

# 4D View Pro リファレンス

## 📌 概要

- 📄 リストボックスの先進的な機能
- 📄 4D View Pro エリアを定義する
- 📄 4D View Pro エリアを使用する
- 📄 4D View ドキュメントの変換
- 📄 4D View Pro フォーミュラ
- 📄 4D View Pro ランゲージ
- 📄 コマンドリスト (文字順)

## 概要

### 4D View Pro について

---

4D View Pro はスプレッドシート機能とリスト表示に関連した先進的な機能のセットを提供します。4D View Pro は4D ユーザーに、旧式の4D View 製品機能の一部に対する現代的で統合された代替を提供します。

4D View Pro は二つの部分から構成されます:

- 行の高さをデベロッパーが完全にコントロールすることを可能とする先進的なリストボックス機能のセット
- デベロッパーがフォームにスプレッドシートを埋め込むことを可能とするコンポーネントと4D フォームエリア (4D v16 R4ではプレビュー)

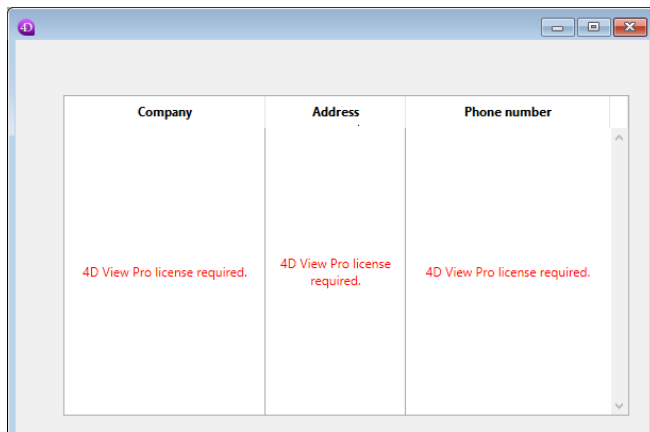
### インストールと有効化

---

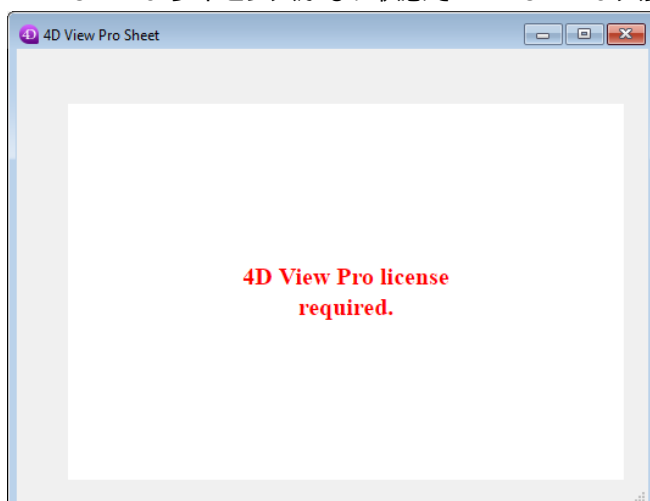
旧式の4D View 製品とは異なり、4D View Pro 機能は4D 自身に直接含まれているので、配布と管理が簡単になります。改めて追加でインストールするものは何ともありません。

しかしながら、4D View Pro は4D View と同じライセンスを必要とします。これらの機能を使用するためには、アプリケーション内にそのライセンスをインストールしておく必要があります。4D Viewライセンスがインストールされていない場合、4D View Pro 機能を必要とするオブジェクトのコンテンツ(リストボックスあるいは4D View Pro エリア)はランタイムでは表示されず、エラーメッセージだけが表示されます。

- 4D View Pro ライセンスがない状態で4D View Pro 機能を使用するリストボックスを表示した場合:



- 4D View Pro ライセンスがない状態で4D View Pro スプレッドシートエリアを表示した場合:






## 📄 リストボックスの先進的な機能

概要の章で説明があるように、4D View Pro は先進的なリストボックス機能のセットからできています。これらのリストボックス機能には以下のようなものが含まれます:

- リストボックスカラムに割り当てられたオブジェクト配列:

Label	Value
Document Name	MyReport
Document Type	PDF ▼
Reference	123456
Category	 ...
Include Abstract	<input checked="" type="checkbox"/>
Printable area size (height)	297 <input type="text"/> mm
Printable area size (width)	210 <input type="text"/> mm
Show Preview	<input type="button" value="Preview..."/>

詳細な情報については、[カラム内でのオブジェクト配列の使用\(4D View Pro\)](#)のページを参照してください。




- リストボックス内の可変行高コントロール:

RowNum	Countries	Population
1	Luxembourg	502 202 ^
2	Latvia	1 973 700
3	Kuwait	4 044 500
4	Croatia	4 284 889
5	Denmark	5 699 220
6	Nicaragua	6 071 045
7	Serbia	7 306 677
8	Honduras	8 249 574
9	Austria	8 572 895
10	Hungary	10 005 000
11	Czech Republic	10 674 947

この機能は以下の要素に基づいています:

- 行高さ配列 プロパティ
  - **LISTBOX Get row height** および **LISTBOX SET ROW HEIGHT** コマンド
  - **LISTBOX Get array** および **LISTBOX SET ARRAY** コマンドの `lk_row_height_array` 定数
- 詳細な情報については、4D ドキュメントを参照してください。

- 自動可変行高:

Picture	Name	Short text	Full text
	ABORT	The <b>ABORT</b> command is used from within an error-handling project method installed using the command <b>ON ERR CALL</b> .	If you use an <i>error-handling</i> project method to catch errors, 4D neither displays its standard error dialog box nor interrupts the execution of your code. Instead, 4D calls your error-handling project method (that you can see as an exception handler), and resumes the execution to the next line of code.
	ASSERT	The <b>ASSERT</b> command evaluates the <code>boolExpression</code> assertion passed in parameter and, if it returns false, stops the code execution with an error message. The command works in interpreted and compiled mode.	If <code>boolExpression</code> is true, nothing happens. If it is false, the command triggers the error: "10518 an displays by default the text of the assertion preceded by the message "Assert failed.". You can intercept this error via a method installed using the <b>ON ERR CALL</b> command, in order, for example, to provide info for a log file. Optionally, you can pass a <code>messageText</code> parameter to display a custom error message instead of the text of the assertion. An assertion is an instruction inserted in the code that is responsible for detecting any anomalies.
	Asserted	The <b>Asserted</b> command has an operation similar to that of the <b>ASSERT</b> command, with one difference in that it returns	It therefore allows the use of an assertion during the evaluation of a condition (see the example). For more information about the operation of assertions and the parameters of this command please refer to the description of the <b>ASSERT</b> command.

この機能は以下のものに基づいています:

- 自動行高 プロパティ(リストボックスレベルおよびカラムレベルに対して利用可能)
- **LISTBOX SET PROPERTY** および **LISTBOX Get property** コマンドの `lk_auto_row_height` 定数
- 高さの境界線を定義するための **LISTBOX SET AUTO ROW HEIGHT** および **LISTBOX Get auto row height** コマンド

詳細な情報については4Dドキュメントを参照してください。

# 4D View Pro エリアを定義する

## 概要

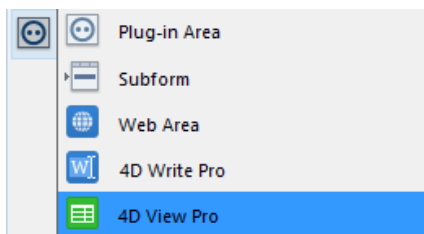
4D View Pro を使用すると、4D フォーム内にスプレッドシートエリアを挿入・表示できるようになります。スプレッドシートとはセルのグリッドを格納したアプリケーションのことで、これらのセルに情報を入力したり、計算を実行させたり、あるいはピクチャーを表示するということができます。

フォーム内で4D View Pro エリアを使用すると、4D View Pro コマンドを使用してスプレッドシートドキュメントを読み込んだり書き出したりすることができます。

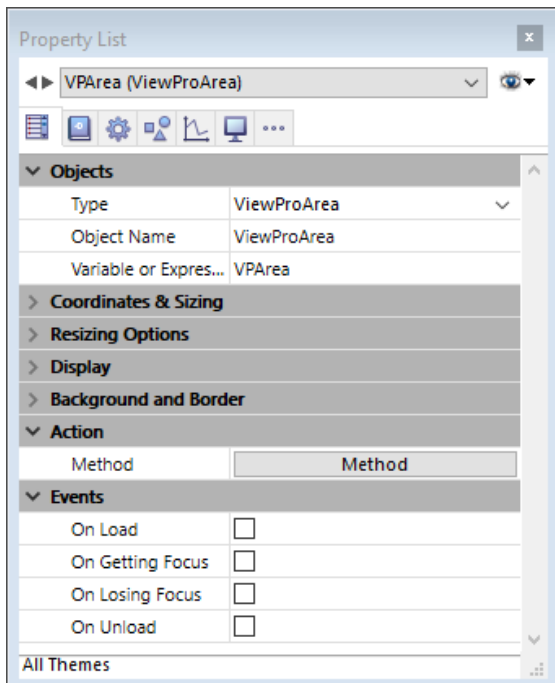
**互換性:** 4D View Pro エリアは32-bit 版の4D では使用できません。

## エリアの作成

4D View Pro ドキュメントは**4D View Pro** という名前の4D フォームオブジェクト内にて表示され、手動で編集されます。このオブジェクトはオブジェクトバーの最後のツール(プラグインエリア、Webエリア、等)の中の一部として含まれています。



4D View Pro フォームエリアはプロパティリスト内にある標準のプロパティを利用して設定することが可能です。例えば、**オブジェクト名**、**変数あるいは式**、**座標**、**表示**、**動作**、**イベント**などです。



- **オブジェクト名:** 4D View Pro ドキュメントを格納して表示する4D フォームエリアの名前
- **変数あるいは式:** 4D View Pro フォームエリア変数。

フォームが実行されると、4D View Pro エリアはデフォルトでスプレッドシートを表示します:

	A	B	C	D	
1					
2					
3					
4					
5					
6					
7					
8					
9					
10					
11					
12					

# 4D View Pro エリアを使用する

## 概要

---


フォーム内で実行されると、4D View Pro エリアはセル編集や式入力などを含む、基本的なスプレッドシート機能を提供します。より先進的な機能は、4D View Pro ランゲージを使用することで利用可能です。

