4D Chart

互換性に関する注意: 4D v13より、4D Chartは4Dから取り除かれ、外部プラグインとして提供されるようになりました。以前のバージョンの4Dではこのプラグインは内蔵されていました。あなたの4Dアプリケーションが4D Chartの機能を使用している場合、この外部化された4D Chartプラグインを、他の4Dプラグインと同様、4Dあるいはデータベースののpluginsフォルダーにインストールする必要があります。
4D Chartプラグインは今後改良される予定がありません。今後はグラフィカルな表現を行うためにSVG描画エンジンの利用を推奨します。この機能は**GRAPH**コマンドを使用したプログラミングでのみ利用可能です。

4Dを使用してデータベースやクリップボードのデータに基づく様々な種類のグラフを作成できます。これを行うために4Dから4D Chartプラグインを呼び出すことができます。

データベーステーブルのフィールドデータを使用して直接、あるいはメソッドの結果としてデータをグラフ化できます。

4Dに4D Chartをインストール (**プラグインやコンポーネントのインストール**参照) したら、グラフにデータベースレコードを割り当てたり、データが更新されるに伴いグラフを更新したりすることができます。

4D Chartは100以上のコマンドを4Dに追加します。4D Chartコマンドを使用すれば新規グラフの作成、動作の設定、ドキュメントを開いたり保存したりするなどの動作を自動化できます。

グラフ

ランゲージの概要

4D Chartを使用する場所

以下の場所で4D Chartを使用できます:

- フォーム上の4D Chartエリア
- 4D Chartプラグインウィンドウ
- 4D Chartオフスクリーンエリア

この節では、データベースでこれらを作成する方法について説明します。

フォーム上の4D Chartエリア

フォームに4D Chartエリアを配置できます。通常4D Chartエリアは入力フォーム上に配置され、ドキュメントを操作できるようにします。情報の表示や印刷に使用するために、4D Chartエリアを出力フォームに置くこともできます。4D Chartエリアがフォーム全体を占めることも、フィールドや他のフォームオブジェクトと空間を共有することもできます。

サイズは自由に設定できます。しかしエリアのサイズが300 x 150ピクセルより小さいと、エリア変数の名前がタイトルに付けられたボタンとして表示されます。ユーザがこのボタンをクリックするとエリアがフルスクリーンモードで表示されます。このメカニズムは*CT SET ENTERABLE* コマンドで無効にできます。

プラグインオブジェクトエリアを使用してフォーム上に4D Chart エリアを作成します。4Dにおいてプラグインエリアオブジェクトは、ボタンや入力可能エリア、スクロールエリアと同様、アクティブオブジェクトタイプのひとつです。プラグインエリアに関する情報は4D Design Referenceと4D Language Referenceマニュアルを参照してください。

フォーム上のドキュメントを参照する必要がある場合、プラグインエリアオブジェクトを作成するときに使用したオブジェクト変数を使用します。

フォーム上に4D Chartエリアを配置することに関する更なる情報は**4D Chartドキュメントの参照**を参照してください。

4D Chart プラグインウィンドウ

Open external window コマンドを使用してプラグインウィンドウを開き、空の4D Chartドキュメントを表示できます。このコマンドに関する情報は、4D Language Referenceマニュアルを参照してください。

Open external window は新しいウィンドウを開き、*plugInArea* 引数でサポートされる4Dプラグインエリアを表示します。またコマンドはエリアのID番号を返します。

4D Chartの場合、*plugInArea* 引数には "*_4D Chart*" を指定します。最初の下線と、“4D”と“Chart”の間のスペースを省略しないでください。この両方がシンタックス要素として必要です。

Open external window はモードレスウィンドウを作成します。これは同時に複数のアクティブウィンドウを開くことを可能にします。コマンドはユーザ入力を待ちませんので、一度に複数のアクティブウィンドウを開くことができます。ウィンドウ間をクリックで切り替え、最前面のエリアを編集できます。ウィンドウタイプがタイトルバーを持つ場合、コントロールメニューボックス (Windows) またはクローズボックス (Macintosh) が追加され、ユーザはウィンドウを閉じることができます。

プログラムでプラグインウィンドウを閉じるには、**Open external window**から返される値を4Dの**CLOSE WINDOW**コマンドに渡します。

例題

以下は**Open external window**を使用する例題です。コードはプラグインウィンドウを開いて、空の4D Chartドキュメントを表示します。

```
vChart:=Open external window(50;50;350;450;8;"Profit Margin Graph";"_4D Chart")
```

続いて、ドキュメントエリアを参照する必要があるときはvChartを使用します:

```
CT GET DEPTH(vChart;vObject;vHoriz;vVert)
```

4D Chart オフスクリーンエリア

オフスクリーンエリアはメモリに格納され、プログラマやユーザからは見えません。オフスクリーンを使用して、ユーザに表示する前や保存する前に、必要に応じてオリジナルに戻すなど、ドキュメントを更新できます。

再描画の必要がないため、4D Chartはオフスクリーンエリアでより高速に動作します。

CT New offscreen area コマンドを使用してオフスクリーンエリアを作成できます。*CT PICTURE TO AREA* コマンドを使用して、(4D Chartエリアを格納した) ピクチャフィールドを、(オフスクリーンの) 4D Chartエリアに配置できます。これらのコマンドについての説明は、エリアテーマのそれぞれのコマンドを参照してください。

オフスクリーンエリアの使用が終わったら、それが使用したメモリを解放するために、オフスクリーンエリアの削除を忘れないでください。データベースを閉じるときにクリアされていないオフスクリーンエリアがあると、4Dはエラーメッセージを表示します。

例題

以下の例題をプロジェクトメソッドに記述します。このコードは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します。フォーム上のボタンを使用して、ユーザは保存済みのドキュメントに戻ることができます。

```
Area:=CT New offscreen area
QUERY([Sales];[Sales]CustID=vCustID)
If(Records in selection([Sales]=1)
   CT PICTURE TO AREA(Area;[Sales]Profits_)
  `グラフをオフスクリーンエリアに格納
   MODIFY RECORD([Sales])
  `Salesレコードを更新
   CT DELETE OFFSCREEN AREA(Area)
  `オフスクリーンエリアが使用したメモリを解放
End if
```

入力フォーム上にボタンを作成して、以下のコードを割り当てます:

```
Profits:=CT Area to picture(Area;-2)
  `元のドキュメントを保持したオフスクリーンエリアを
  `Profits フォーム上のプラグインエリアにロード
```

4D Chartドキュメントの参照

4D Chartドキュメントをコントロールするためにコマンドを使用する際、エリアID番号を使用してドキュメントを指定する必要があります。エリアID番号は4D Chartの内部的なものであり、通常、変数に格納します。

4D Chartドキュメントは３種類の場所に配置できます: フォーム上のエリア、プラグインウィンドウ、またはオフスクリーンエリア。ドキュメントの場所に関わらず、4D Chartエリアを特定し、処理を行わせるにはエリアID番号が必要です。

エリアID番号とエリア変数

4D Chartでは、4D Chartエリア、プラグインウィンドウ、オフスクリーンエリアの場所を格納するために変数を使用します。動作を行わせたいエリアを参照するためには、エリアID番号を格納した変数を引数としてコマンドに渡します。

コマンドの説明中、*area* 引数はドキュメントエリアを特定する変数を指します。

2つのタイプの *area* 変数があります:

