択した場合と同じ処理を行います。印刷ダイアログボックスを表示しません。

WR PRINTは、*area* で指定されたエリアを一度だけ印刷します。セレクション内のレコード毎に*area* を印刷したい場合はWR PRINT MERGEを使用します。

印刷には2つの選択肢があります：

- *mode*に定数`wr_print_references` (値1) を渡すと、参照要素は4D Writeエリア内で二重括弧 (<<>>) 内に表示されます。
- *mode*に定数`wr_printvalues` (値0) を渡すと、参照要素の値が4D Writeエリア内に印刷されます。
WR PRINTは参照を計算しません。印刷前に参照を更新したい場合は、WR PRINTの前に、WR EXECUTE COMMAND (area;wr_cmd_compute_references) を実行します。

*nbCopies*引数で印刷する枚数を指定します。

例題

以下は、*area*を含むフォーム上のボタンで使用されるスクリプトの例です。このボタンをクリックすると、*area*で指定されたエリア内が印刷されます。印刷前には参照が更新されます：

```
WR EXECUTE COMMAND(area;wr_cmd_compute_references)
WR PRINT(area;wr_print_values;1)
```

WR PRINT MERGE (area ; numTable ; display)

引数	型	説明
area	倍長整数	→ 4D Write エリア
numTable	整数	→ テーブル番号
display	整数	→ 印刷設定ダイアログボックスの表示/非表示

説明

WR PRINT MERGEコマンドは、area に含まれるドキュメントを、tableのカレントセレクションのレコード毎に1回ずつ印刷します。table は、差し込みを行うレコードが属するテーブルの番号です。table が0の場合、WR PRINT MERGEは標準の差し込みプリントダイアログボックスを表示し、テーブルの指定、またはそのテーブルのカレントセレクションを変更することができます。

ドキュメントに参照が含まれる場合、印刷前に自動で更新されます。

display引数にはWR Parametersテーマの以下の定数を渡します。:

定数	型	値	コメント
wr no print settings dialog	倍長整数	0	印刷設定ダイアログボックスを表示しない。
wr with print settings dialog	倍長整数	1	印刷設定ダイアログが表示されます。

例題

以下は、[Clients]テーブル (テーブル番号3) の各レコードについて手紙を印刷する例です。この手紙は[Letters]テーブルに保存されています。

```

ALL RECORDS (Clients) `全顧客をセレクションにする
QUERY ([Letters]; [Letters]Ref="Expedite") `Expediteテンプレートを検索
Temp:=WR New offscreen area `オフスクリーンエリアを作成
WR PICTURE TO AREA(Temp; [Letters]Doc_) `オフスクリーンエリアにテンプレートをロード
WR PRINT MERGE(Temp; 3; wr_no_print_settings_dialog) `Clientsテーブルのセレクションを使用して差し込み印刷
WR DELETE OFFSCREEN AREA(Temp) `オフスクリーンエリアを削除
    
```

WR Print settings to BLOB

WR Print settings to BLOB (area) -> 戻り値

引数	型		説明
area	倍長整数	→	4D Writeエリア
戻り値	BLOB	↩	印刷設定を格納したBLOB

説明

WR Print settings to BLOB コマンドは *area* で指定された4D Writeエリアの現在の印刷設定をBlobに格納して返します。*area*は外部ウィンドウ、フォーム上のエリア、あるいはオフスクリーンエリアです。

BLOBには印刷に使用されるすべての設定が格納されます:

- 用紙設定 (用紙サイズ, 向き, 倍率);
- 印刷設定 (枚数, トレイ, etc.).

Note: Windowsでは、BLOBに格納された設定にプリンターを含みます。

このコマンドを使用して、プリンターのモデルやアクセス可能な印刷設定に関係なく、4D Writeエリアの印刷設定を保存できます。返されるBLOBをプログラムで変更してはいけません。このBLOBは*WR BLOB TO PRINT SETTINGS*コマンドでのみ使用できます。

WR Print settings to BLOB コマンドは例えば*WR SET PRINT OPTION*コマンドを使用して印刷設定を一時的に変更する前に、現在の印刷設定を保持するために使用できます。印刷が終了したら、*WR BLOB TO PRINT SETTINGS*コマンドを使用して現在の設定を復元できます。

システム変数およびセット

BLOBが正しく生成されるとシステム変数OKに1が、そうでなければ0が設定されます。

エラー管理

プリンターが選択されていない場合、エラー1014が生成されます。

WR SET PRINT OPTION (area ; option ; value1 ; value2 ; value3)

引数	型		説明
area	倍長整数	⇒	4D Writeエリア
option	倍長整数	⇒	オプション番号
value1	倍長整数	⇒	オプションの値1
value2	倍長整数	⇒	オプションの値2
value3	文字	⇒	オプションの値3

説明

WR SET PRINT OPTIONコマンドは、area 引数で指定した4D Writeエリアの印刷オプションをプログラムで変更するために使用します。このコマンドを使用して定義されたオプションは、area が削除されるまで適用されます。通常4D Writeドキュメントに保存される (用紙方向のような) オプションも保存されます。

4Dおよび他の4D Writeの現在のプリントパラメータは変更されません。

optionは変更するオプションを指定します。値あるいはWR Print optionsテーマの定義済み定数を指定できます。

指定したoption の新しい値をvalue1 および (オプションで) value2 と value3 引数に渡します。渡すことのできる値はoptionにより異なります。

値によってはWR Parametersの定数を使用して指定します。オプションと対応する値について以下の表で説明します:

オプション定数 (値)	value1	value2	value3
<u>wr paper option</u> (1)	高さ 0	幅 0	- 名前
<u>wr orientation option</u> (2)	<u>wr portrait</u> (1), <u>wr landscape</u> (2)	-	-
<u>wr scale option</u> (3)	数値 (%)	-	-
<u>wr number of copies option</u> (4)	数値	-	-
<u>wr paper source option</u> (5)	Windowsのみ: インデックス (数値)	-	-
<u>wr pages from option</u> (6)	数値 (1=デフォルト)	-	-
<u>wr pages to option</u> (7)	数値 (1=デフォルト, ドキュメントの終わり)	-	-
<u>wr color option</u> (8)	<u>wr black and white</u> (1), <u>wr color</u> (2)	-	-
<u>wr destination option</u> (9)	<u>wr send to printer</u> (1), <u>wr send to file</u> (2), <u>wr send to PDF file</u> (3)	0 0 0	- アクセスパス アクセスパス
<u>wr double sided option</u> (11)	Windows のみ: <u>wr single sided</u> (0) (標準) <u>wr double sided</u> (1) (default), <u>wr top binding</u>	- - <u>wr left binding</u> (0)	- -
<u>wr spooler document name option</u> (12)	0	0	ドキュメント名

- wr paper option (1): プリンターで使用できるすべての名前、高さ、幅を取得するには、**PRINT OPTION VALUES**コマンドを使用します。
value3に用紙名 (この場合value1とvalue2には0を渡す)、あるいは用紙の高さをvalue1に、幅をvalue2に渡します。高さおよび幅はピクセルで指定します。
- wr orientation option (2): value1に定数wr portrait (1) または wr landscape (2) を渡します。
- wr scale option (3): value1に倍率を渡します。プリンターによっては倍率の変更ができない点に留意してください。無効な値を渡すと、印刷時にプロパティは100%にリセットされます。
- wr number of copies option (4): value1に印刷枚数を渡します。
- wr paper source option (5): value1には**PRINT OPTION VALUES**コマンドから返されるnamesArray配列の要素に対応するinfo1Array要素の値を渡します。この配列には使用する用紙トレイの名前が含まれます。
注:このオプションはWindowsでのみ使用できます。

- wr pages from option (6): Pass the number of the page where you want printing to start in *value1*. The default value is 1.
- wr pages to option (7): Pass the number of the last page that you want to be printed in *value1*. If you pass -1, the entire document will be printed (-1 is equivalent to passing the last page of the document).
- wr color option (8): In *value1*, pass the constant wr black and white (1) (monochrome) or wr color (2).
Note: このオプションはWindowsでのみ使用できます。
- wr destination option (9): *value1*に以下の定数のいずれかを渡します: wr send to printer (1), wr send to file (2) (Windowsではファイル、Mac OSではPS) または wr send to PDF file (3) (Mac OSのみ)。
*value2*には常に0を渡します。
*value1*が1以外の場合、ドキュメントのアクセスパスを*value3*に渡します。このパスは他のパスが指定されるまで使用されます。同じ名前のファイルが存在する場合、それは置き換えられます。
Windowsのみ: *value3*に空の文字列を渡すかこの引数を省略すると、印刷時にファイルを保存ダイアログが表示されます。処理に失敗すると、プリンター (1) 設定が適用されます。
- wr double sided option (11): 定数wr single sided (0) (標準) または wr double sided (1) を*value1*に渡せま
す。*value1*が1に設定されると、*value2*を使用して綴じのタイプを設定できます: 定数wr left binding (0, デフォルト
値) または wr top binding (1).
Note: このオプションはWindowsでのみ使用できます。
- wr spooler document name option (12): *value3*に、スプーラードキュメント名として表示する名前を渡しま
す。*value1* と *value2*には0を渡します。
標準の処理 (メソッドの場合メソッド名、レコードの場合テーブル名等) を使用または復元するには、空の文字列
を*value3*に渡します。
警告: このステートメントで指定され名前は、新しい名前が指定されるか空の文字列が渡されるまで、セッション中の
すべての印刷ドキュメントで使用されます。

*option*に渡された値が無効かプリンターで利用できない場合、コマンドはエラーを返し、オプションの現在値は変更されませ
ん。このエラーはWR ON ERRORコマンドでインストールされるエラー処理メソッドでとらえることができます。

コマンドが正しく実行されるとOKシステム変数は1に、そうでなければ0に設定されます。

WRユーティリティ

 ユーティリティコマンドについて

 WR COLOR TO RGB

 WR Count

 WR Error number

 WR Error text

 WR FONTS TO ARRAY

 WR Get on error method

 WR Get on event method

 WR ON ERROR

 WR ON EVENT

 WR RGB to color

✦ ユーティリティコマンドについて

このテーマのコマンドや関数はエラーやイベントの処理を提供し、4D Writeエリアの制御を可能にします。

*WR Count*関数は、4D Writeエリアの内容に関する基本的な情報を取得するために使用できます。*WR FONTS TO ARRAY*コマンドを使用してOSにインストールされているフォントのリストを取得できます。

また、カラー管理コマンドを使用して、4D Writeエリアに表示されるカラーをコントロールできます。

WR COLOR TO RGB

WR COLOR TO RGB (color ; red ; green ; blue)

引数	型		説明
color	倍長整数	⇒	色
red	倍長整数	⇐	赤の値を受け取る (0-65535)
green	倍長整数	⇐	緑の値を受け取る (0-65535)
blue	倍長整数	⇐	青の値を受け取る (0-65535)

説明

WR COLOR TO RGBコマンドはcolor で指定された色を、色の3 要素であるred 、 green 、 blueにマッピングします。これらの値の範囲は0から65535です。

color は4D Write で使用される内部番号で、WR RGB to colorで取得することができます。

例題

以下の例題は与えられたカラーから灰色に近いものを計算します：

```
WR COLOR TO RGB(Color;Red;Green;Blue)
Blue:=(Blue+Green+Red)/3
Grey:=WR RGB to color(Blue;Blue;Blue)
```

WR Count (area ; objectNumber) -> 戻り値

引数	型		説明
area	倍長整数	→	4D Write エリア
objectNumber	整数	→	オブジェクト番号
戻り値	倍長整数	↩	オブジェクト数

説明

WR Countコマンドを使用して、指定エリア内にある指定されたオブジェクトのオカレンス数をカウントできます。カウントできるオブジェクトには、次のものがあります：

オブジェクト	定数	ObjectNumber
文字	wr nb characters	0
単語	wr nb words	1
段落	wr nb paragraphs	2
テキストフロー中のピクチャー	wr nb pictures in text flow	3
参照	wr nb objects	4
ハイフン	wr nb soft hyphens	5
改ページ	wr nb page breaks	6
段組み	wr nb column breaks	7
時間オブジェクト	wr nb insertions date time	8
ページ番号	wr nb insertions page number	9
行	wr nb lines	10
ページ	wr nb pages	11
スタイルシート	wr nb stylesheets	12
ページ中の画像 (バックグラウンド)	wr nb pictures in page	13
ハイパーリンク	wr nb hyperlinks	14 (6.7)
RTF式	wr nb RTF expressions	15 (6.7)
HTML式	wr nb HTML expressions	16 (6.7)

- *objectNumber*=3の場合、バックグラウンドピクチャは無視されます。バックグラウンドピクチャをカウントするには*objectNumber* = 13を使用します)。
- *objectNumber*=12の場合、WR Countは標準のスタイルシート (デフォルトスタイルシート) を含むスタイルシート数を返します。
- *objectNumber*=13で画像が複数ページで繰り返されている (ピクチャプロパティでそのように選択されている) 場合、画像は1つとカウントされます。

間違ったarea参照を渡すと、エラー1022が生成されます。

例題

以下のコマンドの例題を参照: [WR SELECT](#), [WR INSERT PAGE NUMBER](#), [WR DELETE PICTURE IN PAGE](#), [WR GET WORDS](#), [WR GET PARAGRAPHS](#)そして[WR UPDATE STYLESHEET](#).