## セレクション、入力、およびナビゲーションの基本

---

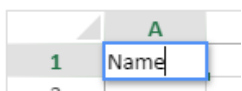
スプレッドシートは行と列から構成されています。各行には番号が割り当てられています。各列には文字(あるいはアルファベットの文字数を超えた場合には文字のグループ)が割り当てられています。行と列の交差する場所がセルになります。セルは選択し、そのコンテンツを編集することができます。

### セレクション

- セルを選択するには、単にセルをクリックするか、キーボードの矢印キーを使用します。その中身(あるいは式)がセルの中に表示されます。
- 複数の連続したセルを選択するためには、マウスをセレクションの端から端へとドラッグします。あるいは**Shift**キーを押しながらセレクションの二つの端をクリックすることでも選択可能です。
- スプレッドシート内にあるすべてのセルを選択するためには、エリアの左上端にある  のセルをクリックします。
- 列全体を選択するためには、対応する文字(あるいは文字セット)をクリックします。
- 行全体を選択するためには、対応する数字をクリックします。
- 連続していないセルのグループを選択するためには、**Ctrl**キー(Windows)あるいは**Command**キー(Mac)を押しながら、選択したいセルをそれぞれ選択します。
- セルの選択を解除するためには、スプレッドシート内のどこかをクリックすれば選択解除されます。

### 入力とナビゲーション

セルをダブルクリックすると、その対応するセルの入力モードへと入ることができます。セルが空でない場合、挿入カーソルはセルのコンテンツの最後に置かれます。



データはセルが選択されていれば、たとえ挿入カーソルが非表示であっても直接入力することができます。その場合、入力した内容はセルのコンテンツを上書きします。

**Tab**キーはセルの内容を確定し、一つ右のセルを選択します。**Shift + Tab** キーの組み合わせで、セル入力を確定した後一つ左のセルを選択します。

**キャリッジリターン**キーはセルの入力を確定し、一つ下のセルを選択します。**Shift + キャリッジリターン**キーの組み合わせで、セル入力を確定したあと一つ上のセルを選択します。

方向キー(矢印)を使用すると、矢印の方向へとセルの選択を移動することができます。

### 式、関数、あるいは参照の入力

---

4D View Pro エリアに式や関数を入力する方法:

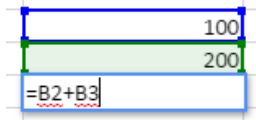
- 式や関数を入力するセルを選択します。
- =(イコール記号)を入力します。



3. 式を入力します。

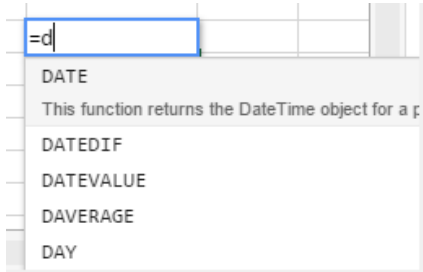
あるいは

セルをクリックし、式の中にセルの参照を入力します。



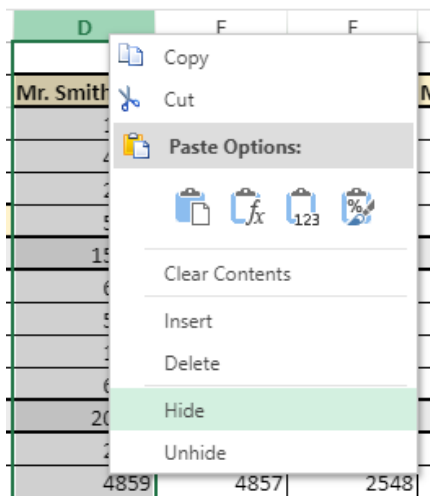
あるいは

入力したい関数の最初の文字を入力します。すると利用可能な関数と参照の一覧のポップアップメニューが表示されるので、必要な要素を選択することができます：



## コンテキストメニュー

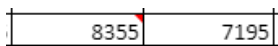
4D View Pro エリアでは、コピー/ペーストといった標準編集機能だけでなく、基本的なスプレッドシート機能も備えている自動コンテキストメニューを利用することができます：



**注：**コンテキストメニューのコピー/カット/ペースト機能はスプレッドシートエリア内のみで動作し、システムのペーストボードへはアクセスしません。しかしながら**Ctrl+c/Ctrl+v** といったシステムショートカットは動作し、エリアと他のアプリケーション間でデータを交換するために使用することは可能です。

このメニューは、クリックしたエリアに応じて、追加の機能を提示します：

- カラム、あるいは行ヘッダーをクリックした場合：コンテンツの**挿入、削除、非表示、再表示**
- セルあるいはセルのレンジへのクリック：
  - **フィルター：**フィルターを使用して行を非表示にします(SpreadJS ドキュメントの[Filtering rows](#) を参照してください)
  - **並べ替え：**カラムのコンテンツを並べ替えます
  - **コメントを挿入：**エリアに対するユーザーのコメントを受け付けます。エリアに対してコメントが入力されている場合、エリアの左上のセルに小さな赤い三角形が表示されます：



## 4D View ドキュメントの変換

**VP Convert from 4D View** コマンドを使用することで、4D View ドキュメントを4D View Pro エリアへと変換することができます。4D View ドキュメントに保存されているほとんどの情報とプロパティ(フォーマット、スタイル、境界線、値、フォーミュラ、セクション、ズーム、等を含む)は自動的に変換されます。基本的に、変換された4D View ドキュメントは4D View エリアでレンダリングされていたのと同じように4D View Pro エリア内にレンダリングされます:

元の4D View ドキュメント:

Éléments de missions	%	Montant HT
Avant Projet Détaillé	20.00 %	4 180.00 €
Dossier de Permis de Construire	23.00 %	4 807.00 €
Assistance à la passation des contrats de travaux	15.00 %	3 135.00 €
Direction de l'exécution des contrats de Travaux	34.00 %	7 106.00 €
Assistance aux Opérations de Réception	8.00 %	1 672.00 €
<b>TOTALUX</b>	<b>100.00 %</b>	<b>20 900.00 €</b>

Avancement	
Taux	Montant HT
100.00 %	4 180.00 €
100.00 %	4 807.00 €
100.00 %	3 135.00 €
46.15 %	3 279.72 €
	<b>15 401.72 €</b>

Total des missions réalisées HT : 15 401,72 €  
Moins précédemment facturé HT : 14 855,10 €  
Montant de la révision HT : 546,62 €

Montant de la note HT : **546,62 €**  
TVA à 19,6% : **107,14 €**  
Montant TTC : **653,76 €**

Versements directs aux sous-traitants : 104,60 €  
Montant TTC à payer : **549,16 €**

4D View Pro エリアに変換されたドキュメント

Éléments de missions	%	Montant HT
Avant Projet Détaillé	20.00 %	4 180.00 €
Dossier de Permis de Construire	23.00 %	4 807.00 €
Assistance à la passation des contrats de travaux	15.00 %	3 135.00 €
Direction de l'exécution des contrats de Travaux	34.00 %	7 106.00 €
Assistance aux Opérations de Réception	8.00 %	1 672.00 €
<b>TOTALUX</b>	<b>100.00 %</b>	<b>20 900.00 €</b>

Avancement	
Taux	Montant HT
100.00 %	4 180.00 €
100.00 %	4 807.00 €
100.00 %	3 135.00 €
46.15 %	3 279.72 €
	<b>15 401.72 €</b>

Total des missions réalisées HT : 15 401,72 €  
Moins précédemment facturé HT : 14 855,10 €  
Montant de la révision HT : 546,62 €

Montant de la note HT : **546,62 €**  
TVA à 19,6% : **107,14 €**  
Montant TTC : **653,76 €**

Versements directs aux sous-traitants : 104,60 €  
Montant TTC à payer : **549,16 €**

変換されたドキュメントがオリジナルと可能な限り同じ状態を維持できる様にとめてはいますが、一部の機能については正常にレンダリングされない可能性があります。これらの機能については以下の の段落にて詳細な説明があります。

## 変換プロセス

---

**注:** 4D View ドキュメントの変換機能は、お客様からの声に基づいて日々改善の途中です。たとえ変換がうまくいったとしても、そのあとも元の4D View BLOB、あるいはドキュメントのコピーを保存しておくことが強く推奨されます。

4D View プラグインドキュメントの現在の状況においては、変換プロセスのためには以下の手順を踏んでいただく必要があります。

1. 4D View ドキュメント(.4pv)をBLOBへと読み込みます:

**注:** 4D View ドキュメントがすでにBLOB フィールドに保存されている場合、手順2へおすすみ下さい。

```
C_BLOB($pvblob)
DOCUMENT TO BLOB("document.4PV";$pvblob)
```

2. 4D View ドキュメントを格納したBLOBで**VP Convert from 4D View** を呼び出します:

```
C_OBJECT($vpObj)
$vpObj:=VP Convert from 4D View($pvblob)
```

3. できたオブジェクトを4D View Pro エリアフォームオブジェクト、あるいは4D View Pro ドキュメントに割り当て、その結果をみます。

```
VP IMPORT FROM OBJECT("4DViewProArea";$vpObj)
```

## 変換の詳細

---

機能	変換ステータス	補足
ドキュメント属性	ドキュメントの表示属性は変換されます: 選択されたセル、ズーム、グリッドの表示	
カラムと行	ドキュメント情報は変換されます: バージョン、題名、執筆者、会社、注釈、作成日および変更日	
枠線	定義されたカラムと行は全てオリジナルのサイズで変換されます。	4D View Pro では一つのシングルバーの枠線と、一つのダブルバーの枠線のみが利用可能です。シングルバーの枠線はオリジナルと同じ様に変換されますが、ダブルバーの枠線はどれも4D View Proのダブルバーモデルへと変換されます。
スプリッタースタイルとフォント	カラムと行のヘッダーは制約なしに変換されます。	
フォーマットとセル名	枠線は太さとカラーが変換されます。グリッドの表示設定も変換されます。	
スプリッター	現在は変換されません	
スタイルとフォント	スタイルとスタイルシートは変換されます。条件付きスタイルはサポートされていません。	廃止予定のテキストスタイル( <i>i.e.</i> 影付き、コンデンスド、など)およびQuickDraw フォントは変換されません。
回転スタイル	回転スタイル(角度付きテキスト)は現在は変換されません。	
フォーマットとセル名	セルフォーマットは似たレンダリングを持つ利用可能なフォーマットへと変換されます。全てのデータ型フォーマットがサポートされます: テキスト、数値、日付と時間、ブール、ピクチャー。セル名は変換されます	フォーマットの変換は一部のドキュメントでは予期せぬ結果を生むことがあります。なぜならフォーマットは時にシステム設定に依存していて(例: 浮動小数点など)、全体的あるいは個別に定義されているからです。