- プラグインエリアオブジェクト名
- プラグインウィンドウまたはオフスクリーンで作成した変数

プラグインオブジェクト名

フォーム上に作成した4D Chartエリアには変数を割り当てると、4Dは自動でこの変数をエリアを参照する変数として認識します。例えば*area* 引数にProfitsを渡すと、Profits変数が参照する4D Chartエリアを指定したことになります。

プラグインウィンドウとオフスクリーンエリア

Open external window や *CT New offscreen area*コマンド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すると、エリアID番号がこれらのコマンドから返されます。このIDを変数に格納し、他のコマンドでプラグインウィンドウやオフスクリーンエリアを参照するために使用します。

変数に値を格納するには、変数を置き、次に代入演算子 (:=) をコマンドの左に記述します。

以下の例題では4D Chart プラグインウィンドウを作成し、エリアID番号を**MyArea**変数に受け取っています:

```
MyArea:=Open external window(30;30;350;450;8;"Profits";"_4D Chart")
```

4D Chartデフォルトエリアの使用

デフォルトエリアはメモリ中のテンプレートで、すべての新しい4D Chartエリアのデフォルトの属性とプラグインウインドウを設定します。引数 *area* に -1 を設定すると、4D Chartエリアで実行可能なコマンドをデフォルトエリア上で実行することができます。メソッドを使用すると、他の任意のエリアで行う処理をデフォルトエリアで実行できます。

デフォルトエリアを使用すると4D Chartエリアに対する不要なコードの実行を軽減することができます。例えば、新規の4D Chartエリアとプラグインウインドウすべてをスクロールバーなしで表示させる場合は、エリアやプラグインウインドウで個別にスクロールバーをオフにする必要がありません。

ユーザは4D Chartエリアとプラグインウインドウの両方の属性を設定できます。フォーム上の新しい4D Chartエリアや新しいプラグインウインドウを表示した場合には、フォーム上で4D Chart エリアまたは新しいプラグインウインドウが開いている際はいつでも、デフォルトエリアがテンプレートとして自動的に使用されます。コードを実行する必要がまったくないので、デフォルトエリアはチャートエリアを高速にカスタマイズする方法を提供します。
デフォルトエリアをすべての新しい4D Chartエリアに適用したくない場合には、4D Chartエリア用のディスク上にテンプレートを作成するか、On Loadイベントが開始した際に適切なコードを配置することによってデフォルトエリアを上書きできます。ディスク上のテンプレートまたはOn Loadイベント時のコードは、デフォルトエリアよりも優先されます。フォームイベントについては4D Language Referenceマニュアルの**Form event**の説明を参照してください。

4D Chart オブジェクトを参照する

4D Chart ドキュメントは、グラフ、軸ラベル、入力したテキスト、ピクチャなどの異なるオブジェクトから構成されます。4D Chartを使用すると、プログラムでこれらのオブジェクトを処理できます。

この節では以下の内容に対して、どのようにプログラミングするかを説明します:

- オブジェクトへの参照
- オブジェクトの配置場所 (座標) の指定
- コマンドのスコープ指定

オブジェクトへの参照

4D Chartドキュメントのすべてのオブジェクトには、一意な番号が与えられています。この番号は、オブジェクトのIDであり、オブジェクトが作成されるときに割り当てられます。

これは、グラフの作成、ツールパレットでのオブジェクトの描画、クリップボードからのオブジェクトの貼り付け、複数のオブジェクトのグループ化、既存のオブジェクトの複製、フィールド参照の貼り付け等をするたびに、新しいIDが割り当てられることを意味します。オブジェクトIDは一意の為、IDはオブジェクトを参照する便利な方法です。オブジェクトIDがドキュメント内部で再使用されることはありません。たとえオブジェクトが削除されても、その番号はドキュメントがある間は "廃棄" されています。

オブジェクトIDは転送可能ではありません。ある4D ChartドキュメントのIDが5であるオブジェクトは、別のドキュメントに貼り付けられると必ずしも同じIDにはなりません。

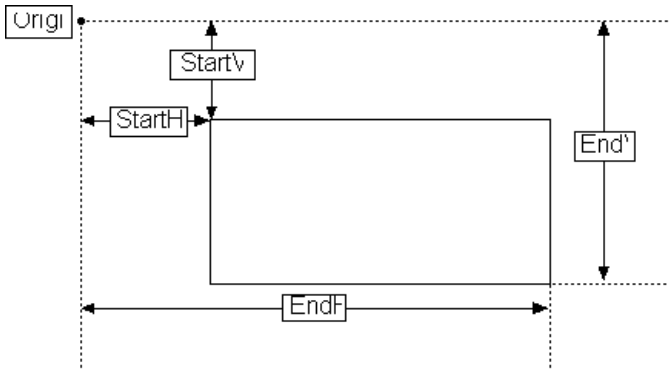
標準オブジェクト作成コマンドはすべて関数であり、結果のオブジェクトのIDを返します。パレットでユーザが作成したオブジェクトとは違い、コマンドで作成されたオブジェクトは自動的には選択されません。

*CT Get ID*関数を使用するとオブジェクトのIDを取得できます。

オブジェクトの配置場所 (座標) を指定する

オブジェクトの位置とサイズをそのオブジェクトの座標といいます。座標の説明または指定を行うすべてのコマンドは、ポイント単位で行います。

位置の説明や指定を行うコマンドは、原点に対して行います。原点は横のルーラと縦のルーラのゼロ点の交差です。次の図は、座標システムを表しています。



コマンドのスコープ指定

多くの4D Chartコマンドには*scope*という引数があります。*scope*は4D Chartドキュメントのどのオブジェクトまたはテキスト文字が、コマンドから影響を受けるのかを指定するものです。

次の表は*scope*の一般的な規則を説明しています。*scope*が与えられたコマンドにどのように影響するかは、コマンドごとの説明を参照してください。

スコープ	影響を受けるテキストまたはオブジェクト
>0	オブジェクトID
0	選択されたオブジェクト
-1	ドキュメントのすべてのオブジェクト
-2	デフォルト値
-3	テキストオブジェクトで選択された文字

チャートデータタイプ

次の表は、各項目、系列、数値のグラフ軸に割り当てることができるデータタイプの情報を示しています。

データ型	項目軸または系列軸?	数値軸?	数値軸と互換性のあるデータ型
文字	Yes	No	-
テキスト	Yes	No	-
数値	Yes	Yes	整数、倍長整数
整数	Yes	Yes	数値、倍長整数
倍長整数	Yes	Yes	数値、整数
日付	Yes	Yes	-
時間	Yes	No	-
ブール	Yes	No	-

Note: ピクチャやBLOBフィールドをグラフに使用できません。

4D Chartエラーの取り扱い

4D Chartは、コードの実行中に発生するエラーを管理する複数の方法を提供します。次に示す方法を任意に組み合せて使用してください。

- *CT Error*関数を使用すると、処理を実行した後にエラーをチェックできます。*CT Error*関数は4D Chartが実行した最後の処理の状態を表すエラーコードを返します。4D Chartエラーコードは、**4D Chart エラーコード**を参照してください。
- *CT ON ERROR*コマンドを使用すると、4D Chartエラーを管理するメソッドをインストールできます。メソッドを*CT ON ERROR* コマンドでインストールした後、4D Chartは4D Chartのエラーが発生したときにそのメソッドを呼び出します。
- 引数に値を返す関数やコマンドを実行した後には、エラーのチェックができます。関数の実行中にエラーが発生した場合、関数は-32000を返します。引数に値を返すコマンドが特定の引数に対する値を取り出しているときにエラーを検出した場合は、その引数に-32000を返します。