WR Error number

WR Error number (area) -> 戻り値

引数	型		説明
area	倍長整数	→	4D Write エリア
戻り値	整数	↩	4D Writeがエリアに対して行った最後の処理のステータス

説明

WR Error number コマンドは、4D Write によって *area* 内で実行された最後の操作のステータスを表す番号を返します。*WR Error number* が 0 を返した場合、最後の処理でエラーが発生しなかったことを意味します。*WR Error number* が 0 を返さなかった場合、*area* 内の最後の操作を行っている途中で何らかのエラーが発生したことになります。

エラーメッセージを取得するには、*WR Error text* を使用します。また、デバッグウィンドウが開いている際にエラーが発生すると、そのデバッグウィンドウ内にエラー番号が表示されます。

例題

[WR Error text](#) の例題参照

WR Error text

WR Error text (error) -> 戻り値

引数	型	説明
error	整数	→ エラー番号
戻り値	文字	↩ errorに対応する、エラーを説明したテキスト

説明

WR Error text コマンドは、*error* で指定されたエラーを説明するテキストを返します。この関数を使用して、*WR Error number* から返されたエラーのメッセージを受け取ることができます。

例題

次の例は、エラーをチェックして、ユーザが Designer かどうかによって異なるエラーメッセージを表示します:

```
$Error:=WR Error number(Area)
If ($Error#0)
  If (Current user="Designer")
    ALERT (WR Error text($Error))
  Else
    ALERT ("問題が発生しました。担当者に連絡してください。")
  End if
End if
```

WR FONTS TO ARRAY (fonts)

引数	型	説明
fonts	文字配列	← 利用可能なフォントを受け取る配列

説明

WR FONTS TO ARRAY コマンドは利用可能なフォント一覧を *fonts* 配列に返します。この一覧は書式パレットのフォントドロップダウンリストに対応します。

fonts はテキストタイプの配列として宣言されなければなりません。

例題

テンプレートで使用しているフォントが現在のシステムにインストールされているかどうかをチェックします。[Fonts]テーブルには必要なフォントのリストが保存されています。[On Startupデータベースメソッド](#)に以下のように書くことができます:

```
ARRAY TEXT (aFonts;0)
WR FONTS TO ARRAY(aFonts)
ALL RECORDS ([Fonts])
While (Not (End selection ([Fonts])))
  If (Find in array (aFonts; [Fonts]Name)=-1)
    ALERT ([Fonts]Name+"フォントが必要です。インストールしてください。")
  End if
  NEXT RECORD ([Fonts])
End while
```

WR Get on error method

WR Get on error method -> 戻り値

引数	型	説明
戻り値	文字 	エラー時に実行されるメソッド名

説明

WR Get on error method コマンドは *WR ON ERROR* でインストールされたエラー処理メソッドの名前を返します。
エラー処理メソッドがインストールされていない場合、空の文字列 ("") が返されます。

🔧 WR Get on event method

WR Get on event method (area ; event) -> 戻り値

引数	型		説明
area	倍長整数	→	4D Writeエリア
event	倍長整数	→	イベントコード
戻り値	文字	↩	インストールされているイベント処理メソッドの名前

説明

WR Get on event methodコマンドを使用して、*area*で指定された4D Writeエリアで、*event* 引数により指定されたイベントに対し、**WR ON EVENT**コマンドでインストールされたイベント処理メソッドの名前を取得できます。

イベント処理メソッドがインストールされていない場合、空の文字列 ("") が返されます。

event 引数には、メソッド名を取得するイベントを示す値を渡します。**WR Events**テーマの以下の定数を渡せます:

定数	型	値	コメント
wr on key	倍長整数	0	キーが押された (矢印キーやリターン等を含む)
wr on double click	倍長整数	1	ダブルクリック
wr on single click	倍長整数	2	シングルクリック
wr on triple click	倍長整数	3	3回クリック
wr on right click	倍長整数	4	右マウスボタンのクリック
wr on activate	倍長整数	5	4D Writeエリアがアクティブまたは非アクティブになった
wr on printing	倍長整数	7	ドキュメント印刷中
wr on ruler	倍長整数	8	ルーラの変更
wr on compute references	倍長整数	9	動的参照が更新された
wr on close	倍長整数	10	4D Writeエリアまたはウィンドウが閉じられた
wr on drag	倍長整数	11	オブジェクトがドラッグされた
wr on drop	倍長整数	12	オブジェクトがドロップされた
wr on timer	倍長整数	13	タイマーサイクルの終了

WR ON ERROR (method)

引数	型	説明
method	文字 	メソッドの名前

説明

WR ON ERROR コマンドは、4D Write のエラーを管理するためのメソッドとして、*method* をインストールします。このメソッドは4D Write エリアでエラーが発生する度に実行されます。これによりアプリケーション内で実行時エラーをモニタすることができます。

呼び出されるメソッドには3つの引数が渡されます:

- \$1はエリアを表します。
- \$2はエラー番号を表します。
- \$3はエラーテキストを表します。

Note: データベースをコンパイルする場合\$1と\$2を倍長整数に、\$3をテキストとして宣言しなければなりません。

メソッドの実行が終了すると、4Dは元のメソッドに制御を返します。*method*が空の文字列の場合、*WR ON ERROR*はインストール済みのエラー処理メソッドをアンインストールします。

例題

4D Writeにエラー処理メソッドをインストールします。

```
WR ON ERROR("WriteArea")
```

WriteAreaメソッドはエラー番号とエラーテキストを表示します。

```
ALERT ("エラー番号 " + String($2) + Char(13) + $3)
```

WR ON EVENT (area ; event ; method)

引数	型		説明
area	倍長整数	⇒	4D Write エリア
event	倍長整数	⇒	イベントコード
method	文字	⇒	実行するメソッド

説明

WR ON EVENTコマンドは、*area*内で*event*で指定されたイベントが発生したときに呼び出されるメソッドとして*method*をインストールします。イベントは4D Writeで処理される前に直接メソッドに渡されます。

*area*が0の場合、*method*は、データベースが閉じられるまで、すべての4D Writeエリアに対するデフォルトのイベントメソッドとなります。エリアに特定のイベントメソッドがインストールされている場合、そのメソッドがデフォルトメソッドに代わって呼び出されます。

*event*引数には傍受するイベントを表す値を渡します。**WR Events**テーマの以下の定数を渡せます：

定数	型	値	コメント
wr on key	倍長整数	0	キーが押された (矢印キーやリターン等を含む)
wr on double click	倍長整数	1	ダブルクリック
wr on single click	倍長整数	2	シングルクリック
wr on triple click	倍長整数	3	3回クリック
wr on right click	倍長整数	4	右マウスボタンのクリック
wr on activate	倍長整数	5	4D Writeエリアがアクティブまたは非アクティブになった
wr on printing	倍長整数	7	ドキュメント印刷中
wr on ruler	倍長整数	8	ルーラの変更
wr on compute references	倍長整数	9	動的参照が更新された
wr on close	倍長整数	10	4D Writeエリアまたはウィンドウが閉じられた
wr on drag	倍長整数	11	オブジェクトがドラッグされた
wr on drop	倍長整数	12	オブジェクトがドロップされた
wr on timer	倍長整数	13	タイマーサイクルの終了

すべてのイベントに対し*method*を有効にするには、*event*に-1を渡します。

*method*は呼び出されるときに、イベント発生時のエリアのステータスを表す7つの引数を受け取ります。コンパイルコマンドを使用して、これらの引数を宣言する必要があります。次の表は、*method*が受け取る引数の説明です：

引数	型	説明
\$1	倍長整数	4D Writeエリア
\$2	整数	Shiftキー
\$3	整数	Alt (Windows), Option (Mac OS)
\$4	整数	Ctrl (Windows), Command (Mac OS)
\$5	整数	イベントタイプ
\$6	整数	イベントタイプにより異なる
\$0	倍長整数	メソッドが値を返す場合

\$1にはイベントが発生したエリアのIDが渡されます。\$2, \$3, そして\$4にはイベント発生時にモディファイアキーが押されていたかどうかを示す値が渡されます。値が0の場合、キーは押されていません。値が1の場合、キーは押されています。\$5にはイベントタイプが渡されます。\$6はイベントのタイプにより異なります。

メソッド変数とevent引数 (\$6)

- *event*=0の場合、\$6にはイベントを呼び出したキーのコードが渡されます。
- *event*=1または2の場合、\$6は参照がシングルまたはダブルクリックされたことを示します。\$6=0のとき、参照は選択されていません。\$6=1のとき、参照は選択されています。

Note: 以下のいずれかのアクションが実行された場合、*method*はクリックを管理する前に呼び出されることがあります：

- 参照 (ハイパーテキストリンク、4DまたはHTML式) をシングルまたはダブルクリックした。
- 右クリック (Windows) または Control-クリック (Mac OS)。Mac OSでは、クリック時にControlキーを押すとポップアップメニューが表示されます。
Windowsでは右クリックを行うドロップダウンメニューが表示されます。両ケースともデータベースフィールドを表示します。互換性のためにイベント4 (*wr on right click*) の利用をお勧めします。

- *event*=3の場合、\$6は段落の選択に関連します。ダブルクリックで呼び出されるイベントメソッドがインストールされておらず、これが\$0:=1で傍受されていない場合、トリプルクリックが参照上で行われることがあります。この場合、\$6に意味はありません。
- *event*=4の場合、\$6は (クリックされた場所に基づく) 表示されようとしているコンテキストメニューのタイプを示します:
 - \$6=1の場合、タイプ1のコンテキストメニュー (ヘッダー/フッター内のクリック) が表示されます。
 - \$6=2の場合、タイプ2のコンテキストメニュー (ボディエリアのテキスト内のクリック) が表示されます。
 - \$6=3の場合、タイプ3のコンテキストメニュー (ボディエリアのピクチャのクリック) が表示されます。
- *event*=5の場合、\$6はエリアがアクティブか非アクティブかを示します。\$6=0の場合、4D Writeエリアは非アクティブです。\$6=1の場合、4D Writeエリアはアクティブです。
- *event*=7でプリントジョブが差し込み印刷の場合、\$6は使用されているテーブル番号を示します。プリントジョブが差し込み印刷でない場合、\$6は0となります。
- *event*=8の場合 (ルーラー上でアクションが行われた)、\$6は意味のある値を返しません。ルーラー上のアクションを許可したくない場合、\$0に1を設定します。
- *event*=9の場合、\$6はドキュメント内でどのマージンがリセットされたかを示します。\$6=0の場合、マージンはボディ内でリセットされました。\$6=1の場合、マージンはヘッダでリセットされました。\$6=2の場合、マージンはフッタでリセットされました。
- *event*=13の場合、ユーザーアクションに関わらず、*method*が自動でXtick毎に呼び出されます (1 tick = 1/60秒)。タイマーは特に編集中のドキュメントを自動でバックアップするセキュリティメカニズムを実装するために使用されます。デフォルトでタイマーは3600 tick (60秒) 毎にイベントを生成します。この周期はWR SET AREA PROPERTYコマンドを使用して変更できます。メソッドは繰り返し実行されるため、アプリケーションを遅くしないよう、*method*が時間のかかる処理を行わないようにしなければなりません。