ユーザー定義の4D フォーマット("|"で始まるもの) は現在は変換されません。

PV SET CELL CONTROL で作成されたコントロール(ボタン/ラジオボタン/チェックボックス/ドロップダウン/コンボボックス)は現在はサポートされていません。

ピクチャー

ピクチャーはサポートされますが、変換には一部制約が付きます

4D View ピクチャーは複数のコーデックを内包することができましたが、現在は廃止予定となっています。変換されたピクチャーはhtmlレンダリングに最適なコーデック(svg、png、jpeg、gif)のみが保持され、base64で保存されます。

トランケート(中央合わせ)/繰り返しフォーマットのピクチャーは現在は変換されません。

背景画像は各ページにおいて繰り返されることはありません(この概念は4D View Proには存在しません)。

印刷動的なリンクフォーミュラ

現在はサポートされていません

4D View ドキュメント内で定義された印刷オプションと印刷設定は変換されます。

フィールドまたは変数にリンクされたセルまたはカラムは、現在はサポートされていません。

フォーミュラは変換されますが、4D コマンド、メソッド、変数またはフィールドへの参照は、現在はサポートされていません。

# 4D View Pro フォーマキュラ

## 概要

---
























4D View Pro ファンクションはフォーマキュラで使用されます。全ての4D View Pro フォーマキュラは値を返す式です。全ての式はオペランドと演算子から成り立っています:

- **演算子:** [定数と演算子](#) を参照してください。
- **オペランド**は、いくつかのカテゴリに分けられます:
  - 値
  - 他のセルへの参照(相対参照、絶対参照、ミックス参照、あるいは名前での参照)
  - 4D 変数、フィールド、ファンクション
  - 4D View Pro ファンクション

フォーマキュラを入力するには、以下のようにします:

1. フォーマキュラあるいはファンクションを入力セルを選択します。
2. = (イコール記号)を入力します。
3. フォーマキュラを入力し、Enter キーを入力します。

4D View Pro ではたくさんのファンクションを使用することができます。このセクションではファンクションのうち、必要不可欠なものだけをまとめた一覧を提供します。4D View Pro によってサポートされるファンクションの一覧全体は、[Spreadsheets documentation](#) にまとめられています。

-  定数と演算子
-  セル参照
-  4D View プラグインフォーマキュラの変換
-  ABS
-  ACOS
-  AND
-  ASIN
-  ATAN
-  AVERAGE
-  COLUMNLETTER
-  COS
-  COUNTA
-  EXP
-  FALSE
-  FINDCELL
-  FV
-  IF
-  INDIRECT
-  ISBLANK
-  LEN
-  LN
-  LOOKUP
-  MAX

- ⚙ MID
- ⚙ MIN
- ⚙ NOT
- ⚙ NOW
- ⚙ NPER
- ⚙ OR
- ⚙ PI
- ⚙ PMT
- ⚙ PV
- ⚙ RAND
- ⚙ RATE
- ⚙ ROUND
- ⚙ ROW
- ⚙ SIN
- ⚙ SQRT
- ⚙ STDEV.P
- ⚙ SUBSTITUTE
- ⚙ SUM
- ⚙ TAN
- ⚙ TEXT
- ⚙ TODAY
- ⚙ TRUE
- ⚙ TYPE
- ⚙ VAR.P

## ✦ 定数と演算子

### データの型ごとの演算子

---

4D View Pro は5つのデータの型をサポートします。それぞれのデータ型について、特定の定数と演算子がサポートされます。

データ型	定数	演算子
数値	1.2	+ (加法)
	1.2 E3	- (減法)
	1.2E-3	* (乗法)
	10.3x	/ (除法)
		^ (べき乗、数値を何回自身にかけるか)
日付	10/24/2017	% (パーセント -- 演算子の前の数値を100で割る)
		+ (日付 + 日数 -> 日付)
		+ (日付 + 時間 -> 日付 + その日の時間)
		- (日付 - 日数 -> 日付)
		- (日付 - 日付 -> 2つの日付間の日数)
時間	10:12:10	経過時間演算子:
		+ (加法)
		- (減法)
		* (経過時間 * 数値 -> 経過時間)
		/ (経過時間 / 数値 -> 経過時間)
文字列	'Sophie' あるいは "Sophie"	& (連結)
ブール	TRUE あるいは FALSE	-

### 比較演算子

---

同じ型の二つのオペランドに対して以下の演算子を使用することができます:

演算子	比較
=	等しい
<>	異なる
>	大きい
<	未満
>=	以上
<=	以下

### 演算子優先順位

---

以下の表は演算子を優先度順に並べたものです(優先度が高い方が上):



演算子	詳細
()	カッコ (グループ化)
-	否定
+	追加
%	パーセント
^	べき乗
* and /	乗法と除法
+ and -	加法と減法
&	連結
= > < >= <= <>	比較

## 式内でのオペランド優先順位

---

二つ以上の異なるオペランドが同じ名前を持つ場合、4D View Pro は以下の順番に沿ってそれぞれの要素の型を決定づけます:

優先順位	要素の型
1	セル参照
2	セル名
3	4D View Proファンクション
4	プロジェクトメソッド
5	4Dコマンド
6	変数

## ✦ セル参照

式の中では、他のセルにセルアドレスで参照をします。こういった式を他のセルにコピーすることも可能です。例えば、C8に入力された以下の式では、その上2つに入力された値を足して結果を表示します。

= C6 + C7

この式は常にセルC6とC7を参照します。つまり、4D View Proは式内で使用する値として他のセルにある中身を参照するように指示されるということです。

こういった式を新しい場所にコピーあるいは移動させた場合、式内のそれぞれのセルアドレスは、入力の仕方に応じて変化するかそのまま止まるかします。

- 参照が変わるものは**相対参照**と呼ばれ、式が入っているセルからどれだけ上下左右に離れているかでセルを参照します。
- 常に特定のセルを参照するものは**絶対参照**と呼ばれています。
- また、**ミックス参照**という、特定の行あるいはカラムに固定した参照をするものを作成することもできます。

### 参照の記法

セル座標(例えばC5など)のみを使用すると、4D View Proは参照を相対参照として解釈します。文字と数字の前に\$(ドル)記号を入れることで、参照を絶対参照にすることができます(例: \$C\$5)。

ドル記号を文字あるいは数字の前だけに入れることで相対参照と絶対参照をミックスすることができます(例: \$C5、C\$5など)。ミックス参照を使用すると、行かカラムを絶対参照として指定しながら、もう片方については相対参照させることが可能になります。

素早くかつ正確に絶対参照をする方法としては、セルを命名し、その名前をセルアドレスの場所に使用することです。命名されたセルへの参照は常に絶対参照になります。

以下の表は異なる記法による効果をまとめたものです:

例	参照のタイプ	詳細
C5	相対	参照はセルC5に対して相対的で、参照が最初に使用されたセルの位置によります。
\$C\$5	絶対	参照は絶対参照です。どこで使用されるかにかかわらず、常にセルC5を参照します。
\$C5	ミックス	参照は常にカラムCを参照しますが、行の参照は相対参照で、参照が最初に使用されたセルの位置によります。
C\$5	ミックス	参照は常に5行目を参照しますが、カラムの参照は相対参照で、参照が最初に使用されたセルの位置によります。
セル名	絶対	参照は絶対参照です。どこで使用されるかにかかわらず、常に命名されたセルを参照します。

## 🌿 4D View プラグインフォーミュラの変換

### 概要

---

**4D View ドキュメントの変換** のページにも記載があるように、4D View プラグインドキュメントの中身とプロパティのほとんどは、4D View Pro ドキュメントへと変換することができます。

フォーミュラもまた変換されます。しかしながら、フォーミュラのランゲージは4D View と4D View Pro で若干異なります。変換した結果として3つの場合が考えられます：

- 4D View の機能(演算子、定数、ファンクション)は4D View Pro のものと完全に同一である：この場合、変換はユーザーからは透過です。
- 4D View の機能は4D View Pro でサポートされているが、異なるファンクションや演算子を使用する：この場合、自動マッピングが実行されます。
- 4D View の機能は4D View Pro では部分的にのみサポートされる、あるいはサポートされていない：この場合、変換されたフォーミュラを想定されるように動くように調整する必要があります。

以下の一覧は、4D View フォーミュラ機能とそれに対応する4D View Pro の機能をまとめたものです。

### 演算子

---

## 4D View 4D View Pro

### 数値演算子

加法	+	+
減法	-	-
乗法	*	*
除法	/	/
剰余	¥	<a href="#">MOD</a>
商	÷	<a href="#">TRUNC(a/b)</a>
べき乗	^	^
パーセント	%	%

### ブール演算子

AND	&	<b>AND</b>
OR		<b>OR</b>
Not	~	<b>NOT</b>

### 文字列演算子

連結	+	&
削除	-	<a href="#">SUBSTITUTE</a> 、例: "Down Trend"->"Down" は、 <a href="#">SUBSTITUTE("Down Trend","Down","")</a> で置き換えられます
位置	¥	<a href="#">FIND</a> (大文字と小文字を区別する)あるいは <a href="#">SEARCH</a> (大文字と小文字を区別しない)