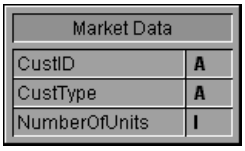
データベース内のレコードを使ってグラフを作成する (例題)

この節では*CT Chart selection*関数と*CT Chart data*関数を使用した2次元グラフと3次元グラフの作成例を紹介します。
配列を使用した2次元グラフと3次元グラフの作成方法は、**配列を使ってグラフを作成する (例題)**を参照してください。
各項目で以下の例を説明します:

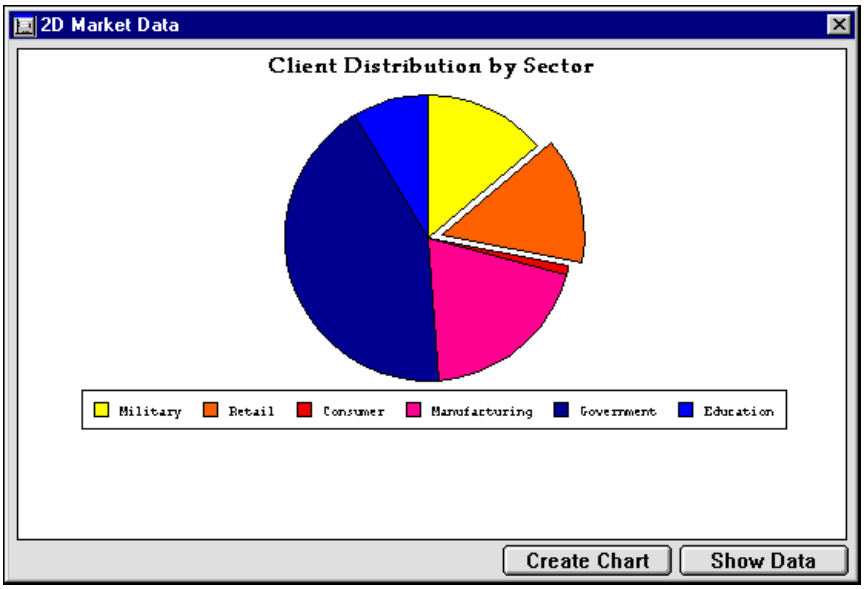
- 例に使用される状況の説明
- サンプルデータベースの構造
- サンプルデータを使用して、すべてメソッドで作成されたグラフ
- サンプルグラフの作成に使用されたコード

フィールドで定義された系列を使って2次元グラフを作成する

あなたがソフトウェア会社を経営し、政府、教育機関、小売りなど、多岐に渡る分野の顧客を抱えているとします。あなたのデータベースは顧客と営業の履歴を記録しています。
データベースでは、以下のテーブルを使用して情報を保存します:



4D Chartを使用して、以下のグラフを生成します。これは、各分野での全営業成績の割合を示したものです。



以下はサンプルグラフを作成するために使用された***GRAPH PROFILE***メソッドのコードです。

```
   `メソッド: GRAPH PROFILE
   `カテゴリー: 顧客タイプ
   `値: 購入されたユニット数

C_LONGINT($Left;$Top;$Right;$Bottom)
C_LONGINT($Area;$Chart;$Title;$Locate)
C_LONGINT($Left2;$Top2;$Right2;$Bottom2)

   `チャート化するレコードのセレクションを作成
ALL RECORDS([Market Data])
```

```
 `系列データフィールドの配列
ARRAY LONGINT($aFields;1)
$aFields{1}:=Field(->[Market Data]NumberOfUnits)

 `インターフェース要素を隠す
CT SET DISPLAY(Area;1;0) `メニューを隠す
CT SET DISPLAY(Area;2;0) `チャートツールを隠す
CT SET DISPLAY(Area;3;0) `オブジェクトツールを隠す
CT SET DISPLAY(Area;6;0) `スクロールバーを隠す
CT SET DISPLAY(Area;9;0) `ルーラーを隠す

 `円グラフを作成する
$Chart:=CT Chart selection(Area;6;1;1;Table(->[Market Data]);2;$aFields)

 `3番目のパイウエッジを切り離す
CT EXPLODE PIE(Area;$Chart;2;10)

  `凡例を配置する場所を設定する(下, 水平方向)
CT SET LEGEND ATTRIBUTES(Area;$Chart;1;0;0;0;8;0;0)

 `左上隅にチャートタイトルを追加する
$Title:=CT Draw text(Area;1;1;210;3;"Client Distribution by Sector")

 `タイトルをフォーマットする (Palatino, 14 ポイント, ボールド, 中央揃え, 黒)
$Color:=CT Index to color(16)
$Font:=CT Font number("Palatino")
CT SET TEXT ATTRIBUTES(Area;$Title;$Font;14;1;$Color;1)

 `エリア内のオブジェクトを中央に揃えるためにエリア境界を取得する
CT GET AREA BOUNDARY(Area;1;$Left;$Top;$Right;$Bottom)

 `チャートをウインドウサイズから50ポイント小さく変更する
CT SIZE(Area;$Chart;$Right-50;$Bottom-50)

 `チャートを中央揃えする
CT GET BOUNDARY(Area;$Chart;$Left2;$Top2;$Right2;$Bottom2)
$Locate:=(($Right-$Left)-($Right2-$Left2))/2
CT MOVE(Area;$Chart;$Locate;$Top2)

 `タイトルを中央揃えする
CT GET BOUNDARY(Area;$Title;$Left2;$Top2;$Right2;$Bottom2)
$Locate:=(($Right-$Left)-($Right2-$Left2))/2
CT MOVE(Area;$Title;$Locate;$Top2)

 `チャートを9ポイント分下げ、タイトルの下に表示されるようにする
CT GET BOUNDARY(Area;$Chart;$Left;$Top;$Right;$Bottom)
CT MOVE(Area;$Chart;$Left;$Top+9)

 `すべのオブジェクトの選択を解除する
```

```
CT SELECT(Area;-1;0)
```

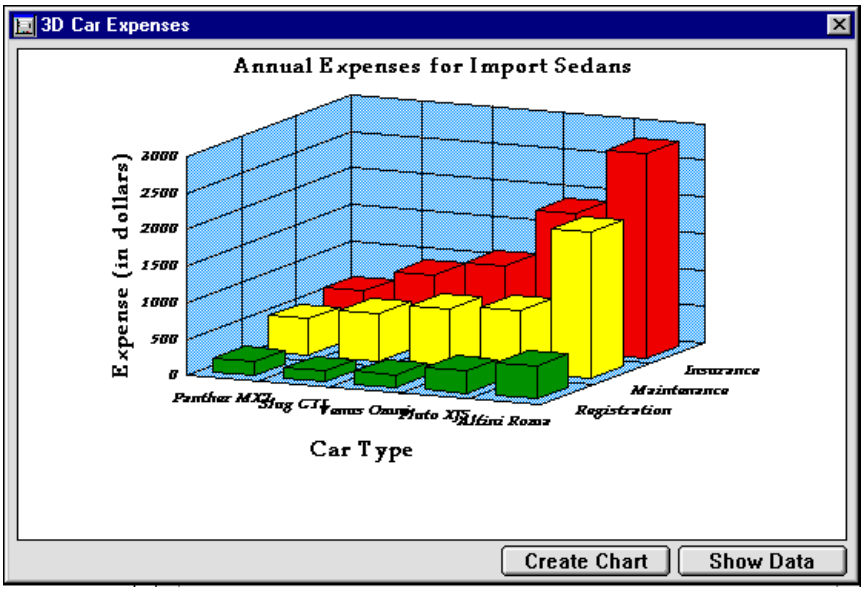
フィールドで定義された系列を使って3次元グラフを作成する

新車の市場においてデータベースを使用して手持ちの金で購入できる最高の車を決定しようとしています。5台の車に選択を絞り込み、それぞれの車で年間の登録、保険、メンテナンスの経費がいくらかかるかを比べたいと思っています。