イベントをフィルタするには*method*を関数として使用し、0または1を返します。これにより、4D Writeで特定の文字を無視するようにすることができます。

\$0に1を設定すると、メソッドは特定のイベントをトラップします。\$0を0に設定するとイベントはトラップされません。例えば、ドキュメント中に"@ "を入力させたくない場合、ドキュメント中に現れるすべての文字をフィルタします。\$6変数が"@ "の文字コードと同じであれば、\$0に1を設定します。

Note: すべての文字をフィルタすると、キーが押されるたびにメソッドが実行されるため、動作は遅くなります。

例題

以下の例題では、イベントのタイプに基づきアクションを実行します:
フォームメソッド:

```
If (Form event=On Load)
  WR ON EVENT(Area;wr_on_key;"ProcName")
  `すべてのキーストロークに対し呼び出す
  WR ON EVENT(Area;wr_on_activate;"ProcName")
  `エリアのステータスをチェック
  DISABLE MENU ITEM(2;1)
  `フォントを変更メニュー項目を無効に
  WR SET AREA PROPERTY(Area;wr_timer_frequency;54000)
  `15 分毎にタイマーイベント
  WR ON EVENT(Area;wr_on_timer;"ProcName")
  `自動保存を設定
End if
```

ProcName メソッド:

```
Case of
  : ($5=wr_on_key)
  `キーストロークを傍受
  If ($6=199) &NBSP; | &NBSP; ($6=200)
  `対応するASCIIコード
  BEEP
```

```
    $0:=1
  Else
    `4D WriteIにイベントを処理させる
    $0:=0
  End if
  :($5=wr_on_activate)
  `エリアのステータスの変更を傍受
  If($6=0)
    `エリアが非アクティブなら
    DISABLE MENU ITEM(2;1)
  Else
    ENABLE MENU ITEM(2;1)
  End if
  :($5=wr_on_timer)
  `15分毎に
  $DocName:="C:\\Temp\\Docs\\TheArea.4W7"
  WR SAVE DOCUMENT(TheArea;$DocName;"4WR7")
End case
```

WR RGB to color (red ; green ; blue) -> 戻り値

引数	型		説明
red	倍長整数	→	赤の要素 (0-65535)
green	倍長整数	→	緑の要素 (0-65535)
blue	倍長整数	→	青の要素 (0-65535)
戻り値	倍長整数	↩	色

説明

WR RGB to color コマンドは、色を管理するために4D Writeで使用される色の番号を返します。この番号はカラーの3要素であるred、green、blueを意味しています。red、green、blueは使用しているシステムのカラーピッカーで用いられている値と同じものです。これらの値の範囲は0から65535です。

例題 1

それぞれのカラーを取得します:

```
RedColor:=WR RGB to color(56576;2048;1536)
  `赤が返されます
GreenColor:=WR RGB to color(0;32768;4352)
  `緑が返されます
BlueColor:=WR RGB to color(0;0;54272)
  `青が返されます
CyanColor:=WR RGB to color(512;43776;59904)
  `シアンが返されます
MagentaColor:=WR RGB to color(64512;62208;1280)
  `マゼンタが返されます
YellowColor:=WR RGB to color(61952;2048;33792)
  `黄が返されます
```

例題 2

以下の例題は2つの色の中間色を返します:

```
WR COLOR TO RGB(c1;r1;g1;b1)
WR COLOR TO RGB(c2;r2;g2;b2)
c3:=WR RGB to color((r1+r2)/2;(g1+g2)/2;(b1+b2)/2)
```

定数のリスト

-  WR Area properties
-  WR Commands
-  WR Count
-  WR Document properties
-  WR Document types
-  WR Events
-  WR Frames
-  WR Page number formats
-  WR Parameters
-  WR Print options
-  WR Select type
-  WR Standard colors
-  WR Tabs
-  WR Text properties
-  WR Text properties values

WR Area properties

定数	型	値	コメント
wr allow drag	倍長 整数	14	areaからのドラッグ許可設定 (0=ドラッグを許可しない, 1=ドラッグを許可する)
wr allow drop	倍長 整数	15	areaへのドロップ許可設定 (0=ドロップを許可しない, 1=ドロップを許可する)
wr allow undo	倍長 整数	2	アクションのバッファを設定: <u>wr no undo</u> (0) = アクションを記憶しない, <u>wr undo allowed</u> (1) = アクションを記憶する
wr confirm dialog	倍長 整数	0	確認ダイアログボックスの表示ステータス設定: <u>wr no dialog</u> (0), <u>wr display dialog</u> (1)
wr convert by token	倍長 整数	12	ドキュメント変換時のフィールド参照の解釈設定: <u>wr convert by names</u> (0), <u>wr convert by numbers</u> (1)
wr convert dialog	倍長 整数	5	4D Write 6.0フィールド変換ダイアログの表示ステータス設定 — area = 0の場合: <u>wr no dialog</u> (0), <u>wr display dialog</u> (1)
wr fixed print size	倍長 整数	4	可変サイズのプリントステータス設定 — area = 0の場合を除く: <u>wr var size printing status</u> (0), <u>wr fixed size printing status</u> (1)
wr load template on server	倍長 整数	11	クライアント/サーバーモードでのテンプレートの検索先設定: <u>wr on client</u> (0), <u>wr on server</u> (1)
wr minimized button title	倍長 整数	6	areaが最小化されているときのボタンタイトル設定: <u>wr area name</u> (0), <u>wr custom title</u> (1) はstringValueに渡された文字列
wr minimum height	倍長 整数	9	ボタンにスイッチする前のareaの最小高さ設定 (ピクセル単位)
wr minimum width	倍長 整数	8	ボタンにスイッチする前のareaの最小幅設定 (ピクセル単位)
wr modified	倍長 整数	3	更新ビットステータス設定 — area = 0の場合を除く: <u>wr dirty bit status false</u> (0), <u>wr dirty bit status true</u> (1)
wr on the fly spellchecking	倍長 整数	16	タイプ時にスペルチェックを行うかどうかの設定 (0=行う, 1=行わない)
wr save preview	倍長 整数	1	ピクチャレビュー作成設定: <u>wr no picture preview</u> (0), <u>wr picture preview creation</u> (1)
wr save template on server	倍長 整数	10	クライアント/サーバーモードでのテンプレート保存先設定: <u>wr on client</u> (0), <u>wr on server</u> (1)
wr timer frequency	倍長 整数	17	<u>wr on timer</u> イベントを生成する間隔設定 (value=tick単位の呼び出し間隔 — 1 tick = 1/60秒 — デフォルトは3600 tick)
wr use saved zoom value	倍長 整数	18	エリアが最後に閉じられたときに保存される、エリアを開くときのズーム値: <u>wr use default zoom</u> (0) = 100 %, <u>wr use saved zoom</u> (1)
wr window title	倍長 整数	7	フルスクリーンやプラグインウィンドウでの4D Writeウィンドウのタイトル (0=エリア名, 1=stringValueに渡されたカスタムタイトル)
wr zoom factor	倍長 整数	13	areaのズーム設定 (value=25%から500%の間)

定数	型	値	コメント
wr cmd 1.5 line space	倍長整数	722	
wr cmd about	倍長整数	10	
wr cmd align center	倍長整数	712	
wr cmd align left	倍長整数	711	
wr cmd align right	倍長整数	713	
wr cmd all borders	倍長整数	1009	
wr cmd auto striketh color	倍長整数	632	
wr cmd auto underline color	倍長整数	646	
wr cmd black back	倍長整数	626	
wr cmd black border	倍長整数	686	
wr cmd black border back	倍長整数	682	
wr cmd black circle bullet	倍長整数	1024	
wr cmd black square bullet	倍長整数	1022	
wr cmd black striketh	倍長整数	633	
wr cmd black text	倍長整数	602	
wr cmd black underline	倍長整数	647	
wr cmd blue border	倍長整数	691	
wr cmd blue striketh	倍長整数	638	
wr cmd blue text	倍長整数	607	
wr cmd blue underline	倍長整数	652	
wr cmd bold	倍長整数	502	
wr cmd borders	倍長整数	754	
wr cmd borders inside	倍長整数	1010	
wr cmd bottom border	倍長整数	1016	
wr cmd capitals	倍長整数	508	
wr cmd centered tab	倍長整数	1032	
wr cmd change case submenu	倍長整数	220	
wr cmd character	倍長整数	751	
wr cmd clear	倍長整数	6	
wr cmd clubs bullet	倍長整数	1027	
wr cmd colors menu	倍長整数	600	
wr cmd columns	倍長整数	756	
wr cmd compute references	倍長整数	803	
wr cmd copy	倍長整数	4	
wr cmd copy ruler	倍長整数	701	
wr cmd cut	倍長整数	3	
wr cmd dark grey back	倍長整数	625	
wr cmd dark grey border	倍長整数	696	
wr cmd dark grey border back	倍長整数	681	
wr cmd dark grey shadow	倍長整数	664	
wr cmd dark grey striketh	倍長整数	643	
wr cmd dark grey text	倍長整数	612	
wr cmd dark grey underline	倍長整数	657	
wr cmd decimal tab	倍長整数	1034	
wr cmd diamonds bullet	倍長整数	1026	
wr cmd doc information	倍長整数	801	
wr cmd doc statistics	倍長整数	802	
wr cmd double spaced	倍長整数	723	
wr cmd double underline	倍長整数	525	
wr cmd edit menu	倍長整数	200	
wr cmd file menu	倍長整数	100	

wr cmd find	倍長整数	208
wr cmd find next	倍長整数	209
wr cmd font dropdown	倍長整数	1002
wr cmd format menu	倍長整数	750
wr cmd freeze references	倍長整数	804
wr cmd full justification	倍長整数	714
wr cmd goto full window	倍長整数	20
wr cmd goto page	倍長整数	807
wr cmd green border	倍長整数	690
wr cmd green striketh	倍長整数	637
wr cmd green text	倍長整数	606
wr cmd green underline	倍長整数	651
wr cmd hatched underline	倍長整数	530
wr cmd help	倍長整数	11
wr cmd insert 4D expression	倍長整数	407
wr cmd insert column break	倍長整数	410
wr cmd insert current date	倍長整数	412
wr cmd insert current hour	倍長整数	411
wr cmd insert date and time	倍長整数	401
wr cmd insert HTML expression	倍長整数	414
wr cmd insert hyperlink	倍長整数	413
wr cmd insert menu	倍長整数	400
wr cmd insert non break space	倍長整数	405
wr cmd insert page break	倍長整数	406
wr cmd insert page number	倍長整数	402
wr cmd insert soft hyphen	倍長整数	404
wr cmd insert special char	倍長整数	409
wr cmd inside bottom border	倍長整数	1008
wr cmd inside top border	倍長整数	1006
wr cmd italic	倍長整数	503
wr cmd language	倍長整数	806
wr cmd left border	倍長整数	1005
wr cmd left tab	倍長整数	1031
wr cmd lgt blue border back	倍長整数	677
wr cmd lgt green border back	倍長整数	676
wr cmd lgt grey border back	倍長整数	679
wr cmd lgt orange border back	倍長整数	674
wr cmd lgt red border back	倍長整数	673
wr cmd lgt violet border back	倍長整数	678
wr cmd lgt yellow border back	倍長整数	675
wr cmd light blue back	倍長整数	621
wr cmd light green back	倍長整数	620
wr cmd light grey back	倍長整数	623
wr cmd light grey border	倍長整数	694
wr cmd light grey shadow	倍長整数	662
wr cmd light grey striketh	倍長整数	641
wr cmd light grey text	倍長整数	610
wr cmd light grey underline	倍長整数	655
wr cmd light orange back	倍長整数	618
wr cmd light red back	倍長整数	617
wr cmd light violet back	倍長整数	622
wr cmd light yellow back	倍長整数	619