### 日付演算子

日付+日にち->日付	+	+
日付+時間->日付+時間	+	+
の日にちのみ	+	+
日付-日にち->日付	-	-
日付-日付->日数	-	-

### 時間演算子

加法	+	+
減法	-	-
乗法	*	*
除法	/	/

### 比較演算子

等しい	=	=
異なる	#	<>
より大きい	>	>
より小さい	<	<
以上	>=	>=
以下	<=	<=

## ファンクション

---

4D View	4D View Pro	補足
Abs	<b>ABS</b>	
AddToDate	<b>DATE</b>	AddToDate(date;years;months;days) は、 <b>DATE</b> ( <b>YEAR</b> (date)+years, <b>MONTH</b> (date)+months, <b>DAY</b> (date)+days)で置き換えられます
date+time	<b>TIME</b>	<b>DATE</b> (date) + <b>TIME</b> (time)
And	<b>AND</b>	
ArcCos	<b>ACOS</b>	
ArcSin	<b>ASIN</b>	
ArcTan	<b>ATAN</b>	
Area	-	n/a (プラグインエリアはありません)
Average	<b>AVERAGE</b>	
Cell	<b>INDIRECT</b>	
Column	<b>COLUMNLETTER</b>	<b>COLUMN</b> は数値を返しません(文字は返しません)
Cos	<b>COS</b>	
Count	<b>COUNTA</b>	
CurrentDate	<b>TODAY</b>	
CurrentTime	<b>NOW</b>	
CVCompound	<b>PV</b>	CVCompound(1%;5;1000)は <b>PV</b> (1%,5,-1000)で置き換えられます
CVSimple	<b>PV</b>	CVSimple(1%;5;5*1000)は <b>PV</b> (1%,5,-1000)で置き換えられます。二つ連続したカンマに注意してください。
Empty	<b>ISBLANK</b>	
Eval4D	-	現在利用不可
Exp	<b>EXP</b>	
False	<b>FALSE</b>	
Find	<b>LOOKUP</b>	
FindCell	<b>FINDCELL</b>	
FVCompound	<b>FV</b>	FVCompound(1%;35;35*1000)は <b>FV</b> (1%,35,-1000)で置き換えられます
FVSimple	<b>FV</b>	FVSimple(12%;35;35*1000)は <b>FV</b> (12%,35,-35*1000)で置き換えられます。二つの連続したカンマに注意してください。
If	<b>IF</b>	
Length	<b>LEN</b>	
Log	<b>LN</b>	
Max	<b>MAX</b>	
Min	<b>MIN</b>	
MonthlyValue	<b>PMT</b>	MonthlyValue(10.5%/12,48,6500)は <b>PMT</b> (10.5%/12,48,-6500)で置き換えられます。
Not	<b>NOT</b>	
Or	<b>OR</b>	
PeriodNumber1	<b>NPER</b>	PeriodNumber1(10.5%/12;166.42;6500)は <b>NPER</b> (10.5%/12,-166.42,6500)で置き換えられます。
PeriodNumber2	<b>NPER</b>	PeriodNumber2(10.5%/12,5000,3000)は <b>NPER</b> (10.5%/12,,3000,-5000)で置き換えられます。二つの連続したカンマに注意してください。
Pi	<b>PI</b>	
Random	<b>RAND</b>	
Range	<b>INDIRECT</b>	Range("A1";"A3")は <b>INDIRECT</b> ("A1:A3")で置き換えられます。A1とA3の

間のコロン記号に注意してください。

Rate1            **RATE**

Rate1(5;1000;3000)は**RATE**(5,-1000,3000)で置き換えられます。

Rate2            **RATE**

Rate2(5,6000,2800)は**RATE**(5,,2800,-6000)で置き換えられます。二つの連続したカンマに注意してください。

Rounding        **ROUND**

Row              **ROW**

Sin               **SIN**

SquareRoot      **SQRT**

StdDeviation    **STDEV.P**

String           **TEXT**

SubString       **MID**

Sum              **SUM**

Tan               **TAN**

True             **TRUE**

Type             **TYPE**

Variance        **VAR.P**

4D View Proで返される型は、4D Viewのものとは異なります。

ABS ( value )

引数	型	説明
value	実数, 式	絶対値を返す数値

## 説明

---

**ABS** ファンクションは`value` 引数で指定した値の絶対値を計算します。`value` 引数が負の値である場合、正の値が返されます。ファンクションは数値データの値あるいは参照を受け付け、数値データを返します。

## 例題

---

```
ABS(-6) //結果:= 6
```

```
ABS(16-26) //結果:= 10
```

```
ABS(6) //結果:= 6
```

## ⚙️ ACOS

ACOS ( value )

引数	型	説明
value	実数	→ アークコサインを求める角度。-1から+1の間

### 説明

---

**ACOS** ファンクションは *value* 引数で指定した値となるアークコサインの角度を計算します。 *value* 引数の値は-1 から+1 の間でなければなりません。

返される角度は0 からPI の間のラジアン単位の角度です。結果を度数に変換するためには、得られた結果に180/PI をかけます。

### 例題

---

```
ACOS(0.5) //結果:= 1.0471975512
```



## AND

AND ( logicalValue {, logicalValue2 , ... , logicalValueN} )

引数	型	説明
logicalValue	ブール, Number, 式	→ 評価する値

### 説明

---

**AND** ファンクションは全ての引数がtrueであればTRUEを返します。それ以外の場合には、少なくとも一つの引数がfalseの場合にはFALSEを返します。

ファンクションはブール値、数値(0または1)、あるいは論理式(TRUE あるいはFALSE)を最大255個まで受け取ることができます。また値を個別に渡すのではなく、単一の配列を指定することもでき、この場合最大で255個までの配列を指定することができます。 *logicalValue* 引数を式として指定することもできます。

### 例題

---

```
AND(D12,E12) //D12 と E12 のセルの値がどちらもtrue か1 であればTRUE、それ以外の場合にはFALSE
```

```
AND(D2:D12) //True if D12 から D12 までのセルの値がどれもtrue か1 であればTRUE、それ以外の場合にはFALSE
```

```
AND(5+3=8,5+1=6) //TRUE
```

```
AND(1,TRUE) //TRUE
```

ASIN ( value )

引数	型	説明
value	実数 →	角度の正弦(sin)。-1から+1の間

## 説明

---

**ASIN** ファンクションはアークサイン、つまり引数で指定されたつまり、正弦(sin) がvalue 引数で指定された値となる角度を返します。このファンクションは数値データを受け付け、数値データを返します。

value 引数には、求めたい角度の正弦を指定します。正弦の値は-1 から+1 の間でなければなりません。

返される角度はラジアン単位になり、 $-\pi/2$  から  $\pi/2$  の間になります。結果を度数に変換するためには、得られた結果に $180/\pi$  をかけます。

## 例題

---

```
ASIN(0.5) //0.5235987756
```

ATAN ( value )

引数	型	説明
value	実数 →	角度の正接(tan)。-1から+1の間

## 説明

---

**ATAN** ファンクションはアークタンジェントを計算します。つまり、正接(tan) が *value* 引数で指定された値となる角度を返します。このファンクションは数値データを受け付け、数値データを返します。

*value* 引数には、求めたい角度の正接を指定します。値は-1 から+1 の間でなければなりません。

返される角度はラジアン単位になり、 $-\pi/2$  から  $\pi/2$  の間になります。結果を度数に変換するためには、得られた結果に $180/\pi$  をかけます。

## 例題

---

```
ATAN(1) //結果:= 0.7853981634
```

## AVERAGE

AVERAGE ( value {, value2 , ... , valueN} )

引数	型		説明
value	実数, 配列	→	計算をしたい数値

### 説明

---

**AVERAGE** ファンクションはvalue 引数の値の平均を計算します。

value 引数には、以下のものを渡すことができます:

- 実数値あるいは倍調整数値
- セルのレンジ、あるいは複数のセルのレンジ

最大で255個まで引数を含めることができます。

### 例題

---

```
AVERAGE(98,72,85) //結果=85
```

```
AVERAGE(A1,B3,D5,E9,L8,L9)
```

```
AVERAGE(R1C1,R3C2)
```

```
AVERAGE(A1:A9)
```

```
AVERAGE(A1:A9,B1:B9,D5:D8)
```

## 🔧 COLUMNLETTER

COLUMNLETTER ( {reference} )

引数	型		説明
reference	CellRef	→	セルあるいはセルのレンジ

### 説明

---

**COLUMNLETTER** ファンクションは*reference* 引数が参照するカラムの文字を返します。

*reference* 引数にはセルあるいはセルのレンジを渡すことができます。*reference* 引数が省略された場合、デフォルトの引数は**COLUMNLETTER** ファンクションが置かれたセルの参照となります。

### 例題

---

```
COLUMNLETTER(A9) //A
```

```
COLUMNLETTER(B1:B5) //B
```

```
COLUMNLETTER() //カレントのカラム文字
```

## COS

COS ( value )

引数	型		説明
value	実数	→	余弦を求める角度

### 説明

---

**COS** ファンクションは`value` 引数で渡した値が指定する角度の余弦(cos)を返します。ファンクションは数値データを受け付け、数値データを返します。

`value` 引数には、余弦を取得したい実数(角度)を渡します。角度はラジアン単位で表現されている必要があります。角度が度単位であった場合、ラジアンに変換するためには $\text{PI}/180$ をかけます。

### 例題

---

```
COS(45*PI()/180) //0.7071067812
```

COUNTA ( value {, value2 , ... , valueN} )

引数	型		説明
value	実数, 配列	→	カウントしたいセルあるいはセルのレンジ

## 説明

---

**COUNTA** ファンクションは、*value* 引数で指定されたセルのうち、空でないセルの数を数えます(つまり数値、テキスト、あるいは論理値を含むセル)。ファンクションはセル参照を受け付け、数値データを返します。

*value* 引数には、最大で255個までの個別のセルあるいは単一の値の配列を渡すことができます。

## 例題

---

```
COUNTA(B2,D2,E4,E5,E6)
```

```
COUNTA(A1:G5)
```

## ⚙️ EXP

EXP ( value )

引数	型	説明
value	Number	評価する数値

### 説明

---

**EXP** ファンクションは、*value* 引数で指定した値を指数として自然対数の底( $e = 2.71828\dots$ ) をべき乗した数値を返します。このファンクションは数値データを受け取り、数値データを返します。

このファンクションは**LN** の逆関数であり、そのため $\text{EXP}(\text{LN}(x))$  は $x$ を返します。

### 例題

---

```
EXP(B3)
```

```
EXP(1) //2.17828...
```



## FALSE

FALSE ()

このコマンドは引数を必要としません

### 説明

---

**FALSE** ファンクションは論理値FALSE(0) を返します。

### 例題

---

```
NOT(FALSE) //TRUE
```

FINDCELL ( toFind , searchRange )

引数	型		説明
toFind	CellRef	→	探す値
searchRange	CellRef	→	検索するセル範囲

## 説明

---

**FINDCELL** ファンクションは、*searchRange* 引数で指定されたレンジ内で、*toFind* 引数で指定された値を探し、それが見つかったセルの参照を返します。この参照は表示することはできませんが、セル参照(*CellRef*) を引数として受け取る他の4D View Pro ファンクションに渡すことができます。

*toFind* 引数は、検索する値を実際に格納しているセルへの参照を格納する必要があります。

## 例題

---

C3のセルに10が格納されているとします:

FINDCELL(C3,A1:B9) //A1:B9 のセルレンジの中に値が実際に見つかった場合には10を返しますが、それ以外の場合にはエラーを返します。

$FV(i, n, m \{, f\})$

引数	型	説明
i	Number	→ 期間の利率
n	Number	→ 期間の数
m	Number	→ 複利の場合: 各期間の終わりの月々の支払い(負の値を使用) 単利の場合: 空の引数を渡します(例題参照)
f	Number	→ 単利の場合: 期間の終わりの最終的な値(負の値を使用)