以下のテーブルにデータを入力します:

Car Expenses	
Car	A
Cost	R
Registration	R
Insurance	R
Maintenance	R

収集した情報を解析するために、3次元グラフで情報をグラフ化することにしました。4D Chartを使用して以下のグラフを作成します:



GRAPH CARSメソッドはこのグラフを作成するために使われたものです。このメソッドは*CT Chart selection*コマンドを使用します。*CT Chart selection*を使用するときにはフィールドの配列を渡します。フィールド名は系列名になり、フィールドの値が数値軸上でグラフ化されます。この例ではRegistration、Insurance、Maintenanceの各フィールドは、系列と数値のために使用されています。

以下は***GRAPH CARS***メソッドです。

```
  `メソッド: GRAPH CARS
  `項目: 車種
  `系列: 経費カゴリー
  `数値: 経費

C_LONGINT($Left;$Top;$Right;$Bottom)
C_LONGINT($Area;$Chart;$Title;$Locate;$i)
C_LONGINT($Left2;$Top2;$Right2;$Bottom2)

  `チャート用のレコードセレクションを生成する
ALL RECORDS([Car Expenses])
ORDER BY([Car Expenses];[Car Expenses]Cost;>)

  `系列と数値のデータをフィールド配列に入れる
ARRAY LONGINT($aFields;3)
$aFields{1}:=Field(->[Car Expenses]Registration)
```

```
$aFields{2}:=Field(->[Car Expenses]Maintenance)
$aFields{3}:=Field(->[Car Expenses]Insurance)

  `インタフェース要素を隠す
CT SET DISPLAY(Area;1;0) `メニューを隠す
CT SET DISPLAY(Area;2;0) `チャートツールを隠す
CT SET DISPLAY(Area;3;0) `オブジェクトツールを隠す
CT SET DISPLAY(Area;6;0) `スクロールバーを隠す
CT SET DISPLAY(Area;9;0) `ルーラを隠す

  `3D棒グラフを作成する
$Chart:=CT Chart selection(Area;100;1;1;Table(->[Car Expenses]);1;$aFields)

  `スケールを設定する
CT SET REAL SCALE(Area;$Chart;0;0;0;0;0;3000;500;100)

 `凡例は表示しない
CT SET LEGEND ATTRIBUTES(Area;$Chart;0;0;-1;-1;0;0;0)

  `チャートタイトルを追加する
CT SET TITLE ATTRIBUTES(Area;$Chart;0;3;0;"Car Type")
CT SET TITLE ATTRIBUTES(Area;$Chart;1;0;0;"") `do not show
CT SET TITLE ATTRIBUTES(Area;$Chart;2;2;3;"Expense (in dollars)")

  `左上隅にタイトルを追加する
$Title:=CT Draw text(Area;1;1;300;3;"Annual Expenses for Import Sedans")

  `タイトルをフォーマットする (Palatino, 14 point, Bold, Center, Black)
$Color:=CT Index to color(16)
$Font:=CT Font number("Palatino")
CT SET TEXT ATTRIBUTES(Area;$Title;$Font;14;1;$Color;1)

  `1番目の系列の色を緑に設定する
$Color:=CT Index to color(10)
CT SET CHART FILL ATTRIBUTES(Area;$Chart;8;100;3;$Color)

  `2番目の系列の色を黄に設定する
$Color:=CT Index to color(2)
CT SET CHART FILL ATTRIBUTES(Area;$Chart;8;200;3;$Color)

  `3番目の系列の色を赤に設定する
$Color:=CT Index to color(4)
CT SET CHART FILL ATTRIBUTES(Area;$Chart;8;300;3;$Color)

  `すべのプロット矩形用に塗りつぶし属性を設定する
$Color:=CT Index to color(8)
For($i;1;3)
   CT SET CHART FILL ATTRIBUTES(Area;$Chart;1;$i;5;$Color)
End for
```

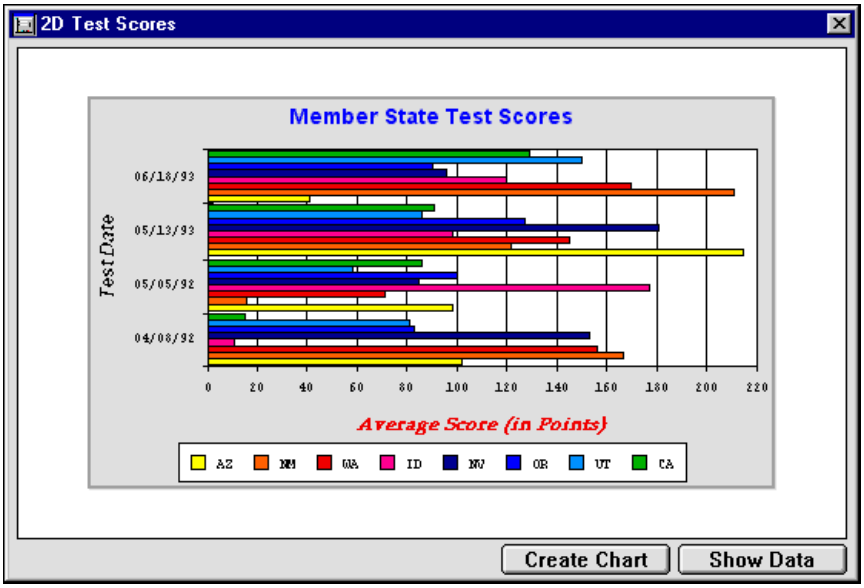
```
  `テキスト属性ラベルを設定する (Palatino, 9 point, bold italic)
$Font:=CT Font number("Palatino")
For($i;0;2)
   CT SET CHART TEXT ATTRIBUTES(Area;$Chart;4;$i;$Font;9;3;-1)
End for

  `タイトル用のテキスト属性を設定する (Palatino, 12 point, bold)
$Font:=CT Font number("Palatino")
For($i;0;2)
   CT SET CHART TEXT ATTRIBUTES(Area;$Chart;5;$i;$Font;14;1;-1)
End for

  `中央揃え用のエリア寸法を取得する
CT GET AREA BOUNDARY(Area;1;$Left;$Top;$Right;$Bottom)

  `チャートをウインドウサイズから50ポイント小さく変更する
CT SIZE(Area;$Chart;$Right-50;$Bottom-50)

  `チャートを中央揃えする
CT GET BOUNDARY(Area;$Chart;$Left2;$Top2;$Right2;$Bottom2)
$Locate:=(($Right-$Left)-($Right2-$Left2))/2
CT MOVE(Area;$Chart;$Locate;$Top2)

  `タイトルを中央揃えする
CT GET BOUNDARY(Area;$Title;$Left2;$Top2;$Right2;$Bottom2)
$Locate:=(($Right-$Left)-($Right2-$Left2))/2
CT MOVE(Area;$Title;$Locate;$Top2)

  `チャートを9ポイント下げる
CT GET BOUNDARY(Area;$Chart;$Left;$Top;$Right;$Bottom)
CT MOVE(Area;$Chart;$Left;$Top+9)

  `すべのオブジェクトの選択を解除する
CT SELECT(Area;-1;0)
```