wr cmd lower case	倍長整数	221
wr cmd med grey border back	倍長整数	680
wr cmd medium grey back	倍長整数	624
wr cmd medium grey border	倍長整数	695
wr cmd medium grey shadow	倍長整数	663
wr cmd medium grey striketh	倍長整数	642
wr cmd medium grey text	倍長整数	611
wr cmd medium grey underline	倍長整数	656
wr cmd new	倍長整数	101
wr cmd no back color	倍長整数	628
wr cmd no border back color	倍長整数	684
wr cmd no borders	倍長整数	1011
wr cmd no bullet	倍長整数	1021
wr cmd no underline	倍長整数	522
wr cmd open	倍長整数	102
wr cmd orange border	倍長整数	688
wr cmd orange striketh	倍長整数	635
wr cmd orange text	倍長整数	604
wr cmd orange underline	倍長整数	649
wr cmd other back color	倍長整数	627
wr cmd other border back color	倍長整数	683
wr cmd other border color	倍長整数	697
wr cmd other bullet	倍長整数	1028
wr cmd other line spacing	倍長整数	724
wr cmd other shadow color	倍長整数	665
wr cmd other striketh color	倍長整数	644
wr cmd other text color	倍長整数	613
wr cmd other underline color	倍長整数	658
wr cmd page setup	倍長整数	106
wr cmd paragraph	倍長整数	752
wr cmd paragraph menu	倍長整数	700
wr cmd paste	倍長整数	5
wr cmd paste ruler	倍長整数	702
wr cmd plain	倍長整数	501
wr cmd preferences	倍長整数	105
wr cmd print	倍長整数	108
wr cmd print merge	倍長整数	109
wr cmd print preview	倍長整数	107
wr cmd red border	倍長整数	687
wr cmd red striketh	倍長整数	634
wr cmd red text	倍長整数	603
wr cmd red underline	倍長整数	648
wr cmd redo	倍長整数	2
wr cmd replace	倍長整数	210
wr cmd replace all	倍長整数	212
wr cmd replace next	倍長整数	211
wr cmd right border	倍長整数	1007
wr cmd right tab	倍長整数	1033
wr cmd save	倍長整数	103
wr cmd save as	倍長整数	104
wr cmd save as template	倍長整数	110
wr cmd select all	倍長整数	7

wr cmd shadow	倍長整数	504
wr cmd show selection	倍長整数	309
wr cmd single spaced	倍長整数	721
wr cmd single underline	倍長整数	523
wr cmd size dropdown	倍長整数	1001
wr cmd small capitals	倍長整数	509
wr cmd spellcheck	倍長整数	805
wr cmd standard bullet	倍長整数	1012
wr cmd status bar	倍長整数	320
wr cmd strikethrough	倍長整数	505
wr cmd style menu	倍長整数	500
wr cmd stylesheet dropdown	倍長整数	1000
wr cmd stylesheets	倍長整数	755
wr cmd subscript	倍長整数	507
wr cmd superscript	倍長整数	506
wr cmd table wizard	倍長整数	408
wr cmd tabs	倍長整数	753
wr cmd title case	倍長整数	223
wr cmd toggle case	倍長整数	224
wr cmd toolbars submenu	倍長整数	330
wr cmd tools menu	倍長整数	800
wr cmd top border	倍長整数	1015
wr cmd underline button	倍長整数	521
wr cmd undo	倍長整数	1
wr cmd upper case	倍長整数	222
wr cmd vertical separator	倍長整数	1035
wr cmd view borders toolbar	倍長整数	334
wr cmd view footer	倍長整数	313
wr cmd view format toolbar	倍長整数	332
wr cmd view frames	倍長整数	317
wr cmd view header	倍長整数	312
wr cmd view HScrollbar	倍長整数	318
wr cmd view invisibles	倍長整数	316
wr cmd view menu	倍長整数	300
wr cmd view menubar	倍長整数	310
wr cmd view normal	倍長整数	302
wr cmd view page	倍長整数	303
wr cmd view pictures	倍長整数	315
wr cmd view references	倍長整数	314
wr cmd view ruler	倍長整数	311
wr cmd view standard toolbar	倍長整数	331
wr cmd view style toolbar	倍長整数	333
wr cmd view VScrollbar	倍長整数	319
wr cmd violet border	倍長整数	692
wr cmd violet striketh	倍長整数	639
wr cmd violet text	倍長整数	608
wr cmd violet underline	倍長整数	653
wr cmd white back	倍長整数	616
wr cmd white border	倍長整数	693
wr cmd white border back	倍長整数	672
wr cmd white circle bullet	倍長整数	1025
wr cmd white square bullet	倍長整数	1023

wr cmd white striketh	倍長整数	640
wr cmd white text	倍長整数	609
wr cmd white underline	倍長整数	654
wr cmd word underline	倍長整数	524
wr cmd yellow border	倍長整数	689
wr cmd yellow striketh	倍長整数	636
wr cmd yellow text	倍長整数	605
wr cmd yellow underline	倍長整数	650

WR Count

定数	型	値	コメント
wr nb characters	倍長整数	0	
wr nb column breaks	倍長整数	7	
wr nb HTML expressions	倍長整数	16	
wr nb hyperlinks	倍長整数	14	
wr nb insertions date time	倍長整数	8	
wr nb insertions page number	倍長整数	9	
wr nb lines	倍長整数	10	
wr nb objects	倍長整数	4	
wr nb page breaks	倍長整数	6	
wr nb pages	倍長整数	11	
wr nb paragraphs	倍長整数	2	
wr nb pictures in page	倍長整数	13	
wr nb pictures in text flow	倍長整数	3	
wr nb RTF expressions	倍長整数	15	
wr nb soft hyphens	倍長整数	5	
wr nb stylesheets	倍長整数	12	
wr nb words	倍長整数	1	

WR Document properties

コメント

(*) プログラムで用紙サイズを設定した場合、4D Writeは仮想プリンタデバイスが使用されるものと見なします。プログラムはデッドマージンを0に設定し、印刷可能なエリアは用紙サイズと等しくなります。この動作は印刷するつもりのないドキュメントで有用です。

定数	型	値	コメント
wr binding	倍 長 整 数	26	カレントのドキュメント単位で表したバインドサイズの設定、環境設定ダイアログボックスの"バインド"エリアに対応。
wr column width	倍 長 整 数	59	現在のドキュメント単位で表した段落幅 (この値は読み込みのみ可能です)。
wr columns spacing	倍 長 整 数	25	現在のドキュメント単位で表した、段落間のスペース、段落ダイアログボックスの"スペース"に対応。
wr data size	倍 長 整 数	43	バイト単位でドキュメントのサイズを取得 (この値は読み込みのみ可能です)。
wr dead left margin	倍 長 整 数	39	用紙左側の、プリンターによって予約されている非印刷領域のサイズを現在のドキュメント単位で取得 (この値は読み込みのみ可能です) (*)
wr dead top margin	倍 長 整 数	40	用紙上側の、プリンターによって予約されている非印刷領域のサイズを現在のドキュメント単位で取得 (この値は読み込みのみ可能です) (*)
wr default tab	倍 長 整 数	22	現在のドキュメント単位で表されたデフォルトの"自動"タブスペース - 環境設定の'デフォルトタブスペース'エリアに対応 (デフォルトで0.5インチ、1.3cm、36ピルセル)
wr different left right pages	倍 長 整 数	19	左と右のページでヘッダ/フッタが異なるかを指定 - 環境設定ダイアログボックスの'左と右ページで異なる'オプションに対応: wr similar (0) または wr different (1)
wr different on first page	倍 長 整 数	18	最初のページのヘッダ/フッタが異なるかを指定 - 環境設定ダイアログボックスの'左と右ページで異なる'オプションに対応: wr similar (0) または wr different (1)
wr draft mode	倍 長 整 数	58	ドキュメントのテキスト入力モード設定: wr wysiwyg (0) また wr draft (1)
wr first page	倍 長 整 数	0	最初のページ番号設定 (デフォルトで1)。例えば10に設定すると次のページは11になる。
wr first page bottom margin	倍 長 整 数	53	最初のページのボディと下端のマージンを現在のドキュメントの単位で設定、他のページには' wr text bottom margin 'を使用。
wr first page top margin	倍 長 整 数	52	最初のページのボディ上端と用紙上端のマージンを現在のドキュメントの単位で設定、他のページには' wr text top margin 'を使用。
wr footer 1st page bottom mg	倍 長 整 数	57	最初のページの下端と用紙下端とのマージンを現在の用紙の単位で設定、他のページには' wr footer bottom margin 'を使用
wr footer	倍		

1st page top margin	長 整 数	56	最初のページのフッタと用紙下端のマージンを現在のドキュメントの単位で設定、他のページには' wr footer top margin 'を使用。
wr footer bottom margin	倍 長 整 数	36	ページフッタの下端と用紙下端のマージンを現在のドキュメントの単位で設定、先頭ページ型のページと異なる場合' wr footer 1st page bottom mg 'を使用。
wr footer top margin	倍 長 整 数	35	ページフッタの上端と用紙下端間のマージンを現在の用紙の単位で設定、最初のページが他と異なる場合は' wr footer 1st page top margin 'を使用。
wr header 1st page bottom mg	倍 長 整 数	55	先頭ページヘッダの下端と用紙上端間のマージンを現在の用紙の単位で設定、他のページには' wr header bottom margin 'を使用。
wr header 1st page top margin	倍 長 整 数	54	先頭ページのヘッダの上端とページの上端間のマージンを現在のドキュメントの単位で設定、他のページには' wr header top margin 'を使用。
wr header bottom margin	倍 長 整 数	34	ページヘッダの下端と用紙上端間のマージンを現在の用紙の単位で設定、先頭ページが他のページと異なる場合' wr header 1st page bottom mg 'を使用。
wr header top margin	倍 長 整 数	33	ページヘッダの上端と用紙上端間のマージンを現在の用紙の単位で設定、先頭ページが他のページと異なる場合' wr header 1st page top margin 'を使用。
wr horizontal splitter	倍 長 整 数	45	水平スプリッターの表示ステータス: wr hidden (0) または wr displayed (1)
wr language	倍 長 整 数	23	ドキュメントに割り当てる言語設定 (American English = 1033, Australian English = 3081, English = 2057, Catalan = 1027, Danish = 1030, Dutch = 1043, Finnish = 1035, French = 1036, French Canadian = 3084, German = 1031, Italian = 1040, Norwegian Bokmal = 1044, Norwegian Nynorsk = 2068, Portuguese Brazil = 1046, Portuguese Iberian = 2070, Spanish = 1034, Swedish = 1053, Russian = 1049, Czech = 1029, Hungarian = 1038, Polish = 1045)
wr links color	倍 長 整 数	47	未訪問のハイパーリンクのカラー設定
wr number of columns	倍 長 整 数	24	ドキュメントの段組み数設定Gets or sets the number of columns of the document
wr opposite pages	倍 長 整 数	27	ドキュメントの両面モード設定 - 環境設定ダイアログボックスの'両面ページ'オプションに対応: wr single sided pages (0) または wr double sided pages (1)
wr paper height	倍 長 整 数	38	用紙の高さを現在のドキュメントの単位で設定 (*)
wr paper width	倍 長 整 数	37	用紙の幅を現在のドキュメントの単位で設定 (*)
wr printable	倍 長	42	左上マージンから開始して、縦方向の印刷可能領域を取得 (この値は読み込みのみ可能)。The