## 説明

FV ファンクションは、次の計算を行います:

- 複利を用いた場合の最終的な値の合計
- あるいは、単利を用いた場合の最終的な値の合計

投資の間に得られた値を計算するためには、月々の支払いが期間の終わりに支払われる場合、 $m$  引数を渡して  $f$  引数を省略します。計算のためのフォーミュラは以下のようになります:

$$FV(i,n,m) = m \times \frac{(1+i)^n - 1}{i}$$

単利を用いた場合の最終的な値の合計を計算するためには、 $f$  引数を渡して、 $m$  引数のプレースホルダーには空の文字列(,,)を渡します。計算のためのフォーミュラは以下のようになります:

$$FV(i,n,,f) = f \times (1+i)^n$$

## 例題 1

**複利の場合:** 35ヶ月間に渡って、月々€1,000 を口座に貯金し、利率が年利12% つく場合を考えます。

$FV(12\%,35,-1000) // 41660.275603126$

## 例題 2

**単利の場合:** 上記の例題と同じ場合を考えます。

$FV(12\%,35,, -35*1000) // 1847986.69$

## IF

IF ( valueTest , valueTrue , valueFalse )

引数	型	説明
valueTest	式	→ 評価する値あるいは式
valueTrue	式	→ テストがの評価がTRUEだった場合に返される値
valueFalse	式	→ テストがの評価がFALSEあるいは0だった場合に返される値

### 説明

---

**IF** ファンクションは評価を実行し、与えられた二つの値のうちの一つを評価の結果に基づいて返します。このファンクションは数値(ブール)データを受け取り、あらゆるデータ型を返します。

*valueTest* 引数で指定した値が評価されます。数値データ型であるか、評価の結果が数値データである必要があり、このときゼロでない値はTRUE を意味し、0 はFALSE を意味します。以下の比較演算子のどれか一つを使用することができます:より大きい(>)、より小さい(<)、等しい(=)、等しくない(<>)。 *valueTest* 引数の結果に応じて、以下のように値を返します:

- 0 でない値(TRUE)場合、 *valueTrue* 引数が返されます。
- 0 (FALSE)の場合、 *valueFalse* 引数が返されます。

### 例題

---

営業成績の値を格納しているB1を評価したい場合を考えます。

```
IF(B1<200,"Declining result","Good result") //B1>200であれば"Good result" と表示
```

## 🔧 INDIRECT

INDIRECT ( cell | cellRange )

引数	型	説明
----	---	----

cell   cellRange	CellRef	➡ セルへの参照、参照として定義された名前、あるいはセルまたはレンジへのテキスト文字列参照
------------------	---------	---

### 説明

---

**INDIRECT** ファンクションは *cell* 引数で指定したセルの中身を返します。 *cell* 引数(必須)はセル参照であれば、\$A\$1 のような絶対参照や文字列を含めてどんなものでも可能です。**INDIRECT** ファンクションは、フォーミュラ自身を変えることなく、フォーミュラ内でのセルへの参照を変更する際などに使用してください。

**INDIRECT** ファンクションは、セルのレンジに対する内部参照を返すこともできます(この参照は表示することはできませんが、他の4D View Pro ファンクションに渡すことができます)。

### 例題

---

```
INDIRECT("A1") //A1のセルの中身を返します
```

```
COLUMN(INDIRECT("A1:A3")) //カラム1  
ROW(INDIRECT("A1:F1")) //行1
```

ISBLANK ( value )

引数	型	説明
value	CellRef, 式, Number, テキスト	→ 評価する値

## 説明

---

**ISBLANK** ファンクションはセルの中身、数値、テキスト、何らかの式が空かどうかをテストします。このファンクションは値が空のセル、あるいは空のデータを参照する場合にはTRUEを返します。

**注:** 空の文字列("")を格納しているセルも、空であると見なされます。

## 例題

---

```
IF(ISBLANK(A1);"Error";0) //A1のセルが空の場合には"Error"
```

```
ISBLANK(B1)
```

```
ISBLANK(A4-52)
```

```
ISBLANK(4) //FALSE
```

## LEN

LEN ( value )

引数	型	説明
value	テキスト, CellRef	→ 文字列の長さをカウントしたいテキスト

### 説明

---

**LEN** ファンクションは *value* 引数で指定された文字列内の文字数を返します。

*value* 引数には、長さを調べたいテキストを渡します。引数は文字列か、文字列の値のセルへの参照である必要があります。

**注:** スペースも文字数として数えられます。

### 例題

---

```
LEN("4D, Inc.") //8
```

LN ( value )

引数	型		説明
value	Number	→	評価する0より大きい数値

## 説明

---

**LN** ファンクションは *value* 引数で指定された値の自然対数を返します。このファンクションは数値データを受け付け、数値データを返します。

*value* 引数には、正の数値(0 より大きい値)を渡します。

**注:** **LN** は **EXP** の逆関数であり、そのため、LN(EXP(x)) の結果はxとなります。

## 例題

---

```
LN(10) //2.30258509...
```

```
LN(EXP(1)) //1
```



## LOOKUP

LOOKUP ( toFind , intervalToSearch , returnInterval )

引数	型	説明
toFind	CellRef, テキスト, Number, ブール	→ 検索する値
intervalToSearch	CellRef	→ 検索するセルのレンジ
returnInterval	CellRef	→ 対応する値を検索するセルのレンジ

### 説明

**LOOKUP** ファンクションは、*toFind* 引数で指定された値を引数で指定された*intervalToSearch* セルのレンジ内で探し、*returnInterval* 引数で指定したレンジで使用されている対応する値を返します。

*toFind* 引数は、検索する値を実際に格納しているセルへの参照を含んでいる必要があります。

*intervalToSearch* 引数で指定されたレンジは、昇順に並べ替えられている必要があります。**LOOKUP** は*toFind* 引数で設定した値と同じかそれより大きい最初の値を使用するからです。

*toFind* 引数の値が見つからない場合、*toFind* 引数より小さいか等しい値のうち、*intervalToSearch* 引数で指定したレンジ内の最大の値が返されます。

### 例題

	A	B	C
1		1	520
2		2	380
3		3	697
4		4	437
5		5	578
6		6	185
7			
8		3	
9	=LOOKUP(A8, B1:B6, C1:C6)		
10			

A8セルに入っている"3"という値が検索する値です。B1:B6 が検索する範囲となります。C1:C6 が返される範囲となります。B3 に検索している値が入っています。返される範囲に入っている対応する値はC3 のセルに入っていて、これは"697"なのでこれが返されます。

## MAX

MAX ( value {, value2 , ... , valueN} )

引数	型	説明
value	実数, 配列	→ 評価する数値あるいは数値の配列

### 説明

---

**MAX** ファンクションは *value* 引数で渡された値の中から最大値を返します。ファンクションは数値データを受け付け、数値データを返します。

*value* 引数には、評価する値を渡します。それぞれの引数は数値、あるいは数値の配列を渡すことができます。単一の配列(セルのレンジ)あるいは複数の配列(セルのレンジ)を渡すことができます。

配列あるいは参照がテキスト、論理値、あるいは空のセルを含む場合、**MAX** はそれらの値を無視します。しかしながら、ゼロ値のセルは計算に含まれます。

### 例題

---

```
MAX(A1,B2,C3,D4,E5)
```

```
MAX(A1:A9)
```

```
MAX(2,15,12,3,7,19,4) //19
```

MID ( value , startFrom {, numChars} )

引数	型	説明
value	テキスト, CellRef, 式	→ 抜粋したい文字列を含んでいるテキスト
startFrom	Number	→ テキスト内から取得する最初の文字の位置を表す数値
numChars	Number	→ 返す文字数

## 説明

**MID** ファンクションは *value* 引数で指定した文字列の中から、*startFrom* 引数で指定した位置から指定した数だけの文字を返します。*value* 引数にはテキストデータを受け付け、*startFrom* および *numChars* 引数は数値データを受け付けます。返されるデータはテキストデータです。

*value* 引数には、抜粋したい文字を含んでいるテキスト文字列を渡します。文字列、文字列を返す式、文字列を格納しているセルへの参照を渡すことができます。

*startFrom* 引数には、テキストから抜粋したいテキストの最初の文字の位置を表す数値を渡します。テキスト内の最初の文字を指定する値は1になります。整数値でない場合、数値は整数へと切り下げられます。数値、数値を返す式、あるいは数値を格納しているセルへの参照を渡すことができます。*startFrom* 引数の挙動は、次のような場合には変化します:

- *value* 引数の長さより大きい場合、空の文字列("")が返されます。
- *value* 引数の長さより小さいが、*startFrom* + *numChars* の合計が *value* 引数の長さを超える場合、テキストの終わりまでの文字が返されます。

*numChars* 引数には、*value* 引数のテキストから取得した文字数を渡します。整数値でない値を渡した場合、数値は整数へと切り下げられます。

## 例題

```
MID(B17,5,8)
```

```
MID("hello world",7,20) //"world"
```

## MIN

MIN ( value {, value2 , ... , valueN} )

引数	型		説明
value	実数, 配列	→	評価する数値あるいは数値の配列

### 説明

---

**MIN** ファンクションは、*value* 引数の値の中から最も低い(最小の)値を返します。ファンクションは数値データを受け付け、数値データを返します。

*value* 引数には、評価する値を渡します。それぞれの引数は数値、あるいは数値の配列(単一の配列(セルのレンジ)あるいは複数の配列(セルのレンジ))を渡すことができます。

配列あるいは参照がテキスト、論理値、あるいは空のセルを含む場合、**MIN** はそれらの値を無視します。しかしながら、ゼロ値のセルは計算に含まれます。

### 例題

---

```
MIN(2,15,12,3,7,19,4) //2
```

## ⚙️ NOT

NOT ( logicalValue )

引数	型	説明
logicalValue	ブール, Number, 式	→ 評価する値

### 説明

---

**NOT** ファンクションは引数の論理値を逆転させます。

*logicalValue* 引数にはブールか数値を渡すことができます。値が0の場合、ファンクションはTRUE を返します。値が0以外の場合、ファンクションはFALSE を返します。

### 例題

---

```
NOT(A3)
```

```
NOT(D5>100)
```

```
NOT(0) //TRUE
```

```
NOT(TRUE) //FALSE
```

```
NOT(12) //FALSE
```

NOW ()