レコードのデータによって定義された系列を使って2次元グラフを作成する

教育プログラムの西日本地区担当者であり、該当地区の生徒達の標準テストスコアを別のテスト日付と比較したいとします。データベースには、テスト日付と出身地にタグが付いて、該当地区の生徒のスコアが保存されています。

Test Scores	
Student ID	**I**
Test Date	**D**
Score	**I**
State	**A**

4D Chartを使用して、以下のグラフを作成します:



前述のグラフは***GRAPH SCORES 2D***メソッドで作成されています。以下は***GRAPH SCORES 2D***メソッドのコードです:

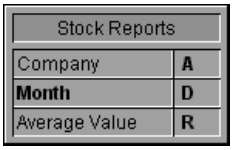
```
 `メソッド: GRAPH SCORES 2D
 `項目: テスト日付
 `系列: メンバーの地域
 `数値: テストスコア

C_LONGINT($Left;$Top;$Right;$Bottom)
C_LONGINT($Left2;$Top2;$Right2;$Bottom2)
C_LONGINT($Area;$Chart;$Title;$Locate;$Score;$Color;$Font;$Rect)

ALL RECORDS([Test Scores])
ORDER BY([Test Scores];[Test Scores]Test Date;>)

 `インタフェース要素を隠す
CT SET DISPLAY(Area;1;0) `メニューを隠す
CT SET DISPLAY(Area;2;0) `チャートツールを隠す
CT SET DISPLAY(Area;3;0) `オブジェクトツールを隠す
CT SET DISPLAY(Area;6;0) `スクロールバーを隠す
CT SET DISPLAY(Area;9;0) `ルーラを隠す

  `2D 棒グラフを作成する
$Chart:=CT Chart data(Area;2;1;1;1;Table(->[Test Scores]);2;4;3)

  `水平棒グラフにする
ARRAY LONGINT($aOptions;4)
$aOptions{1}:=1 `方向: 水平
$aOptions{2}:=0 `積み重ねなし
$aOptions{3}:=0 `重ね
$aOptions{4}:=50 `間隔
CT SET CHART OPTIONS(Area;$Chart;$aOptions)

  `軸タイトルを表示する
CT SET TITLE ATTRIBUTES(Area;$Chart;0;2;3;"Test Date")
CT SET TITLE ATTRIBUTES(Area;$Chart;2;3;0;"Average Score (in Points)")
```

```
  `項目軸タイトルをフォーマット (Helvetica, Black, Bold italic, 12 point)
$Color:=CT Index to color(16)
$Font:=CT Font number("Helvetica")
CT SET CHART TEXT ATTRIBUTES(Area;$Chart;5;0;$Font;12;3;$Color)

  `数値軸タイトルをフォーマット (Palatino, Red, Bold Italic, 12 point)
$Color:=CT Index to color(4)
$Font:=CT Font number("Palatino")
CT SET CHART TEXT ATTRIBUTES(Area;$Chart;5;2;$Font;12;3;$Color)

  `凡例位置を下部中央、横方向にする
CT SET LEGEND ATTRIBUTES(Area;$Chart;1;0;0;0;8;0;0)

  `チャートタイトルを左上隅に追加する
$Title:=CT Draw text(Area;1;1;350;3;"Member State Test Scores")

  `チャートタイトルをフォーマットする (Geneva, 14 point, Bold, Center, Blue)
$Color:=CT Index to color(7)
$Font:=CT Font number("Geneva")
CT SET TEXT ATTRIBUTES(Area;$Title;$Font;14;1;$Color;1)

  `カスタムスケールを使用する
CT SET REAL SCALE(Area;$Chart;0;0;0;0;0;220;20;5)
 `Get window dimensions to use for centering
CT GET AREA BOUNDARY(Area;1;$Left;$Top;$Right;$Bottom)

  `チャートをウインドウサイズから50ポイント小さく変更する
CT SIZE(Area;$Chart;$Right-50;$Bottom-50)

  `チャートを横方向中央揃えにする
CT GET BOUNDARY(Area;$Chart;$Left2;$Top2;$Right2;$Bottom2)
$Locate:=(($Right-$Left)-($Right2-$Left2))/2
CT MOVE(Area;$Chart;$Locate;$Top2)

  `タイトルを横方向中央揃えにする
CT GET BOUNDARY(Area;$Title;$Left2;$Top2;$Right2;$Bottom2)
$Locate:=(($Right-$Left)-($Right2-$Left2))/2
CT MOVE(Area;$Title;$Locate;$Top2)

  `チャートをタイトルから10ポイント分下げる
CT GET BOUNDARY(Area;$Chart;$Left;$Top;$Right;$Bottom)
CT MOVE(Area;$Chart;$Left;$Top+10)

  `チャートとタイトルに灰色の矩形で枠組みをする
$Rect:=CT Draw rectangle(Area;$Left-2;$Top2-2;$Right+2;$Bottom+2+10;0)
CT SET FILL ATTRIBUTES(Area;$Rect;3;CT Index to color(13))
CT SET LINE ATTRIBUTES(Area;$Rect;3;CT Index to color(15);1)

  `すべのオブジェクトを縦方向に中央揃えする
CT GET AREA BOUNDARY(Area;1;$Left;$Top;$Right;$Bottom)
```

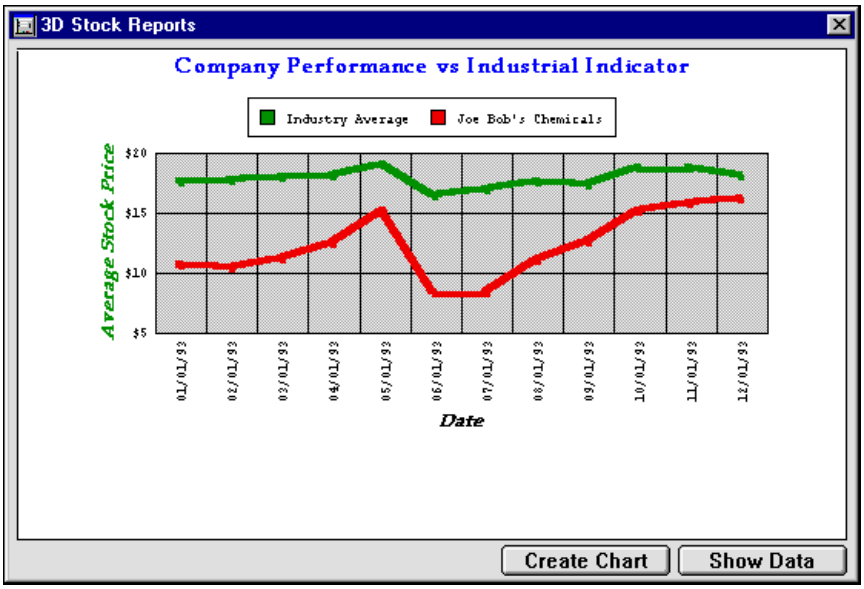
```
CT GET BOUNDARY(Area;-1;$Left2;$Top2;$Right2;$Bottom2)
$Locate:=(($Bottom-$Top)-($Bottom2-$Top2))/2
CT MOVE(Area;-1;$Left2;$Locate)

  `矩形を背面に送る
CT SELECT(Area;-1;0) `すべの選択を解除する
CT SELECT(Area;$Rect;1) `矩形を選択する
CT DO COMMAND(Area;24002) `背面に送る
CT SELECT(Area;-1;0) `すべの選択を解除する

  `すべの選択を解除する
CT SELECT(Area;-1;0)
```