printable height	整数	42	bottom dead margin equals the paper height; the top dead margin-the printable height.
wr printable width	倍長整数	41	左上マージンから開始して、横方向の印刷可能領域を取得 (この値は読み込みのみ可能)。The right dead margin equals the paper width; the left dead margin-the printable width.
wr right first page	倍長整数	28	先頭ページが右か左かを設定 - デフォルトで右ページ: <code>wr left page (0)</code> または <code>wr right page (1)</code>
wr text bottom margin	倍長整数	32	ページボディ下端と用紙下端のマージンを現在のドキュメントの単位で設定、最初のページだけに別の設定を行うには' <code>wr first page bottom margin</code> 'を使用。
wr text inside margin	倍長整数	29	テキストの左側と、右ページの場合用紙の左側間、左ページの場合右側間のマージンを、現在のドキュメントの単位で設定 (ページモードで使用)。
wr text left margin	倍長整数	29	ページの左側と用紙の左側間のマージンを、現在のドキュメントの単位で設定 (通常モードで使用)。
wr text outside margin	倍長整数	30	テキストの右側と、右ページの場合用紙の右側間、左ページの場合左側間のマージンを、現在のドキュメントの単位で設定 (ページモードで使用)。
wr text right margin	倍長整数	30	ページの右側用紙の右側間のマージンを、現在のドキュメントの単位で設定 (通常モードで使用)。
wr text top margin	倍長整数	31	Gets or sets the margin between the top of the page body and the top edge of the paper expressed in the current document unit, use ' <code>wr first page top margin</code> ' for the first page if different from others
wr undo buffer size	倍長整数	44	取り消しバッファをバイト単位で設定 (この値は読み込みのみ可能)。
wr unit	倍長整数	21	ドキュメントの現在の単位を設定 - 環境設定ダイアログボックスの'単位'ポップアップメニューに対応: <code>wr centimeters (0)</code> , <code>wr inches (1)</code> または <code>wr pixels (2)</code>
wr vertical splitter	倍長整数	46	縦スプリッターの表示設定: <code>wr hidden (0)</code> または <code>wr displayed (1)</code>
wr view borders palette	倍長整数	14	罫線ツールバーの表示ステータス設定: <code>wr hidden (0)</code> or <code>wr displayed (1)</code>
wr view column separators	倍長整数	17	複数段組みモードに置いて、段組み間の縦分割線の表示を設定 - 段組みダイアログボックスの縦分割オプションに対応: <code>wr hidden (表示しない) (0)</code> または <code>wr displayed (表示する) (1)</code>
wr view first page footer	倍長整数	51	先頭ページのフッタの表示ステータス設定: <code>wr hidden (0)</code> または <code>wr displayed (1)</code> , 他のページには' <code>wr view footers</code> 'を使用。

wr view first page header	10 長 整 数 倍	50	先頭ページのヘッダの表示ステータス設定: <u>wr hidden</u> (0) または <u>wr displayed</u> (1), 他のページには' <u>wr view headers</u> 'を使用。
wr view footers	長 整 数 倍	5	フッタの表示ステータス設定: <u>wr hidden</u> (0) または <u>wr displayed</u> (1), 先頭ページが他のページと異なる場合' <u>wr view first page footer</u> 'を使用。
wr view format palette	長 整 数 倍	12	フォーマットツールバーの表示ステータス: <u>wr hidden</u> (0) または <u>wr displayed</u> (1)
wr view frame area	長 整 数 倍	49	フォーム中の領域の外周に表示されるフレームの設定: <u>wr hidden</u> (フレームなし) (0) または <u>wr displayed</u> (フレームあり)(1)
wr view frames	長 整 数 倍	3	テキストフレームの表示ステータス: <u>wr hidden</u> (0) または <u>wr displayed</u> (1)
wr view headers	長 整 数 倍	4	ヘッダの表示ステータス: <u>wr hidden</u> (0) or <u>wr displayed</u> (1), 先頭ページが他のページと異なる場合は' <u>wr view first page header</u> 'を使用)。
wr view Hscrollbar	長 整 数 倍	7	横スクロールバーの表示ステータス設定: <u>wr hidden</u> (0) または <u>wr displayed</u> (1)
wr view invisible chars	長 整 数 倍	15	非表示文字の表示ステータス設定: <u>wr hidden</u> (0) または <u>wr displayed</u> (1)
wr view menubar	長 整 数 倍	10	メニューバーの表示ステータス設定: <u>wr hidden</u> (0) または <u>wr displayed</u> (1)
wr view mode	長 整 数 倍	1	ドキュメント表示モードの設定: <u>wr page mode</u> (0) または <u>wr normal mode</u> (1)
wr view pictures	長 整 数 倍	6	ピクチャの表示ステータス設定: <u>wr hidden</u> (0) または <u>wr displayed</u> (1)
wr view references	長 整 数 倍	16	参照の表示ステータス設定: <u>wr hidden</u> (0) または <u>wr displayed</u> (1)
wr view rulers	長 整 数 倍	2	ルーラの表示ステータス設定: <u>wr hidden</u> (0) または <u>wr displayed</u> (1)
wr view standard palette	長 整 数 倍	11	標準ツールパレットの表示ステータス設定: <u>wr hidden</u> (0) または <u>wr displayed</u> (1)
wr view statusbar	長 整 数 倍	9	ステータスバーの表示ステータス設定: <u>wr hidden</u> (0) または <u>wr displayed</u> (1)

wr view style palette	数 倍 長 整 数	13	スタイルツールバーの表示ステータス設定: <u>wr hidden</u> (0) または <u>wr displayed</u> (1)
wr view Vscrollbar	数 倍 長 整 数	8	縦スクロールバーの表示ステータス設定: <u>wr hidden</u> (0) または <u>wr displayed</u> (1)
wr visited links color	数 倍 長 整 数	48	訪問済みハイパーリンクのカラー設定
wr widow orphan	数 倍 長 整 数	20	Gets or sets whether widows and orphans are taken into account - corresponds to the 'Widow and Orphan Control' option in the Preferences dialog box: <u>wr ignored</u> (0) or <u>wr managed</u> (1)

WR Document types

定数	型	値	コメント
wr 4D Write document	文字列	4WR7	4D Writeカレントバージョンフォーマットのドキュメント
wr 4D Write template	文字列	4WT7	4D Writeテンプレートフォーマットドキュメント
wr HTML 3 document	文字列	HTM3	HTML 3.0フォーマットテキスト
wr HTML 4 document	文字列	HTML	HTML 4.0フォーマットテキスト
wr Macintosh text document	文字列	ASCM	Mac OSフォーマットテキスト
wr RTF document	文字列	RTF	RTFフォーマットテキスト
wr unicode document UTF16	文字列	ASCU	Unicode 16-byteフォーマットテキスト
wr unicode document UTF8	文字列	ASC8	Unicode 8-byteフォーマットテキスト
wr Windows text document	文字列	ASCW	Windowsフォーマットテキスト

WR Events

定数	型	値	コメント
wr on activate	倍長整数	5	4D Writeエリアがアクティブまたは非アクティブになった
wr on close	倍長整数	10	4D Writeエリアまたはウィンドウが閉じられた
wr on compute references	倍長整数	9	動的参照が更新された
wr on double click	倍長整数	1	ダブルクリック
wr on drag	倍長整数	11	オブジェクトがドラッグされた
wr on drop	倍長整数	12	オブジェクトがドロップされた
wr on key	倍長整数	0	キーが押された (矢印キーやリターン等を含む)
wr on printing	倍長整数	7	ドキュメント印刷中
wr on right click	倍長整数	4	右マウスボタンのクリック
wr on ruler	倍長整数	8	ルーラの変更
wr on single click	倍長整数	2	シングルクリック
wr on timer	倍長整数	13	タイマーサイクルの終了
wr on triple click	倍長整数	3	3回クリック

WR Frames

定数	型	値	コメント
wr first footer	倍長整数	6	
wr first header	倍長整数	5	
wr left footer	倍長整数	4	
wr left header	倍長整数	3	
wr right footer	倍長整数	2	
wr right header	倍長整数	1	
wr text frame	倍長整数	0	

WR Page number formats

定数	型	値	コメント
wr 123	倍長整数	0	1, 2, 3...
wr abc	倍長整数	1	a, b, c...
wr ABC	倍長整数	2	A, B, C...
wr i ii iii	倍長整数	3	i, ii, iii...
wr I II III	倍長整数	4	I, II, III...

WR Parameters

定数	型	値	コメント
wr above text	倍 長 整 数	0	ピクチャをテキストの上に挿入する
wr after insertion point	倍 長 整 数	0	検索は挿入位置からドキュメントの最後まで行われます。
wr allowed access	倍 長 整 数	0	エリアにフルアクセスがあります。
wr apply to characters	倍 長 整 数	0	文字にスタイルシートを適用する。
wr apply to paragraphs	倍 長 整 数	1	段落にスタイルシートを適用する。
wr area name	倍 長 整 数	0	エリアのデフォルト名
wr at end of document	倍 長 整 数	1	テキストはドキュメントの最後に挿入される。
wr at insertion point	倍 長 整 数	0	テキストは現在の挿入位置に挿入される。
wr behind text	倍 長 整 数	1	ピクチャはテキストの後ろに挿入される。この場合、テキストと背景の属性に注意を払う必要があります。"なし"を選択するとテキストの後ろのピクチャを表示することができます。
wr black and white	倍 長 整 数	1	白黒カラーオプション
wr case sensitive	倍 長 整 数	1	検索は大文字と小文字を区別して行われます。
wr centimeters	倍 長 整 数	0	
wr checking off	倍 長 整 数	0	
wr checking on	倍 長 整 数	1	

wr color	長 整 数	2	
wr convert by names	倍 長 整 数	0	
wr convert by numbers	倍 長 整 数	1	
wr custom link appearance	倍 長 整 数	0	カスタマイズされたアピランスの利用を許可します。この場合リンクを選択し、 <i>WR SET TEXT PROPERTY</i> コマンドを使用してスタイルを定義できます。
wr custom title	倍 長 整 数	1	
wr default link appearance	倍 長 整 数	1	デフォルトのハイパーリンクアピランス (青色および下線) を保持します。デフォルトのカラーは <i>WR SET DOC PROPERTY</i> コマンドを使用してプログラムで変更できます。
wr different	倍 長 整 数	1	
wr dirty bit status false	倍 長 整 数	0	
wr dirty bit status true	倍 長 整 数	1	
wr display dialog	倍 長 整 数	1	
wr displayed	倍 長 整 数	1	
wr document type link	倍 長 整 数	2	ドキュメントタイプのリンクを挿入
wr double sided	倍 長 整 数	1	
wr double sided pages	倍 長 整 数	1	
wr draft	倍 長 整	1	