このコマンドは引数を必要としません

## 説明

---

**NOW** ファンクションはカレントの日付の時間を返します。これはDateTime オブジェクトを返します。

このファンクションは、このファンクションを格納しているスプレッドシートあるいはセルが再計算された際に更新されます。

## 例題

---

今日が2018年1月8日の11:25:42だった場合:

```
NOW() //1/8/2018 11:25:42
```

`NPER ( i , m , p {, f} )`

引数	型	説明
i	Number	→ 期間あたりの利率
m	Number	→ 期間の終わりに払う月々の支払い(負の値を使用) 取得した値:空の引数を渡します(例題参照)
p	Number	→ ローンのカレントの値
f	Number	→ 未来のローンの値(負の値を使用)

## 説明

---

**NPER** ファンクションはローンを返済し終わるのに必要な期間の数を返します。2種類のフォーミュラを使用することができます:

- 最初のフォーミュラは、月々の返済額が分かっている場合に使用します:

$$\text{NPER}(i,m,p) = \frac{\log((m - 1 \times p)/m)}{\log(1 + i)}$$

- 二つ目のフォーミュラは、返済総額が分かっている場合に使用します:

$$\text{NPER}(i,,f,p) = \frac{\log(f/p)}{\log(1 + i)}$$

## 例題 1

---

6,500 ユーロを年利10.5% で借り入れ、月々166.42 ユーロずつ返済する場合は考えます:

```
NPER(10.5%/12,-166.42,6500) //48
```

## 例題 2

---

3,000 ユーロを年利10.5% で借り入れ、月々の総支払い額が5,000 ユーロであると分かっている場合は考えます:

```
NPER(10.5%/12,,3000,-5000) //58
```

## OR

OR ( logicalValue {, logicalValue2 , ... , logicalValueN} )

引数	型	説明
logicalValue	ブール, 配列, 式	→ 評価する値

### 説明

---

**OR** ファンクションは全ての引数の評価がfalseであった場合にはFALSEを返します。少なくとも一つの引数がtrueであった場合にはTRUEを返します。

ファンクションはブール値、数値(0または1)、あるいは論理式(TRUE あるいはFALSE)を最大255個まで受け取ることができます。また値を個別に渡すのではなく、単一の配列を指定することもでき、この場合最大で255個までの配列を指定することができます。 *logicalValue* 引数を式として指定することもできます。

### 例題

---

```
OR(B3,B6,B9)
```

```
OR(D2:D12)
```

```
OR(TRUE,FALSE,FALSE) //TRUE
```

```
OR(TRUE()) //TRUE
```

```
OR(FALSE(),FALSE()) //FALSE
```

```
OR(1+1=1,2+2=5) //FALSE
```

```
OR(5+3=8,5+4=12) //TRUE
```



## **PI**

PI ()

このコマンドは引数を必要としません

### **説明**

---

**PI** ファンクションはPi (円周率)の値を3.1415926536として返します。

## PMT

PMT ( i , n , p )

引数	型		説明
i	Number	→	期間に対する利率
n	Number	→	期間の数
p	Number	→	ローンのカレントの値(負の値を使用)

### 説明

---

**PMT** ファンクションは、月々のローン支払いの値を返します。

**PMT** のフォーミュラは、以下のようになります:

$$\text{PMT}(i,n,p) = \frac{i}{1 - (1 + i)^{-n}}$$

### 例題

---

6,500 ユーロを借り入れ、10.5% の利率で48ヶ月かけて返済する場合は考えます:

```
PMT(10.5%/12,48,-6500) //166.42
```

PV ( i , n , m {, f} )

引数	型	説明
i	Number	→ 期間の利率
n	Number	→ 期間の数
m	Number	→ 複利の場合: 各期間の終わりの月々の支払い(負の値を使用) 単利の場合: 空の引数を渡します(例題参照)
f	Number	→ 単利の場合: 期間の終わりの最終的な値(負の値を使用)

## 説明

PV ファンクションは、次の計算を行います:

- 複利を用いたカレントの値の合計
- あるいは単利を用いたカレントの値の合計

複利を用いたカレントの値の合計を計算するためには、*m* 引数を渡して、*f* 引数を省略します。計算のためのフォーミュラは以下ようになります:

$$PV(i,n,m) = m \times \frac{1 - (1 + i)^{-n}}{i}$$

単利を用いたカレントの値の合計を計算するためには、*f* 引数を渡して、*m* 引数のプレースホルダーには空の文字列(,,)を渡します。計算のためのフォーミュラは以下ようになります::

$$PV(i,n,,f) = \frac{f}{(1 + i)^n}$$

### 例題 1

**複利:** 年利が12%(つまり月々1%)で、向こう5ヶ月に渡って月々€1,000 を支払うローンの場合を考えます。

```
PV(1%,5,-1000) //4853,4312393251
```

### 例題 2

**単利の場合:** 年利が12%(つまり月々1%)で、向こう5ヶ月に渡って月々€1,000 を支払うローンの場合を考えます。

**注:** シンタックス内の、",," にカンマが二つある点に注意してください。

```
PV(1%,5,,,-5*1000) //4757,328438033744
```

## RAND

RAND ()

このコマンドは引数を必要としません

### 説明

---

**RAND** ファンクションは0と0.9999999....の間のランダムな数字を返します。

### 例題

---

```
RAND()
```

```
RAND()*100
```

```
INT(RAND()*100)
```

## ⚙ RATE

RATE ( n , m , p { , f } )

引数	型		説明
n	Number	→	期間の数
m	Number	→	期間の終わりに払う月々の支払い(負の値を使用) 取得した値:空の引数を渡します(例題参照)
p	Number	→	ローンのカレントの値
f	Number	→	未来のローンの値(負の値を使用)

### 説明

---

**RATE** ファンクションは引数として渡された値から対応する利率を返します。二つのフォーミュラが使用可能です:

- 最初のフォーミュラは、期間の終わりに支払う月々の支払い額がわかっている場合に使用します。

$$i = \frac{m \times (1 - (1 + i)^{-n})}{p}$$

- 二つ目のフォーミュラは、取得した値が分かっている場合に使用します。

$$u_{n+1} = \frac{m \times (1 - (1 + u_n)^{-n})}{p}$$

### 例題 1

---

3,000 ユーロを借り入れ、月々の支払いが向こう5ヶ月にわたって1,000 ユーロである場合を考えます:

```
RATE(5,-1000,3000) //0.19
```

### 例題 2

---

2,800 ユーロを借り入れ、月々の支払いが向こう5ヶ月にわたって6,000 ユーロである場合を考えます:

```
RATE(5,,2800,-6000) //0.16
```

## ROUND

ROUND ( value , places )

引数	型		説明
value	Number	→	丸める数字
places	Number	→	小数点位置の数

### 説明

---

**ROUND** ファンクションは *value* 引数で指定された値を、*places* 引数で指定した小数点の位置を使用して丸め演算をします。

*value* 引数を使用して丸めたい数値を指定します。任意の数値データ(あるいは数値データを格納するセル参照)を渡すことができます。

**注:** 丸め演算の向きは上向きと下向きとになる場合があります。

*places* 引数を使用して、丸めを行う桁の位置を指定します。places は以下のルールに従います:

- places が正の値であるとき、数値は指定された桁数の小数の残すように丸められます。
- places が0であるとき、数値は最も近い整数へと丸められます。
- places が負の値であるとき、数値は小数点の左側へ指定された桁だけ移動されたところの最も近い0、00にまとめられます。

### 例題

---

```
ROUND(A3,-2)
```

```
ROUND(C4,B2)
```

```
ROUND(PI(),5) //3.14159
```

```
ROUND(29.2,-2) //29.2は100より0に近いので結果は0。
```

```
ROUND(-1.963,0) //-2
```

## ROW

ROW ( {reference} )

引数	型		説明
reference	CellRef	→	セルあるいはセルのレンジ

### 説明

---

**ROW** ファンクションは、*reference* 引数で指定した参照の行番号を返します。

*reference* 引数にはセルあるいはセルのレンジを渡すことができます。*reference* 引数が省略された場合、デフォルトの引数は、**ROW** ファンクションが置かれたセルの参照になります。

### 例題

---

```
ROW(B2) // 2
```

```
ROW(B1:B5) // 1
```

## SIN

SIN ( value )

引数	型	→	説明
value	実数		正弦を求める角度

### 説明

---

**SIN** ファンクションは`value` 引数で渡した値が指定する角度の正弦(sin)を返します。ファンクションは数値データを受け付け、数値データを返します。

`value` 引数には、余弦を取得したい実数(角度)を渡します。ラジアン単位で表現されている必要があります。角度が度単位であった場合、ラジアンに変換するためには $\text{PI}/180$ をかけます。

### 例題

---

```
SIN(B4)
```

```
SIN(30*PI()/180) //0.5
```



## SQRT

SQRT ( value )

引数	型		説明
value	Number	→	平方根を取得した0以上の数値

### 説明

---

**SQRT** ファンクションは *value* 引数で指定された値の正の平方根を返します。  
*value* 引数は0を含む正の数値である必要があります。そうでない場合、エラーが返されます。

### 例題

---

```
SQRT(B4)
```

```
SQRT(256) //16
```

## STDEV.P

STDEV.P ( value {, value2 , ... , valueN} )

引数	型	説明
value	Number, CellRef	評価する値

### 説明

**STDEV.P** ファンクションは数値の母集団に対する標準偏差を返します。標準偏差とは平均(平均値)に対する値の分散の度合いを表したものです。

*value*、*value2*...、引数には、母集団に対応する数値の引数を渡します。それぞれの引数には、セル、セルのレンジ、数値を渡すことができます。このファンクションは最大で255の引数を受け取ることができます。

**注:** データが母集団のサンプルでしかない場合、標準偏差を求めるには**STDEV** ファンクションを使用してください。

**STDEV.P** は"偏った"あるいは"n"メソッドを使用して計算されます。**STDEV.P** ファンクションは、以下のフォーミュラを使用します:

$$\sqrt{\frac{\sum (x - \bar{x})^2}{n}}$$

### 例題

```
STDEV.P(A1,B2,C3,D4,E5,F6)
```

```
STDEV.P(A1:A9)
```

```
STDEV.P(95,89,73,87,85,76,100,96,96) //結果は 8.8079649700473
```

## ⚙️ SUBSTITUTE

SUBSTITUTE ( *str* , *toReplace* , *replacement* {, *instance*} )