レコードのデータにより定義された系列を使って3次元グラフを作成する

自分の会社の株式が同業他社と比べてどのような動きをしているのかを知りたいと仮定します。各企業の株式の終値を4Dデータベースで記録する場合もあります。データベースでは、株価は月平均で記録されています。



以下のグラフは、Joe Bob's Chemicals社の株価を業界の平均と比較しています。グラフ上のデータは何の操作もせずにデータベースから直接取られたものです。



STOCKS CHART メソッドは、データベースから直接取られたフィールド値を使用してグラフを生成します。データベースに格納された値は既に平均として計算されているので、追加の計算を行う必要はありません。代りに日々の値を記録していた場合には、月ごとの平均値を計算し、それを配列に格納してから *CT Chart arrays* 関数を使用してグラフを作成します。

月ごとの値を平均化する例は**配列を使ってグラフを作成する (例題)**を参照してください。

以下は***STOCKS CHART***メソッドです:

```
`メソッド: GRAPH STOCKS
`項目: 月
`系列: 企業と業界のインデックス
`値: 平均株価

C_LONGINT($Left;$Top;$Right;$Bottom)
C_LONGINT($Left2;$Top2;$Right2;$Bottom2)
```

```
C_LONGINT($Area;$Chart;$Title;$Locate;$Font;$Color)

  `チャート用のレコード選択を生成する
ALL RECORDS([Stock Reports])
ORDER BY([Stock Reports];[Stock Reports]Month;>)

  `インターフェース要素を隠す
CT SET DISPLAY(Area;1;0) `メニューを隠す
CT SET DISPLAY(Area;2;0) `チャートツールを隠す
CT SET DISPLAY(Area;3;0) `オブジェクトツールを隠す
CT SET DISPLAY(Area;6;0) `スクロールバーを隠す
CT SET DISPLAY(Area;9;0) `ルーラーを隠す

  `3D線グラフを作成する
$Chart:=CT Chart data(Area;101;2;1;1;Table(->[Stock Reports]);2;1;3)

  `スケールのセットアップ
CT SET REAL SCALE(Area;$Chart;0;0;0;0;5;20;5;1)

  `チャートを両方向に0度回転させる
CT SET 3D VIEW(Area;$Chart;0;0)

  `背景のカラーを設定する(グレー)
CT SET CHART FILL ATTRIBUTES(Area;$Chart;1;1;5;CT Index to color(15))

  `系列のカラーを設定する(緑; 赤)
CT SET CHART FILL ATTRIBUTES(Area;$Chart;8;100;3;CT Index to color(10))
CT SET CHART FILL ATTRIBUTES(Area;$Chart;8;200;3;CT Index to color(4))

  `系列の線のカラーを設定する(緑、赤、4ポイント)
CT SET CHART LINE ATTRIBUTES(Area;$Chart;8;100;3;CT Index to color(10);4)
CT SET CHART LINE ATTRIBUTES(Area;$Chart;8;200;3;CT Index to color(4);4)

  `チャートテキストの属性を設定する (Palatino, bold italic, 12 point)
$Font:=CT Font number("Palatino")
$Color:=CT Index to color(16) `black
CT SET CHART TEXT ATTRIBUTES(Area;$Chart;5;0;$Font;12;3;$Color)
$Color:=CT Index to color(10) `green
CT SET CHART TEXT ATTRIBUTES(Area;$Chart;5;2;$Font;12;3;$Color)

  `グラフ軸ラベルを表示する
CT SET LABEL ATTRIBUTES(Area;$Chart;2;3;0;"$###,##0")
CT SET LABEL ATTRIBUTES(Area;$Chart;0;3;3;"##/##/##")
CT SET LABEL ATTRIBUTES(Area;$Chart;1;0;0;"") `このラベルを隠す

  `項目軸と数値軸のタイトルを追加する
CT SET TITLE ATTRIBUTES(Area;$Chart;0;3;0;"Date")
CT SET TITLE ATTRIBUTES(Area;$Chart;2;2;3;"Average Stock Price")

  `凡例位置を上部中央、横方向に設定する
```

```
CT SET LEGEND ATTRIBUTES(Area;$Chart;1;0;0;0;7;0;0)

  `チャートタイトルを左上隅に追加する
$Title:=CT Draw text(Area;1;1;350;3;"Company Performance vs Industrial
Indicator")

  `チャートタイトルをフォーマット (Palatino, 14 point, Bold, Center, Blue)
$Color:=CT Index to color(7)
$Font:=CT Font number("Palatino")
CT SET TEXT ATTRIBUTES(Area;$Title;$Font;14;1;$Color;1)

  `エリアオブジェクトを中央揃えするためにエリア境界を取得する
CT GET AREA BOUNDARY(Area;1;$Left;$Top;$Right;$Bottom)

  `チャートのウインドウサイズを50ポイント少なく再定義する
CT SIZE(Area;$Chart;$Right-50;$Bottom-50)

  `チャートを中央揃えする
CT GET BOUNDARY(Area;$Chart;$Left2;$Top2;$Right2;$Bottom2)
$Locate:=(($Right-$Left)-($Right2-$Left2))/2
CT MOVE(Area;$Chart;$Locate;$Top2)

  `チャートタイトルを中央揃えする
CT GET BOUNDARY(Area;$Title;$Left2;$Top2;$Right2;$Bottom2)
$Locate:=(($Right-$Left)-($Right2-$Left2))/2
CT MOVE(Area;$Title;$Locate;$Top2)

  `チャートを10ポイント下げる
CT GET BOUNDARY(Area;$Chart;$Left;$Top;$Right;$Bottom)
CT MOVE(Area;$Chart;$Left;$Top+10)

  `すべのオブジェクトの選択を解除する
CT SELECT(Area;-1;0)
```

配列を使ってグラフを作成する (例題)

以下の2節では*CT Chart arrays*関数を使用して、2次元グラフと3次元グラフを作成する例を示します。

データベースのレコードを使用してグラフの作成をプログラムしたい場合は**データベース内のレコードを使ってグラフを作成する (例題)**を参照してください。

各項目では以下について説明します:

- 例に使用される状況の説明
- サンプルデータベースのストラクチャ
- サンプルデータを使用して、すべてメソッドで作成されたグラフ
- サンプルグラフの作成に使用されたコード

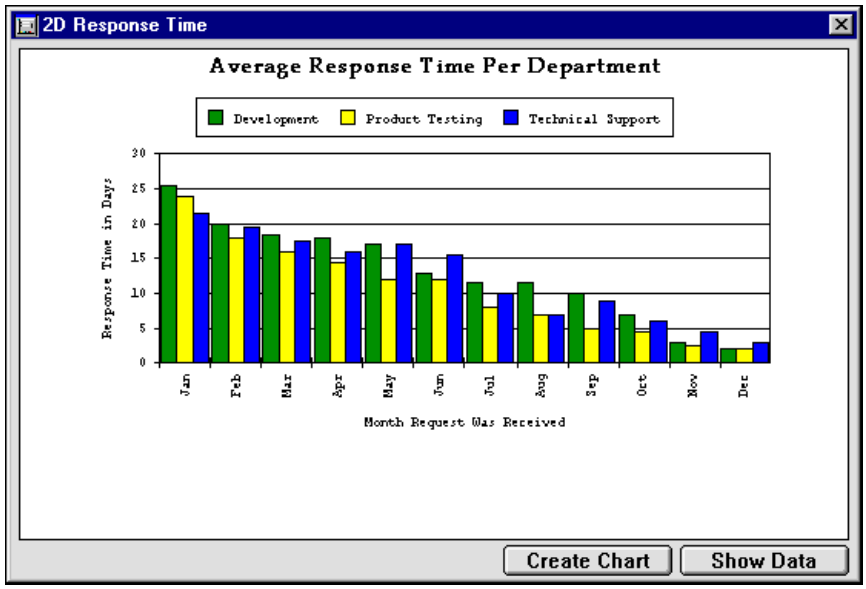
配列を使って2次元グラフを作成する

部品を製造する企業のエンジニアリング部門の新しいマネージャになったとします。あなたの新しい業務の1つとして、ご自分が管理する部署が顧客からの支援要求に応答するときの時間を短縮することがあります。