wr drag allowed	数 倍 長 整 数	1	
wr drag not allowed	数 倍 長 整 数	0	
wr drop allowed	数 倍 長 整 数	1	
wr drop not allowed	数 倍 長 整 数	0	
wr enabled command	数 倍 長 整 数	0	コマンドは呼び出されたときに実行されます。
wr fixed size printing status	数 倍 長 整 数	1	
wr hidden	数 倍 長 整 数	0	
wr ignore uppercase	数 倍 長 整 数	0	検索は大文字と小文字を区別せずに行われます。
wr ignored	数 倍 長 整 数	0	
wr inches	数 倍 長 整 数	1	
wr into the text flow	数 倍 長 整 数	0	ピクチャはテキストフロー中に挿入されます。この場合他の引数は使用されず、ピクチャは挿入ポイントに挿入されるか現在選択されている箇所を置き換えます。
wr landscape	数 倍 長 整 数	2	
wr layout and print settings	数 倍 長 整 数	0	プリントおよび用紙設定が使用されます。
wr left binding	数 倍 長 整 数	0	

wr left page	調整 数	0	
wr locked command	長 整数	1	コマンドは呼び出し時に実行されず、メニューは使用不可になります。
wr locked document	長 整数	1	ドキュメントはロックされます。
wr managed	長 整数	1	
wr method type link	長 整数	0	メソッドタイプのリンクを挿入。
wr no date format	長 整数	0	日付フォーマットなし
wr no dialog	長 整数	0	
wr no picture preview	長 整数	0	
wr no print settings dialog	長 整数	0	印刷設定ダイアログボックスを表示しない。
wr no time format	長 整数	0	時間フォーマットなし
wr no undo	長 整数	0	アクションを記憶しない
wr normal mode	長 整数	1	
wr on client	長 整数	0	
wr on current page	長 整数	-4	ピクチャはページ上に挿入され、現在のページ (挿入ポイントあるいは現在選択されている箇所) に表示されます。
wr on left hand pages	長 整数	-12	ピクチャはページ上に挿入され、偶数と奇数ページのヘッダが異なる場合、左側のページにのみ表示されます。

wr on right hand pages	倍 長 整 数	11	- ピクチャはページ上に挿入され、偶数と奇数ページのヘッダが異なる場合、右側のページにのみ表示されます。そうでなければすべてのページに表示されます。
wr on server	倍 長 整 数	1	
wr page mode	倍 長 整 数	0	
wr page number	倍 長 整 数	0	
wr partial match	倍 長 整 数	0	文字列は単語全体あるいはより長い単語の一部です。
wr picture preview creation	倍 長 整 数	1	
wr pixels	倍 長 整 数	2	
wr portrait	倍 長 整 数	1	
wr print references	倍 長 整 数	1	
wr print settings only	倍 長 整 数	1	プリント設定のみが使用されます。
wr print values	倍 長 整 数	0	
wr replace all	倍 長 整 数	1	単語のすべてのオカレンスが置換されます。
wr replace next	倍 長 整 数	0	単語の次のオカレンスのみが置換されます。
wr restricted access	倍 長 整 数	1	ユーザはエリア情報に読み込みのみでアクセスできます。
wr right page	倍 長	1	

wr night page	整数	1	
wr screen updating off	倍長整数	0	スクリーンの更新を無効にする。
wr screen updating on	倍長整数	1	スクリーンの更新を有効にする。
wr send to file	倍長整数	2	
wr send to PDF file	倍長整数	3	
wr send to printer	倍長整数	1	
wr similar	倍長整数	0	
wr single sided	倍長整数	0	
wr single sided pages	倍長整数	0	
wr top binding	倍長整数	1	
wr total number of pages	倍長整数	1	
wr undo allowed	倍長整数	1	アクションを記憶する。
wr unlocked document	倍長整数	0	ドキュメントはアンロックされます。
wr URL type link	倍長整数	1	URLタイプのリンクを挿入
wr use default zoom	倍長整数	0	

wr use saved zoom	倍 長 整 数	1	
wr var size printing status	倍 長 整 数	0	
wr whole document	倍 長 整 数	1	検索は挿入ポイントからドキュメントの終わりまで実行され、さらに先頭から挿入ポイントまで検索されます。
wr whole word	倍 長 整 数	1	単語は区切り文字 (スペース、句読点等) の間になければなりません。
wr with print settings dialog	倍 長 整 数	1	印刷設定ダイアログが表示されます。
wr wysiwyg	倍 長 整 数	0	

WR Print options

定数	型	値	コメント
wr color option	倍長整数	8	
wr destination option	倍長整数	9	
wr double sided option	倍長整数	11	
wr number of copies option	倍長整数	4	
wr orientation option	倍長整数	2	
wr pages from option	倍長整数	6	
wr pages to option	倍長整数	7	
wr paper option	倍長整数	1	
wr paper source option	倍長整数	5	
wr scale option	倍長整数	3	
wr spooler document name option	倍長整数	12	

 **WR Select type**

定数	型	値	コメント
wr select characters	倍長整数	0	Selects the characters located between <i>begin</i> と <i>end</i> の間に位置する文字を選択。この場合WR SET SELECTIONを使用するのと同じ。
wr select column break	倍長整数	8	ドキュメント中で <i>begin</i> で指定されるランクである段組みブレイクを選択。 <i>end</i> は省略しなければなりません。
wr select date and time	倍長整数	11	ドキュメント中で <i>begin</i> により指定されるランクの日付および時間変数を選択。 <i>end</i> は省略しなければなりません。選択は自動でボディテキストに更新・挿入される日付と時刻にのみ繰り越されません。
wr select expression	倍長整数	1	ドキュメント中で <i>begin</i> により指定されるランクの参照を選択。 <i>end</i> は省略しなければなりません。
wr select HTML expression	倍長整数	13	ドキュメント中で <i>begin</i> により指定されるランクのHTML式を選択。 <i>end</i> は省略しなければなりません。
wr select hyperlink	倍長整数	12	ドキュメント中で <i>begin</i> により指定されるランクのハイパーリンクを選択。 <i>end</i> は省略しなければなりません。
wr select hyphen	倍長整数	9	ドキュメント中で <i>begin</i> により指定されるランクのハイフンを選択。 <i>end</i> は省略しなければなりません。
wr select page break	倍長整数	7	ドキュメント中で <i>begin</i> により指定されるランクの改ページを選択。 <i>end</i> は省略しなければなりません。
wr select page number	倍長整数	10	ドキュメント中で <i>begin</i> により指定されるランクのページ番号を選択。 <i>end</i> は省略しなければなりません。選択はテキストのボディに挿入されるページ番号にのみ持ち越されます。
wr select paragraphs	倍長整数	2	<i>begin</i> と <i>end</i> の間に位置する段落を選択します。
wr select picture	倍長整数	4	ドキュメント中で <i>begin</i> により指定されるランクのピクチャを選択。 <i>end</i> は省略しなければなりません。
wr select RTF expression	倍長整数	14	ドキュメント中で <i>begin</i> により指定されるランクのRTF式を選択。 <i>end</i> は省略しなければなりません。
wr select ruler	倍長整数	3	(ドキュメント中でドキュメントの先頭から始まるランクの) X番目のルーラを使用する段落を選択。 <i>end</i> は省略しなければなりません。
wr select style	倍長整数	5	(ドキュメント中でドキュメントの先頭から始まるランクの) X番目のスタイルを使用する単語を選択。 <i>end</i> は省略しなければなりません。

wr select
word

長
整
数

6 挿入ポイントの位置の単語を選択。

WR Standard colors

定数	型	値	コメント
wr automatic	倍長整数	-1	
wr black	倍長整数	0	
wr blue	倍長整数	3381759	
wr dark grey	倍長整数	6710886	
wr green	倍長整数	52249	
wr light blue	倍長整数	11790079	
wr light green	倍長整数	11796403	
wr light grey	倍長整数	13421772	
wr light orange	倍長整数	16767398	
wr light red	倍長整数	16757683	
wr light violet	倍長整数	16761087	
wr light yellow	倍長整数	16777164	
wr medium grey	倍長整数	10066329	
wr orange	倍長整数	16750848	
wr red	倍長整数	16711680	
wr violet	倍長整数	13369599	
wr white	倍長整数	16777215	
wr yellow	倍長整数	16770560	

WR Tabs

定数	型	値	コメント
wr centered tab	倍長整数	2	中央合わせ
wr decimal tab	倍長整数	4	小数点
wr left tab	倍長整数	1	左揃え
wr right tab	倍長整数	3	右揃え
wr vertical separator tab	倍長整数	5	縦分割

WR Text properties

定数	型	値	コメント
wr bold	倍長整数	0	
wr border back color	倍長整数	38	
wr border line color	倍長整数	39	
wr border line style	倍長整数	40	
wr border spacing	倍長整数	45	
wr bottom border	倍長整数	47	
wr bullet	倍長整数	34	
wr capital case	倍長整数	6	
wr first indent	倍長整数	36	
wr font number	倍長整数	7	
wr font size	倍長整数	8	
wr inside bottom border	倍長整数	44	
wr inside top border	倍長整数	43	
wr italic	倍長整数	1	
wr justification	倍長整数	32	
wr left border	倍長整数	41	
wr left margin	倍長整数	35	
wr line spacing	倍長整数	33	
wr links appearance	倍長整数	14	
wr right border	倍長整数	42	
wr right margin	倍長整数	37	
wr shadow	倍長整数	2	
wr shadow color	倍長整数	13	
wr strikethrough	倍長整数	3	
wr strikethrough color	倍長整数	11	
wr stylesheet number	倍長整数	15	
wr superscript or subscript	倍長整数	5	
wr tab	倍長整数	64	
wr text back color	倍長整数	10	
wr text color	倍長整数	9	
wr top border	倍長整数	46	
wr underline	倍長整数	4	
wr underline color	倍長整数	12	
wr user property	倍長整数	16	

WR Text properties values

定数	型	値	コメント
wr 1 pt line	倍長整数	0	
wr 2 pts line	倍長整数	1	
wr 3 pts line	倍長整数	2	
wr black circle bullet	倍長整数	108	
wr black square bullet	倍長整数	110	
wr capitals	倍長整数	1	
wr centered	倍長整数	1	
wr clubs bullet	倍長整数	118	
wr diamonds bullet	倍長整数	117	
wr dotted line	倍長整数	3	
wr double 1 pt line	倍長整数	6	
wr double dotted line	倍長整数	4	
wr double inside 2 pts line	倍長整数	7	
wr double outside 2 pts line	倍長整数	9	
wr double underline	倍長整数	3	
wr full justified	倍長整数	3	
wr half pt line	倍長整数	10	
wr hatched underline	倍長整数	4	
wr left justified	倍長整数	0	
wr no bullet	倍長整数	0	
wr no links appearance	倍長整数	0	
wr none	倍長整数	0	
wr quarter pt line	倍長整数	11	
wr right justified	倍長整数	2	
wr single underline	倍長整数	1	
wr small capitals	倍長整数	2	
wr subscript	倍長整数	2	
wr superscript	倍長整数	1	
wr triple center 2 pts line	倍長整数	8	
wr triple dotted line	倍長整数	5	
wr unvisited links appearance	倍長整数	1	
wr visited links appearance	倍長整数	2	
wr white circle bullet	倍長整数	109	
wr white square bullet	倍長整数	111	
wr word underline	倍長整数	2	

目録 付録

- 付録A: ショートカット
- 付録B: メニュー項目番号
- 付録C: エラーコード
- 付録D: 削除されたv6.0.xコマンド
- 付録E: 廃止予定コマンド