引数	型	説明
<i>str</i>	CellRef	→ 文字を置き換えたい文字列あるいはその文字列を格納しているセルへの参照
<i>toReplace</i>	テキスト	→ 置き換えたい文字列
<i>replacement</i>	テキスト	→ 既存の文字列を置き換える新しい文字列
<i>instance</i>	Number	→ 既存の文字列のうちどのオカレンスを置き換えるかを指定(省略時は全てのオカレンスを置き換え)

### 説明

---

**SUBSTITUTE** ファンクションは *str* 引数で指定されたテキスト内にある *toReplace* 引数で指定した文字列を *replacement* 引数の内容で置き換えて、編集されたテキストを返します。

デフォルトで、**SUBSTITUTE** は最初の *toReplace* 引数を置き換えます。 *instance* 引数を渡すことでどのオカレンスを置き換えるかを定義することができます。

*toReplace* 引数で指定された内容が見つからない場合、**SUBSTITUTE** は *str* 引数をそのまま返します。

### 例題

---

```
SUBSTITUTE("Hello You","You","World") //Hello World
```

## SUM

SUM ( value {, value2 , ... , valueN} )

引数	型		説明
value	Number, CellRef	→	総和をとる数値

### 説明

---

**SUM** ファンクションはセルあるいはセルのレンジの総和を返します。

引数には、評価する値を渡します。各引数には数値、あるいは数値の配列を渡すことができます(単一の配列(セルのレンジ)あるいは複数の配列(セルのレンジ))。

**注:** + 演算子を使用すると、定数によって渡された数値でない値と、参照によって渡された数値でない値の自動変換が行われます。**SUM** ファンクションでは定数によって渡された数値でない値に対しては自動変換を提供しますが、参照として渡された数値でない値に関しては無視します。

### 例題

---

```
SUM(A1,B7,C11)  
SUM(A1:A9)  
SUM(A2:A14,B2:B18,D12:D30)  
SUM(95,89,73,87,85,76,100,96,96) //797
```

## TAN

TAN ( value )

引数	型	→	説明
value	実数		正接を求める角度

### 説明

---

**TAN** ファンクションは、*value* 引数で指定された角度の正接(tan) を返します。このファンクションは数値データを受け付け、数値データを返します。

*value* 引数には、正接を取得したい実数(角度)を渡します。角度はラジアンで表現されている必要があります。角度が度単位であった場合、ラジアンに変換するためにはPI/180 をかけます。

### 例題

---

```
TAN(B3)
```

```
TAN(45*PI()/180) //1
```

## TEXT

TEXT ( value , format )

引数	型		説明
value	Number, CellRef	→	テキストにフォーマットする数値
format	テキスト	→	値に適用するフォーマット

### 説明

---

**TEXT** ファンクションは *format* 引数で指定したフォーマットに則った *value* 引数の数字で作成された文字列を返します。

*value* 引数には数値のデータあるいは数値データを含むセルへの参照を渡し、*format* 引数にはテキストの引数を渡します。

### 例題

---

```
TEXT(A1,"$0.00") //A1 に10が格納されていれば$10.00
```

```
TEXT(100,"0.00€") //100.00€
```

TODAY ()

このコマンドは引数を必要としません

## 説明

---

**TODAY** ファンクションはカレントの日時の、日付と時間を返します。これはDateTime オブジェクトを返します。このファンクションは、このファンクションを格納しているスプレッドシートあるいはセルが再計算された際に更新されます。

## 例題

---

カレントの日付が2018年1月8日の月曜日だった場合:

```
TODAY() //1/8/2018 0:00:00
```

## TRUE

TRUE ()

このコマンドは引数を必要としません

### 説明

---

**TRUE** ファンクションは論理値のTRUE(1)を返します。

### 例題

---

```
TRUE() //TRUE
```



## ⚙️ TYPE

TYPE ( value )

引数	型	説明
value	→	評価する値

### 説明

---

**TYPE** ファンクションは、*value* 引数で指定した値の型を数値で返します。

返される値は以下の通りです:

値の型	返される数値
数値	1
日付/時間オブジェクト	1
TimeSpanオブジェクト	1
テキスト	2
ブール	4
エラー	16
配列	64

この**TYPE** ファンクションは、例えば特定のセルに格納される値の型によって他のファンクションの実行が変化する場  
合などに使用します。**TYPE** ファンクションは異なる型のデータを受け取るファンクションを呼び出す際に特に有用で  
す。

### 例題

---

```
TYPE(G15)

TYPE(42) //1

TYPE("String") //2

TYPE(TRUE) //4
```

## VAR.P

VAR.P ( value {, value2 , ... , valueN} )

引数	型	説明
value	Number, 配列	分散を取得する値

### 説明

---

**VAR.P** ファンクションは、数値を使用する母集団全体に基づいた分散を返します。

*value* 引数には、評価する値を渡します。各引数には数値か、あるいは数字の配列(単一の配列(セルのレンジ)あるいは複数の配列(セルのレンジ))を渡すことができます。

**注:** このファンクションは渡された引数が母集団全体だとみなします。渡すデータが母集団のサンプルのみの場合、偏差を計算するには[VAR.S](#) ファンクションを使用してください。

### 例題










---

```
VAR.P(B3,C4,B2,D10,E5)
```

```
VAR.P(A1:A9)
```

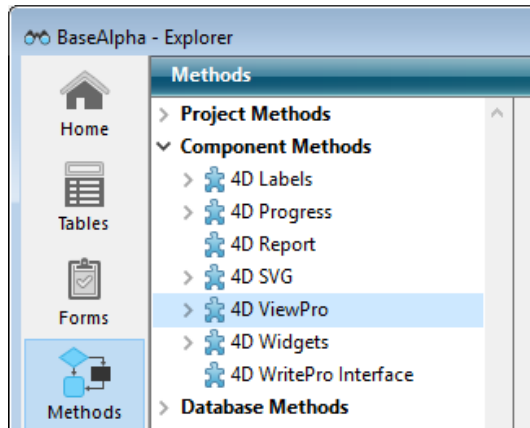
```
VAR.P(98,85,76,87,92,89,90) //39.265306122449
```

# 4D View Pro ランゲージ

-  4D View Pro コマンドについて
-  4D View Pro エリアの操作
-  On VP Ready フォームイベント
-  VP Convert from 4D View
-  VP EXPORT DOCUMENT
-  VP Export to object
-  VP IMPORT DOCUMENT
-  VP IMPORT FROM OBJECT
-  VP NEW DOCUMENT

## 🔗 4D View Pro コマンドについて

4D View Pro スプレッドシート機能はビルトインの4Dコンポーネントです。つまり、**4D ViewPro** の要素は、データベースエクスプローラーにおいて、メソッドページのコンポーネントメソッド内に表示されます：



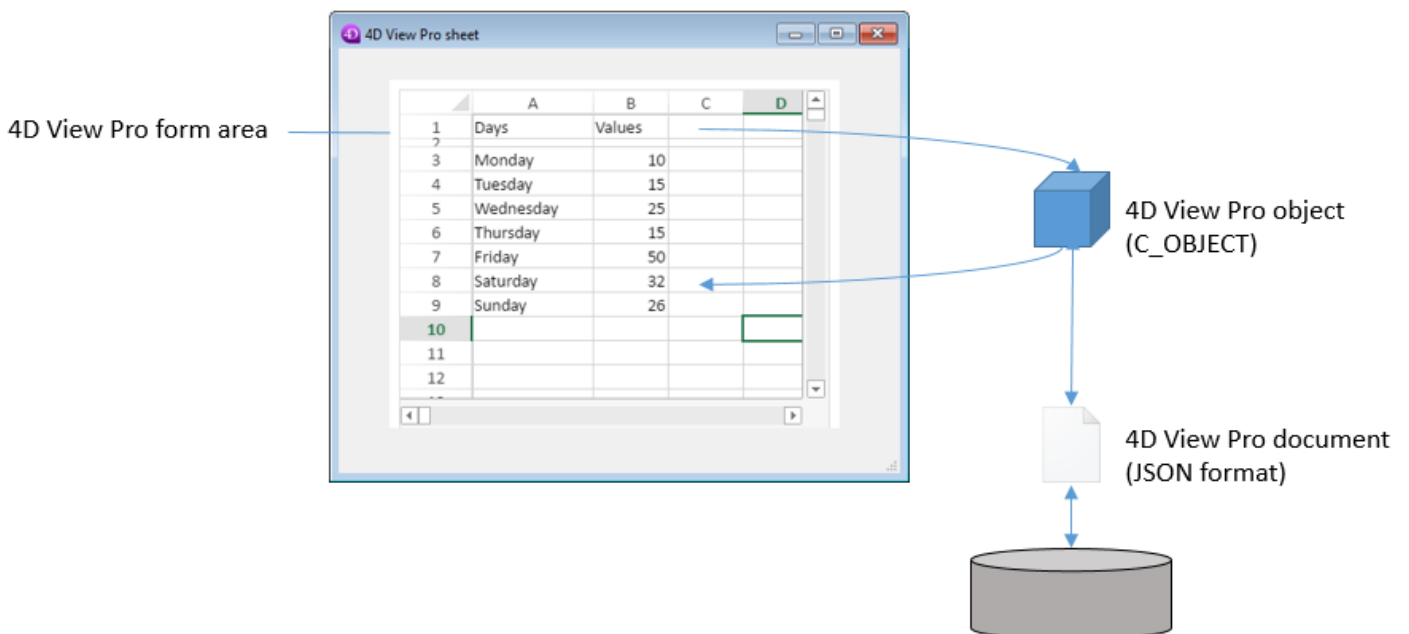
この項目を展開すると、すべてのコンポーネントコマンドを見ることができます。これらのコマンドは、ほかの4Dランゲージコマンドと同様に4Dメソッドエディターにて使用することができます。

## 4D View Pro エリアの操作

### アーキテクチャー

フォーム内に設置した4D View Pro エリアには、操作対象となる次の要素があります：

- **4D View Pro フォームエリア** (4Dフォームオブジェクト)：4D View Pro オブジェクトを格納して表示します。このエリアはオブジェクト名によって定義されます (プロパティリストの "オブジェクト名" プロパティ)。
- **4D View Pro オブジェクト (C\_OBJECT 型の変数または式)**：スプレッドシートの中身を格納します (後述参照)。4D View Pro オブジェクトを4D View Proフォームエリアに関連づけるには、プロパティリストの "変数あるいは式" プロパティを使います。
- **4D View Pro ドキュメント (.4vp ドキュメント)**：スプレッドシートの中身を JSON形式で保存します。



4D View Pro オブジェクトをフォームエリア内に読み込むと、その処理完了時に On VP Ready フォームイベントが4Dによって生成されます。エラーを避けるため、このエリアを対象とした4D View Proコードは、On VP Ready イベント内で実行する必要があります。