よって、各部署の応答時間を追跡するためにデータベースを設計することにしました。データベースの構造は以下の通りです:

Response Time	
Department	**A**
Request Date	**D**
Completion Date	D

収集した情報を解析するために、去年の各月について、各部署の平均応答時間 (完了日 - 要求日 / 要求数) をグラフ化することにしました。4D Chartを使用して、以下のグラフを生成します:



このグラフは***GR RESPONSE***メソッドで生成、フォーマットされます。このメソッドは配列にデータを入れ、その配列からグラフを生成してから、軸タイトルや系列のカラーなどのグラフの特定の外観をカスタマイズします。

メソッドの最初の部分は4Dコマンドと関数を使用して配列の作成とデータの指定を行っています。項目配列の内容はコードで指定され、系列配列の内容はデータベースから直接取り込まれ、数値配列はデータ操作の結果です。数値配列のサイズは項目配列のサイズを系列配列のサイズで乗算したものと同じです。

このメソッドは、レコードの選択を操作するためにセットを使用します。セットを作成した後は、必要とされるレコードの選択を検索によって変更し、何度でもレコードのオリジナルセットに戻すことができます。このメソッドでは、セットはグラフ全体のレコードの選択に使用されます。各月の各部署のレコードの検索の結果行われる選択で値が決まるので、選択は数値配列へのデータの指定中に変更されます。配列にデータが指定された後、4D Chartコマンドがグラフの作成とその機能の変更のために使用されます。

配列にデータが指定された後、4D Chartコマンドがグラフの作成とその機能の変更のために使用されます。
以下が***GR RESPONS***メソッドです：

```
  `メソッド: GRAPH RESPONSE
  `項目: 月
  `系列: 部門名
  `値: 作業期間の平均(日単位)

C_LONGINT($x;$y;$z;$Counter)
C_LONGINT($Left;$Top;$Right;$Bottom)
C_LONGINT($Left2;$Top2;$Right2;$Bottom2)
C_LONGINT($Area;$Chart;$Title;$Locate;$Duration;$Color;$Font)

  `項目配列を定義し、値を入れる
ARRAY STRING(3;$aCategories;12)
$aCategories{1}:="Jan"
$aCategories{2}:="Feb"
$aCategories{3}:="Mar"
$aCategories{4}:="Apr"
$aCategories{5}:="May"
$aCategories{6}:="Jun"
$aCategories{7}:="Jul"
$aCategories{8}:="Aug"
$aCategories{9}:="Sep"
$aCategories{10}:="Oct"
$aCategories{11}:="Nov"
$aCategories{12}:="Dec"

  `チャート用のレコードセレクションを生成する
  `後で使用するためにレコードをセットに格納する
QUERY BY FORMULA([Response Time];Year of([Response Time]Request Date)=1993)
CREATE SET([Response Time];"sChartData")

  `系列配列を部門名で定義し、データを入れる
ARRAY STRING(20;$aSeries;0)
DISTINCT VALUES([Response Time]Department;$aSeries)

  `数値の数でチャートを判断する
  `(数値の数=項目の数*系列の数)
  `数値配列を設定する
ARRAY REAL($aValues;12*Size of array($aSeries))

  `数値配列にデータを入れる
  `各部門で、月ごとの平均作業期間を検出する
$Counter:=0  `カウンタは数値の数を追跡する
For($x;1;Size of array($aSeries))  `部門の数だけループする
   For($y;1;12)  `12ヵ月分ループする
      $Counter:=$Counter+1
      QUERY SELECTION([Response Time];[Response Time]Department=$aSeries{$x})
      QUERY SELECTION BY FORMULA([Response Time];Month of([Response Time]Request Date)=$y)
```

```
      If(Records in selection([Response Time])>0)
         $Duration:=0 `期間をインクリメントするカウンタ
         For($z;1;Records in selection([Response Time]))
            GOTO SELECTED RECORD([Response Time];$z)
            $Duration:=$Duration+([Response Time]Completion Date-[Response Time]Request Date)
         End for
         $aValues{$Counter}:=$Duration/Records in selection([Response Time])
      End if
      USE SET("sChartData") `レコードの元のセレクションを復元する
   End for
End for

  `インターフェース要素を隠す
CT SET DISPLAY(Area;1;0) `メニューを隠す
CT SET DISPLAY(Area;2;0) `チャートツールを隠す
CT SET DISPLAY(Area;3;0) `オブジェクトツールを隠す
CT SET DISPLAY(Area;6;0) `スクロールバーを隠す
CT SET DISPLAY(Area;9;0) `ルーラーを隠す

  `棒グラフを作成する
$Chart:=CT Chart arrays(Area;2;1;$aCategories;$aSeries;$aValues)

  `項目軸と数値軸のタイトルを追加する
CT SET TITLE ATTRIBUTES(Area;$Chart;2;2;3;"Response Time in Days")
CT SET TITLE ATTRIBUTES(Area;$Chart;0;3;0;"Month Request Was Received")

  `系列のカラーを設定する(緑、黄色、青)
CT SET CHART FILL ATTRIBUTES(Area;$Chart;8;100;3;CT Index to color(10))
CT SET CHART FILL ATTRIBUTES(Area;$Chart;8;200;3;CT Index to color(2))
CT SET CHART FILL ATTRIBUTES(Area;$Chart;8;300;3;CT Index to color(7))

  `凡例位置を上部中央、横方向に設定する
CT SET LEGEND ATTRIBUTES(Area;$Chart;1;0;0;0;7;0;0)

  `チャートタイトルを左上隅に追加する
$Title:=CT Draw text(Area;1;1;300;3;"Average Response Time Per Department")

  `チャートタイトルをフォーマット (Palatino, 14 point, Bold, Center, Black)
$Color:=CT Index to color(16)
$Font:=CT Font number("Palatino")
CT SET TEXT ATTRIBUTES(Area;$Title;$Font;14;1;$Color;1)

  `中央揃えするためにチャートエリアの寸法を検出する
CT GET AREA BOUNDARY(Area;1;$Left;$Top;$Right;$Bottom)

  `チャートのウインドウサイズを50ポイント小さく再定義する
CT SIZE(Area;$Chart;$Right-50;$Bottom-50)

  `チャートを中央揃えする
CT GET BOUNDARY(Area;$Chart;$Left2;$Top2;$Right2;$Bottom2)
```

```
$Locate:=(($Right-$Left)-($Right2-$Left2))/2
CT MOVE(Area;$Chart;$Locate;$Top2)

  `チャートタイトルを中央揃えする
CT GET BOUNDARY(Area;$Title;$Left2;$Top2;$Right2;$Bottom2)
$Locate:=(($Right-$Left)-($Right2-$Left2))/2
CT MOVE(Area;$Title;$Locate;$Top2)

 `チャートを10ポイント下げる
CT GET BOUNDARY(Area;$Chart;$Left;$Top;$Right;$Bottom)
CT MOVE(Area;$Chart;$Left;$Top+10)

  `すべのオブジェクトの選択を解除する
CT SELECT(Area;-1;0)
```