特別なキー

スクロールに加え、4D Writeでは以下のキーコンビネーションを使用できます。

Key	Explanation
Home	Moves the insertion point to the beginning of the line
End	Moves the insertion point to the end of the line
Ctrl (or Command) + Home	Moves the insertion point to the beginning of the document
Ctrl (or Command) + End	Moves the insertion point to the end of the document
Page Up	Scrolls one page up (does not modify the current selection)
Page Down	Scrolls one page down (does not modify the current selection)
Enter	Inserts a column break or a page break (depending on the current mode)
Ctrl (or Command) + left arrow	Moves the insertion point to the beginning of the current word or to the beginning of the previous word if the insertion point was already at the beginning of the current word.
Ctrl (or Command) + right arrow	Moves the insertion point to the end of the current word or to the end of the following word if the insertion point was already at the end of the current word
Ctrl (or Command) + up arrow	Moves the insertion point to the beginning of the current paragraph
Ctrl (or Command) + down arrow	Moves the insertion point to the end of the current paragraph
Ctrl (or Command) + Delete	Deletes the next word or the letters located on the right of the cursor.
Ctrl (or Command) + Backspace	Deletes the next word or the letters located on the left of the cursor
Shift (in combination with any of the above keys to move the insertion point or view)	Extends or reduces the current selection

クリックの組み合わせ

4D Writeでは以下のマウスクリックの組み合わせを使用できます:

組み合わせ	説明
シングルクリック	挿入ポイントの移動、選択されたテキストの選択解除
ダブルクリック	ダブルクリックが行われた箇所の単語とその後ろのスペースを選択
トリプルクリック	段落を選択
左マージンのクリック	クリックされた行を選択
左マージンでのダブルクリック	クリックされた段落を選択
Shift+クリック	クリックされた場所まで選択部を拡張
Ctrl+クリック (Mac OSではCommand+クリック)	ページにペーストされたピクチャのしたのテキストを選択
右クリック (Windows)/Control+クリック (Mac)	挿入ポイントにフィールドを挿入するためのポップアップメニューを表示

📌 付録B: メニュー項目番号

以下の表はメニュー項目ごとのコマンド番号を示しています。これらの番号は将来の4D Writeバージョンでメニュー項目が変更されたり移動されたりしても変更されることはありません。詳細は[WR EXECUTE COMMAND](#)コマンドの説明を参照してください。以下のコードは[WR ON COMMAND](#)、[WR LOCK COMMAND](#)、そして[WR GET COMMAND INFO](#)コマンドでも使用されます。

これらのコマンドを使用する際、メニュー項目番号または定数のいずれかを渡せます。定数は[WR Commands](#)テーマにも一覧があります。

メニュー	ツールバー表示	コマンド	#	定数	
ファイル	なし	(メニュー自身)	100	wr cmd file menu	
	あり	新規	101	wr cmd new	
	あり	開く	102	wr cmd open	
	あり	保存	103	wr cmd save	
	なし	別名で保存...	104	wr cmd save as	
	なし	テンプレートとして保存	110	wr cmd save as template	
	なし	環境設定...	105	wr cmd preferences	
	なし	用紙設定...	106	wr cmd page setup	
	あり	プリントプレビュー	107	wr cmd print preview	
	あり	プリント...	108	wr cmd print	
	なし	差し込みプリント...	109	wr cmd print merge	
	なし	フルウィンドウを開く/フォームに戻る	20	wr cmd goto full windows	
	編集	なし	(メニュー自身)	200	wr cmd edit menu
		あり	取り消し機能 (vary)	1	wr cmd undo
あり		やり直し機能 (vary)	2	wr cmd redo	
あり		カット	3	wr cmd cut	
あり		コピー	4	wr cmd copy	
あり		ペースト	5	wr cmd paste	
なし		クリア	6	wr cmd clear	
なし		すべてを選択	7	wr cmd select all	
あり		検索...	208	wr cmd find	
なし		次を検索	209	wr cmd find next	
なし		置換...	210	wr cmd replace	
なし		次を置換	211	wr cmd replace next	
なし		文字変換	220	wr cmd change case submenu	
なし		/ 小文字	221	wr cmd lower case	
なし		/ 大文字	222	wr cmd upper case	
なし		/ 先頭大文字	223	wr cmd title case	
なし		/ 先頭以外大文字	224	wr cmd toggle case	
なし		選択範囲を表示	309	wr cmd show selection	
なし		ページへ移動...	807	wr cmd goto page	
表示		なし	(メニュー自身)	300	wr cmd view menu
	なし	通常	302	wr cmd view normal	
	なし	ページ	303	wr cmd view page	
	なし	ツールバー	330	wr cmd toolbars submenu	
	なし	/ 標準ツールバー	331	wr cmd view standard toolbar	
	なし	/ 書式ツールバー	332	wr cmd view format toolbar	
	なし	/ スタイルツールバー	333	wr cmd view style toolbar	
	なし	/ 罫線ルーラー	334	wr cmd view borders toolbar	
	なし	ルーラ表示	311	wr cmd view ruler	
	なし	ヘッダ表示	312	wr cmd view header	
	なし	フッタ表示	313	wr cmd view footer	
	あり	参照表示	314	wr cmd view references	
	なし	ピクチャ表示	315	wr cmd view pictures	
	あり	非表示文字表示	316	wr cmd view invisibles	
	なし	枠表示	317	wr cmd view frames	
	なし	横スクロールバー表示	318	wr cmd view HScrollbar	
	なし	縦スクロールバー表示	319	wr cmd view VScrollbar	
	なし	メニューバー表示	310	wr cmd view menubar	
	なし	ステータスバー表示	320	wr cmd status bar	
	挿入	なし	(メニュー自身)	400	wr cmd insert menu

	なし	日付と時間...	401	wr cmd insert date and time
	あり	現在時刻	411	wr cmd insert current hour
	あり	現在日付	412	wr cmd insert current date
	なし	ページ番号...	402	wr cmd insert page number
	なし	特殊文字...	409	wr cmd insert special char
	なし	ソフトハイフン	404	wr cmd insert soft hyphen
	なし	ノンブレイクスペース	405	wr cmd insert No break space
	なし	改段組	410	wr cmd insert column break
	なし	ページブレイク	406	wr cmd insert page break
	なし	HTML式 ...	414	wr cmd insert HTML expression
	なし	ハイパーリンク...	413	wr cmd insert hyperlink
	なし	4D式 ...	407	wr cmd insert 4D expression
スタイル	なし	(メニュー自身)	500	wr cmd style menu
	なし	標準	501	wr cmd plain
	あり	太字	502	wr cmd bold
	あり	斜体	503	wr cmd italic
	なし	シャドウ	504	wr cmd shadow
	なし	取り消し線	505	wr cmd strikethrough
	なし	下線	520	
	なし	/ 下線なし	521	wr cmd no underline
	なし	/ 単下線	522	wr cmd continuous underline
	なし	/ 単語下線	523	wr cmd word underline
	なし	/ 二重下線	524	wr cmd double underline
	なし	/ 点下線	525	wr cmd hatched unde
	あり	下線ボタン	530	wr cmd underline button
	なし	上付き	506	wr cmd superscript
	なし	下付き	507	wr cmd subscript
	なし	英大文字	508	wr cmd capitals
	なし	小型英大文字	509	wr cmd small capitals
文字色	なし	(メニュー自身)	600	wr cmd colors menu
		テキスト	601	
		/ 黒	602	wr cmd black text
		/ 赤	603	wr cmd red text
		/ 橙	604	wr cmd orange text
		/ 黄	605	wr cmd yellow text
		/ 緑	606	wr cmd green text
		/ 青	607	wr cmd blue text
		/ すみれ	608	wr cmd violet text
		/ 白	609	wr cmd white text
		/ 薄い灰色	610	wr cmd light grey text
		/ 灰色	611	wr cmd medium grey text
		/ 濃い灰色	612	wr cmd dark grey
		/ その他...	613	wr cmd other text color
		背景色	615	
		/ 背景色なし	628	wr cmd no back color
		/ 白黒背景色	616	wr cmd white back
		/ 薄い赤背景色	617	wr cmd light red back
		/ 薄い橙背景色	618	wr cmd light orange back
		/ 薄い黄背景色	619	wr cmd light yellow back
		/ 薄い緑背景色	620	wr cmd light green back
		/ 薄い青背景色	621	wr cmd light blue back
		/ 薄いすみれ背景色	622	wr cmd light violet back

/ 薄い薄灰色背景色	623	wr cmd light grey back
/ 灰色背景色	624	wr cmd medium grey back
/ 濃い灰色背景色	625	wr cmd dark grey back
/ 黒背景色	626	wr cmd black back
/ その他...	627	wr cmd other back color
取り消し線	631	
/ 自動取り消し線色	632	wr cmd auto striketh color
/ 黒取り消し線	633	wr cmd black striketh
/ 赤取り消し線	634	wr cmd red striketh
/ 橙取り消し線	635	wr cmd orange striketh
/ 黄取り消し線	636	wr cmd yellow striketh
/ 緑取り消し線	637	wr cmd green striketh
/ 青取り消し線	638	wr cmd blue striketh
/ すみれ取り消し線	639	wr cmd violet striketh
/ 白取り消し線	640	wr cmd white striketh
/ 薄い灰色取り消し線	641	wr cmd light grey striketh
/ 灰色取り消し線	642	wr cmd medium grey striketh
/ 濃い灰色取り消し線	643	wr cmd dark grey striketh
/ その他の取り消し線...	644	wr cmd other striketh color
下線	645	
/ 自動下線色	646	wr cmd auto underline color
/ 黒下線	647	wr cmd black underline
/ 赤下線	648	wr cmd red underline
/ 橙下線	649	wr cmd orange underline
/ 黄下線	650	wr cmd yellow underline
/ 緑下線	651	wr cmd green underline
/ 青下線	652	wr cmd blue underline
/ すみれ下線	653	wr cmd violet underline
/ 白下線	654	wr cmd white underline
/ 薄い灰色下線	655	wr cmd light grey underline
/ 灰色下線	656	wr cmd medium grey underline
/ 濃い灰色下線	657	wr cmd dark grey underline
/ その他の下線色...	658	wr cmd other underline color
シャドウ	661	
/ 薄い灰色シャドウ	662	wr cmd light grey shadow
/ 灰色シャドウ	663	wr cmd medium grey shadow
/ 濃い灰色シャドウ	664	wr cmd dark grey shadow
/ その他のシャドウ色...	665	wr cmd other shadow color
段落背景	671	
/ 段落背景色なし	684	wr cmd no border back color
/ 白段落背景色	672	wr cmd white border back
/ 薄い赤段落背景色	673	wr cmd lgt red border back
/ 薄い橙段落背景色	674	wr cmd lgt orange border back
/ 薄い黄段落背景色	675	wr cmd lgt yellow border back
/ 薄い緑段落背景色	676	wr cmd lgt green border back
/ 薄い青段落背景色	677	wr cmd lgt blue border back
/ 薄いすみれ段落背景色	678	wr cmd lgt violet border back
/ 薄い灰色段落背景色	679	wr cmd lgt grey border back
/ 灰色段落背景色	680	wr cmd med grey border back
/ 濃い灰色段落背景色	681	wr cmd dark grey border back
/ 黒段落背景色	682	wr cmd black border back
/ その他の段落背景色...	683	wr cmd other border back color