### 4D View Pro オブジェクト

4D View Pro オブジェクトはドキュメントを表現し、4D View Proコマンドによって管理されます。4D View Pro オブジェクトは次のプロパティを持ちます：

プロパティ	値の型	説明
version	Longint	コンポーネントの内部バージョン
dateCreation	Timestamp	作成日
dateModified	Timestamp	最終更新日
meta	Object	4Dデベロッパー専用の任意コンテンツ
spreadJS	Object	4D View Proコンポーネント専用

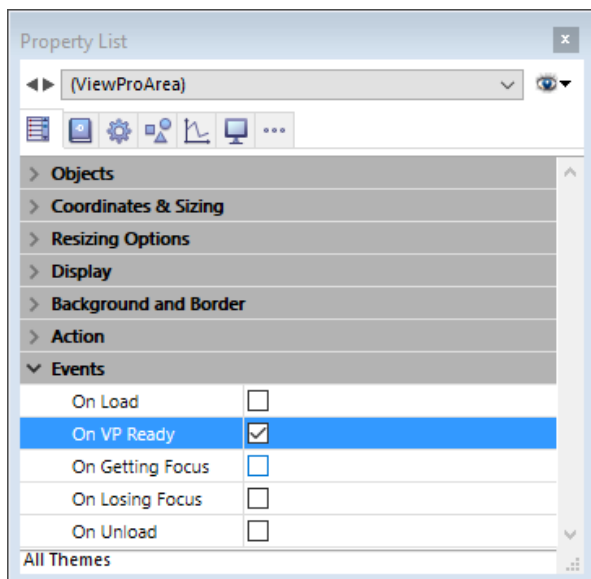
## On VP Ready フォームイベント

4D View Pro エリアを初期化するためのコード (値の読み込みなど) は当該エリアの On VP Ready フォームイベント内に書く必要があります。この **Form event** はエリアの読み込みが完了したときに生成されるため、有効なコンテキスト内でコードを実行するには、同イベントの利用が確実です。On VP Ready フォームイベントが生成される前に 4D View Pro コマンドを呼び出してしまうと、エラーが返されます。

**注:** 4D View Pro エリアは非同期的に4Dフォーム上に読み込まれます。したがって、標準の On load フォームイベントは4D View Pro エリアの読み込み完了を保証せず、4D View Proの初期化コードに利用することはできません。On VP Ready は常に On load の後に生成されます。

4D View Pro エリアに初期化コードを挿入するには:

1. 4D View Pro エリアのプロパティリストにて **On VP Ready** イベントをチェックします:



2. 4D View Pro エリア用の初期化コードを On VP Ready イベント内に書きます:

```
// フォームメソッドまたはエリアオブジェクトメソッド
Case of
  :(Form event=On VP Ready)
    ... // 初期化コード
End case
```

## ⚙️ VP Convert from 4D View

VP Convert from 4D View ( 4DViewDocument ) -> 戻り値

引数	型		説明
4DViewDocument	BLOB	→	4D View ドキュメント
戻り値	Object	↩	4D View Pro オブジェクト

### 説明

---

**VP Convert from 4D View** コマンドを使用すると、旧式の4D View ドキュメントを4D View Pro オブジェクトへと変換することができます。

**注:** このコマンドは、ご使用の環境に旧式の4D View プラグインがインストールされていなくても使用可能です。

*4DViewDocument* 引数には変換する4D Viewドキュメントを格納するBLOB変数やフィールドを渡します。コマンドは、4D View ドキュメントに元々保存されていた全ての情報を4D View Pro 属性へと変換した*object* を返します。

4D View ドキュメントのほとんどの情報は直接変換されます。サポートされていないプロパティ、あるいは変換の際に修正する必要があるプロパティについては、[4D View ドキュメントの変換](#) のページに一覧があります。

### 例題

---

BLOB に保存されている4D View エリアから4D View Pro オブジェクトを取得したい場合を考えます:

```
C_OBJECT($vpObj)
$vpObj:=VP Convert from 4D View($pvblob)
```

## ⚙️ VP EXPORT DOCUMENT

VP EXPORT DOCUMENT ( vpAreaName ; filePath )

引数	型	説明
vpAreaName	テキスト	→ 4D View Pro エリアフォームオブジェクト名
filePath	テキスト	→ ドキュメントのパス名

### 説明

---

**VP EXPORT DOCUMENT** コマンドは、*vpAreaName* で指定した 4D View Pro エリアに関連付けられている 4D View Pro オブジェクトを、*filePath* ごとにディスク上のドキュメントに書き出します。

*vpAreaName* には 4Dフォーム上に設置されたエリアのオブジェクト名プロパティを渡します。オブジェクト名が無効な場合にはエラーが返されます。

*filePath* には書き出すドキュメントのパスおよびファイル名を渡します。ドキュメント名のみを指定した場合、ドキュメントは4Dのストラクチャーファイルと同じ階層に保存されます。

*filePath* に拡張子を渡さなかった場合、自動的に 4D View Pro ドキュメントの拡張子 ".4vp" がドキュメント名に追加されます。

### 例題

---

"VPArea"の中身をディスク上の 4D View Pro ドキュメントに書き出します:

```
C_TEXT($docPath)
```

```
$docPath:="C:\\Bases\\ViewProDocs\\MyExport.4VP"
```

```
VP_EXPORT_DOCUMENT("VPArea";$docPath)
```

```
// MyExport.4VP がディスク上に保存されます
```



## ⚙️ VP Export to object

VP Export to object ( vpAreaName ) -> 戻り値

引数	型	説明
vpAreaName	テキスト	→ 4D View Pro エリアフォームオブジェクト名
戻り値	Object	↩ 4D View Pro オブジェクト

### 説明

---

**VP Export to object** コマンドは、*vpAreaName* で指定した 4D View Pro エリアに関連付けられている 4D View Pro オブジェクトを返します。このコマンドによって、たとえば 4D View Pro エリアを 4D データベースのオブジェクトフィールドに保存することができます。

*vpAreaName* には 4D フォーム上に設置されたエリアのオブジェクト名プロパティを渡します。オブジェクト名が無効な場合にはエラーが返されます。

4D View Pro オブジェクトについての詳細は [4D View Pro オブジェクト](#) を参照ください。

### 例題

---

4D View Pro エリアの "version" プロパティを取得します:

```
C_OBJECT($vpAreaObj)
C_LONGINT($vpVersion)
$vpAreaObj:=VP Export to object("vpArea")
// $vpVersion:=OB Get($vpAreaObj;"version")
$vpVersion:=$vpAreaObj.version
```

## VP IMPORT DOCUMENT

VP IMPORT DOCUMENT ( vpAreaName ; filePath )

引数	型	説明
vpAreaName	テキスト	→ 4D View Pro エリアフォームオブジェクト名
filePath	テキスト	→ ドキュメントのパス名

### 説明

**VP IMPORT DOCUMENT** コマンドは、*vpAreaName* に指定した 4D View Pro エリアに、*filePath*の4D View Pro ドキュメントを読み込んで表示させます。エリア内に挿入されていたデータがあれば、それらは読み込んだドキュメントによって置換されます。

*vpAreaName* には 4Dフォーム上に設置されたエリアのオブジェクト名プロパティを渡します。オブジェクト名が無効な場合にはエラーが返されます。

*filePath* には読み込むドキュメントのパスおよびファイル名を渡します。このコマンドで読み込むことができるのは 4D View Pro ドキュメント (拡張子 ".4vp") に限ります。ドキュメントがデータベースのストラクチャーファイルと同じ階層にある場合にはファイル名だけで指定することが可能ですが、それ以外の場合にはフルパスを指定する必要があります。

*filePath* 引数が無効だった場合や、対象ファイルが存在しなかったり、壊れたりしている場合には、エラーが返されません。

### 例題

フォームが開かれたときに、ディスク上に保存されているデフォルトの 4D View Pro ドキュメントを読み込みます:

```
C_TEXT($docPath)
If(Form event=On VP Ready) // 4D View Pro エリアの読み込みが完了しています
    $docPath:="C:\\Bases\\ViewProDocs\\MyExport.4VP"
    VP_IMPORT DOCUMENT("VPArea";$docPath)
End if
```

## ⚙️ VP IMPORT FROM OBJECT

VP IMPORT FROM OBJECT ( vpAreaName ; viewPro )

引数	型		説明
vpAreaName	テキスト	➡	4D View Pro エリアフォームオブジェクト名
viewPro	Object	➡	4D View Pro オブジェクト

### 説明

---

**VP IMPORT FROM OBJECT** コマンドは、*vpAreaName* に指定した 4D View Pro エリアに、*viewPro* の 4D View Pro オブジェクトを読み込んで表示させます。エリア内に挿入されていたデータがあれば、それらは読み込んだオブジェクトのデータで置換されます。

*vpAreaName* には 4Dフォーム上に設置されたエリアのオブジェクト名プロパティを渡します。オブジェクト名が無効な場合にはエラーが返されます。

*viewPro* には有効な 4D View Pro オブジェクトを渡します。このオブジェクトは手動で作成するほか、**VP Export to object** を使って取得することができます。4D View Pro オブジェクトについての詳細は **4D View Pro オブジェクト** を参照ください。

*viewPro* オブジェクトが無効な場合には、エラーが返されます。

### 例題

---

オブジェクトフィールドに保存してあるスプレッドシートを読み込みます:

```
QUERY([VPWorkBooks];[VPWorkBooks]ID=10)
VP IMPORT FROM OBJECT("ViewProArea1";[VPWorkBooks]SPBook)
```

## ⚙️ VP NEW DOCUMENT

VP NEW DOCUMENT ( vpAreaName )

引数	型	説明
vpAreaName	テキスト	→ 4D View Pro エリアフォームオブジェクト名

### 説明

---

**VP NEW DOCUMENT** コマンドは、*vpAreaName* に指定した 4D View Pro エリアに、新規のデフォルトドキュメントを読み込んで表示させます。エリア内に挿入されていたデータがあれば、それらは新規の空ドキュメントによって置換されます。

*vpAreaName* には 4Dフォーム上に設置されたエリアのオブジェクト名プロパティを渡します。オブジェクト名が無効な場合にはエラーが返されます。

### 例題

---

"myVPArea" フォームオブジェクトに空ドキュメントを表示します:

```
VP NEW DOCUMENT("myVPArea")
```

## 4D View Pro リファレンス - コマンドリスト (文字順)

P V

⚙️ PV