		罫線	685	
		/ 黒罫線	686	wr cmd black border
		/ 赤罫線	687	wr cmd red border
		/ 橙罫線	688	wr cmd orange border
		/ 黄罫線	689	wr cmd yellow border
		/ 緑罫線	690	wr cmd green border
		/ 青罫線	691	wr cmd blue border
		/ すみれ罫線	692	wr cmd violet border
		/ 白罫線	693	wr cmd white border
		/ 薄い灰色罫線	694	wr cmd light grey border
		/ 灰色罫線	695	wr cmd medium grey border
		/ 濃い灰色	696	wr cmd dark grey border
		/ その他の罫線色...	697	wr cmd other border color
段落	なし	(メニュー自身)	700	wr cmd paragraph menu
	なし	ルーラーコピー	701	wr cmd copy ruler
	なし	ルーラーペースト	702	wr cmd paste ruler
	あり	(箇条書き)	1012	wr cmd standard bullet
	なし	箇条書き	1020	
	なし	/ なし	1021	wr cmd no bullet
	なし	/ 黒四角	1022	wr cmd black square bullet
	なし	/ 白四角	1023	wr cmd white square bullet
	なし	/ 黒丸	1024	wr cmd black circle bullet
	なし	/ 白丸	1025	wr cmd white circle bullet
	なし	/ ダイヤモンド	1026	wr cmd diamonds bullet
	なし	/ クローバー	1027	wr cmd clubs bullet
	なし	/ その他の箇条書き...	1028	wr cmd other bullet
	あり	左揃え	711	wr cmd align left
	あり	中央揃え	712	wr cmd align center
	あり	右揃え	713	wr cmd align right
	あり	均等配置	714	wr cmd full justification
	あり	1行間	721	wr cmd single spaced
	あり	1.5行間	722	wr cmd 1.5 line space
	あり	2行間	723	wr cmd double spaced
	なし	その他の行間	724	wr cmd other line spacing
書式	なし	(メニュー自身)	750	wr cmd format menu
	なし	章...	751	wr cmd character
	なし	段落...	752	wr cmd paragraph
	なし	タブ...	753	wr cmd tabs
	なし	罫線...	754	wr cmd borders
	あり	左罫線	1005	wr cmd left border
	あり	上罫線	1015	wr cmd top border
	あり	右罫線	1007	wr cmd right border
	あり	下罫線	1016	wr cmd bottom border
	あり	内上罫線	1008	wr cmd inside top border
	あり	内下罫線	1006	wr cmd inside bottom border
	あり	全罫線	1009	wr cmd all borders
	あり	内罫線	1010	wr cmd borders inside
	あり	罫線なし	1011	wr cmd no borders
	なし	スタイルシート...	755	wr cmd stylesheets
	なし	段落...	756	wr cmd columns
ツール	なし	(メニュー自身)	800	wr cmd tools menu
	なし	テーブルウィザード...	408	wr cmd table wizard

なし	スペル...	805	wr cmd spellcheck
なし	言語...	806	wr cmd language
なし	ドキュメント情報...	801	wr cmd doc information
なし	ドキュメント統計...	802	wr cmd doc statistics
なし	参照を再計算	803	wr cmd compute references
なし	参照をテキストに置換	804	wr cmd freeze references

メニューとサブメニューについて

これらの定数のうちいくつかはメニュー (例: `wr cmd view menu`) またはサブメニュー (例: `wr cmd change case submenu`) を参照します。

これらのコマンドは `WR GET COMMAND INFO` と `WR LOCK COMMAND` コマンドでのみ利用できます (`WR LOCK COMMAND` はメニューやサブメニュー全体を無効にしたり、再度有効にしたりします)。

これらの定数が `WR EXECUTE COMMAND` や `WR ON COMMAND` コマンドで使用されても、効果はありません。

✚ 付録C: エラーコード

下記は4D Writeから返されるエラーコードの一覧です。これらのコードは*WR Error number*、*WR Error text*そして*WR ON ERROR*コマンドで使用されます。

コード エラーテキスト

- 1002 印刷中にエラーが起きました。
- 1003 左マージンパラメータが無効です（右マージンを閉じることができません）。
- 1004 インデントパラメータが無効です（右マージンを閉じることができません）。
- 1005 右マージンパラメータが無効です（左マージンまたはインデントを閉じることができません）。
- 1006 タブパラメータが無効です。
- 1007 配列パラメータが無効です：配列が有効なタイプまたはサイズでないか、または配列自体がありません。
- 1012 ファイルが保存されてません。
- 1013 選択範囲が無効です（開始が0以下か、あるいは開始が終了より大きい値です）。
- 1015 ファイルが読み込まれていません。
- 1016 メニューまたはアイテム参照が無効です。
- 1017 このフィールドは、4D Writeのフィールドにはなれないみたいです。
- 1022 外部コマンドに渡されるエリアパラメータが無効です。
- 1023 4Dファイル参照番号が無効です。
- 1024 4Dのテキスト変数およびフィールドは、最大32000バイトです。
- 1028 WR Select関数に渡される位置が無効です。
- 1032 このファイルは存在しません。
- 1034 ピクチャが選択されていません。
- 1035 サイズパラメータが無効です。
- 1036 位置パラメータが無効です。
- 1038 このスタイルは存在しません。
- 1041 このコマンドを実行するにはメモリが足りません。
- 1044 イベントタイプが無効です。
- 1047 フィールド参照が無効です。
- 1048 オプション番号が無効です。
- 1051 このパスは存在しません。
- 1054 第1パラメータが無効です。
- 1055 第2パラメータが無効です。
- 1056 第3パラメータが無効です。
- 1057 第4パラメータが無効です。
- 1060 サブフィールドは挿入できません。
- 1065 このピクチャは壊れているようです。
- 1066 256以上のタブストップは作成できません。
- 1067 タブの位置が無効です。
- 1068 タブの整列が無効です。
- 1069 Blobを挿入できません。
- 1072 削除するハイフンはありません。
- 1073 式が無効です。
- 1074 Blobが無効です。
- 1075 テキストプロパティが範囲外です。
- 1076 テキストプロパティの値が範囲外です。
- 1077 フォントがシステム内にありません。
- 1078 不明のスタイルシートです。
- 1079 ドキュメントプロパティが範囲外です。
- 1080 ドキュメントプロパティの値が範囲外です。
- 1081 印刷中にセクションが変更されました。
- 1082 保存先番号が無効です。
- 1083 ページ番号内のピクチャが無効です。
- 1084 タブ番号が無効です。
- 1085 ページ番号の書式が範囲外です。
- 1086 ページ番号が無効です。
- 1087 段組み番号が無効です。

- 1088 行番号番号が無効です。
- 1089 オプション番号が無効です。
- 1090 統計番号が無効です。
- 1091 枠線参照が無効です。
- 1092 コマンド番号が無効です。
- 1093 印刷できません：ドキュメントはすでに印刷中です。
- 1094 予約されたスタイルシートです。
- 1095 ファイルを開けません。
- 1096 ファーストセーブされたWordファイルは開けません。
- 1097 ドキュメントが損傷していたため復旧されました。
- 1098 文字数が不正です。
- 1099 ページレイアウト情報が不正です。
- 1100 Wordドキュメントのいくつかのピクチャは読み込めません。

付録D: 削除されたv6.0.xコマンド

以下の4D Write version 6.0.xコマンドはバージョン6.5以降保守されていません。これらのコマンドは"WR"のあとに"R"がつけられ、現在のバージョンの4D Writeでは無視されます。

WR R Append break
WR R Append document
WR R Close document
WR R Create document
WR R EXPORT TRANSLATORS
WR R IMPORT TRANSLATORS
WR R INSTALL DEBUG WINDOW
WR R ModuleInfo
WR R REMOVE DEBUG WINDOW
WR R SET GLOBAL OPTIONS
WR R SUBSCRIBE

📌 付録E: 廃止予定コマンド

以前の4D Writeにあったいくつかのコマンドや関数は、バージョン6.5からよりパワフルで、プラグインに新しい機能をもたらす新しいレーチンに置き換えられました。以前のアプリケーションとの互換性を保ち、順次開発者が新しいコードに置き換えを行うことができるようにするため、これらの廃止予定コマンドは一時的に保守されています。これらのコマンドには"WR"のあとに"O"がつけられています。しかし新規の開発でこれらのコマンドを利用することは推奨されません。

バージョン11の4D Writeより、これらのコマンドはプラグインコマンドのリストに表示されません。以降のバージョンでは廃止される予定です。今後、コード中でこれらのコマンドが使用されている場合は、すみやかに新しいコマンドや関数に置き換えることを強くおすすめします。

以下は廃止予定コマンド、および現在の4D Writeで提供される代替のコマンドや解決法のリストです。

廃止予定コマンド	代替のコマンドやソリューション
WR O DISPLAY SCROLLBARS	<i>WR SET DOC PROPERTY</i>
WR O ON MENU	<i>WR ON COMMAND</i>
WR O DISPLAY RULER	<i>WR SET DOC PROPERTY</i>
WR O DISPLAY MENUBAR	<i>WR SET DOC PROPERTY</i>
WR O Get page	<i>WR GET CURSOR POSITION</i>
WR O Is Hyphen	<i>WR SELECT</i>
WR O PICTURE TO AREA	<i>WR PICTURE TO AREA</i>
WR O Find	<i>WR Find</i>
WR O EXPERT COMMAND	<i>WR LOCK COMMAND</i>
WR O CREATE STYLESHEET	<i>WR Create stylesheet</i>
WR O MOVE PICTURE	行揃え属性を使用 (左、右、中央) または段落のマージンを設定してピクチャを移動し、ピクチャを扱う新しいコマンドを使用する。
WR O DO COMMAND	<i>WR EXECUTE COMMAND</i>
WR O TEXT ALIGNMENT	<i>WR SET TEXT PROPERTY</i>
WR O SET ATTRIBUTES	<i>WR SET FONT</i>
WR O LINE SPACING	<i>WR SET TEXT PROPERTY</i>
WR O SET MARGINS	<i>WR SET TEXT PROPERTY</i>
WR O SET PACK OPTIONS	<i>WR SET DOC PROPERTY</i>
	<i>WR SET AREA PROPERTY</i>
WR O OPTIONS	<i>WR SET AREA PROPERTY</i>
WR O SET PREFERENCES	<i>WR SET DOC PROPERTY</i>
WR O SET TABS	<i>WR SET TAB</i>
WR O RESIZE PICTURE	<i>WR SET PICTURE SIZE</i>
WR O Area to picture	<i>WR Area to picture</i>
WR O Picture to offscreen area	<i>WR New offscreen area</i>
WR O INSERT HYPHEN	<i>WR EXECUTE COMMAND</i>
WR O INSERT PICTURE	<i>WR INSERT PICTURE</i>
WR O Get ScrollBars	<i>WR Get doc property</i>
WR O GET ATTRIBUTES	<i>WR Get font</i>
WR O GET STYLESHEET	<i>WR GET STYLESHEET INFO</i>
WR O GET PICTURE	<i>WR GET PICTURE SIZE</i>
WR O GET MARGINS	<i>WR Get text property</i>
WR O Get pack options	<i>WR Get doc property</i>
WR O GET PREFERENCES	<i>WR Get doc property</i>
WR O GET RULER	<i>WR Get text property</i>
WR O GET TABS	<i>WR GET TAB</i>

WR O SET STYLESHEET	<i>WR SET STYLESHEET INFO</i>
WR O CHANGE STYLE	<i>WR SET TEXT PROPERTY</i>
WR O Font name	4Dコマンドを使用する。
WR O Count Stylesheet	<i>WR Count</i>
WR O Page number	<i>WR INSERT PAGE NUMBER</i>
WR O Font number	4Dコマンドを使用する。
WR O COMPUTE NOW	<i>WR EXECUTE COMMAND</i>
WR O Replace	<i>WR Replace</i>
WR O AUTO SAVE	<i>WR Area to picture</i>
WR O STATISTICS	<i>WR Count</i>
WR O MENU STATUS	<i>WR GET COMMAND INFO</i>
WR O REMOVE HYPHEN	<i>WR SELECT</i>
	<i>WR DELETE SELECTION</i>
WR O DELETE STYLESHEET	<i>WR DELETE STYLESHEET</i>
WR O STRUCTURE ACCESS	<i>WR LOCK COMMAND</i>
WR O Save to picture	<i>WR Area to picture</i>

4D Write - コマンドリスト (文字順)

A B C D E F G I L M N O P R S T U

-  WR ADD STYLESHEET TAB
-  WR ADD TAB
-  WR APPLY STYLESHEET
-  WR Area to blob
-  WR Area to picture