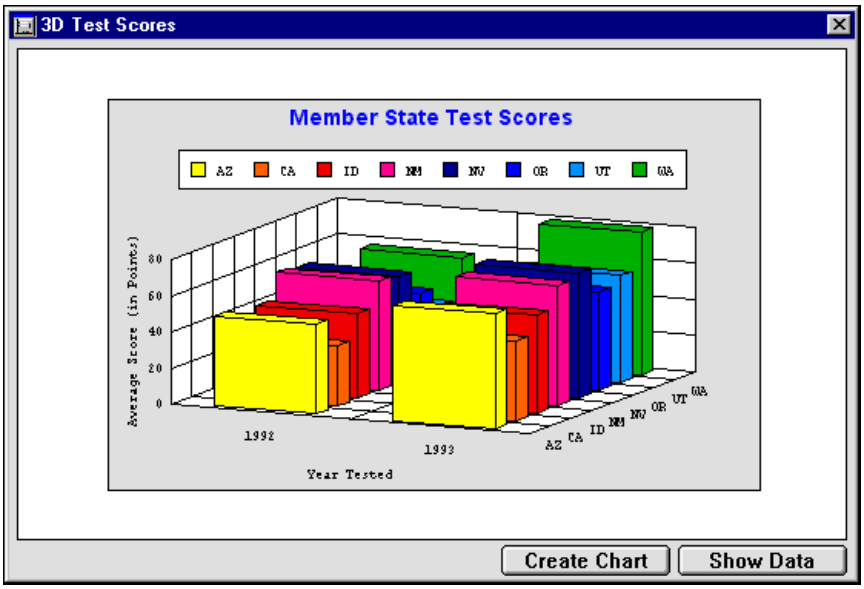
配列を使って3次元グラフを作成する

例えば、教育プログラムの西日本地区担当者であり、該当地区の生徒達のテストスコアが向上したかどうかを判断したいと仮定します。データベースには、テスト日付と出身地にタグが付いて、該当地区の生徒のスコアが保存されています。

Note: これは**データベース内のレコードを使ってグラフを作成する (例題)**の**レコードのデータによって定義された系列を使って2次元グラフを作成する**で使用されたデータと同じです。

Test Scores	
Student ID	**I**
Test Date	**D**
Score	**I**
State	**A**

毎年複数のテスト日があるので、データベースから直接データをグラフ化することはできません。その代わりに、データを配列に入れるメソッドを作成し、各年を "項目"、各県を "系列" として使用することができます。



前述のグラフは***GRAPH SCORES 3D***メソッドを使用して生成されています。このメソッドは配列にデータを入れ、配列からグラフを生成した後、グラフの特定の外観をカスタマイズします。

項目配列の内容はコードに直接指定されます。系列配列の内容はデータベースから取られます。数値配列の内容は年ごとの各都道府県のテストスコアの平均です。数値配列のサイズは項目配列のサイズを系列配列のサイズで乗算したものと同じです。

4D Chartコマンドを使用すると、4D Chartメニューバー、スクロールバー、ルーラ、ツールパレットは隠されます。4D Chartエリアは入力不可に設定され、ユーザはそのエリアのオブジェクトを選択したり、それに対する変更をまったく行えなくなります。

以下は***GRAPH SCORES 3D***メソッドのコードです:

```
 `メソッド: GRAPH SCORES 3D
 `項目: 調査年
 `系列: メンバーの県
 `値: テストスコア

C_LONGINT($x;$y;$z;$Counter)
C_LONGINT($Left;$Top;$Right;$Bottom)
C_LONGINT($Left2;$Top2;$Right2;$Bottom2)
C_LONGINT($Area;$Chart;$Title;$Locate;$Score;$Color;$Font;$Rect)

  `項目配列を定義し、データを指定する
ARRAY STRING(4;$aCategories;2)
$aCategories{1}:="1992"
$aCategories{2}:="1993"

  `数値配列を設定する
ALL RECORDS([Test Scores])
ARRAY STRING(2;$aSeries;0)
DISTINCT VALUES([Test Scores]State;$aSeries)

  `値配列の次元
 `(グラフへの数値の数=項目*系列)
ARRAY REAL($aValues;2*Size of array($aSeries))

  `値配列を作成する
 `各県ごとに平均を算出する
$Counter:=0 `カウンタは数値を追跡する
For($x;1;Size of array($aSeries)) `県の数だけループする
   For($y;1;2) `2年間だけループする
      $Counter:=$Counter+1
      QUERY([Test Scores];[Test Scores]State=$aSeries{$x})
      QUERY SELECTION BY FORMULA([Test Scores];
      String(Year of([Test Scores]Test Date))=$aCategories{$y})
      If(Records in selection([Test Scores])>0)
         $Score:=0 `スコアの合計を保持する
         For($z;1;Records in selection([Test Scores]))
            GOTO SELECTED RECORD([Test Scores];$z)
            $Score:=$Score+[Test Scores]Score
         End for
 `スコアの平均を出す
         $aValues{$Counter}:=$Score/Records in selection([Test Scores])
      End if
   End for
End for

  `セレクションを復元
ALL RECORDS([Test Scores])

  `インタフェース要素を隠す
CT SET DISPLAY(Area;1;0) `メニューバーを隠す
```

```
CT SET DISPLAY(Area;2;0) `チャートツールを隠す
CT SET DISPLAY(Area;3;0) `オブジェクトツールを隠す
CT SET DISPLAY(Area;6;0) `スクロールバーを隠す
CT SET DISPLAY(Area;9;0) `ルーラを隠す

  `3D 棒グラフを作成する
$Chart:=CT Chart arrays(Area;100;1;$aCategories;$aSeries;$aValues)

  `軸タイトルを表示または隠す
CT SET TITLE ATTRIBUTES(Area;$Chart;0;3;0;"Year Tested")
CT SET TITLE ATTRIBUTES(Area;$Chart;1;1;0;"State") `Hide title
CT SET TITLE ATTRIBUTES(Area;$Chart;2;2;3;"Average Score (in Points)")

  `凡例位置を上部中央、横方向に設定する
CT SET LEGEND ATTRIBUTES(Area;$Chart;1;0;0;0;7;0;0)

  `チャートタイトルを左上隅に追加する
$Title:=CT Draw text(Area;1;1;350;3;"Member State Test Scores")

  `チャートタイトルをフォーマット (Geneva, 14 point, Bold, Center, Blue)
$Color:=CT Index to color(7)
$Font:=CT Font number("Geneva")
CT SET TEXT ATTRIBUTES(Area;$Title;$Font;14;1;$Color;1)

  `中央揃えのためにウインドウの寸法を取得する
CT GET AREA BOUNDARY(Area;1;$Left;$Top;$Right;$Bottom)

  `チャートをウインドウサイズから50ポイント小さく変更する
CT SIZE(Area;$Chart;$Right-50;$Bottom-50)

  `チャートを横方向に中央揃えする
CT GET BOUNDARY(Area;$Chart;$Left2;$Top2;$Right2;$Bottom2)
$Locate:=(($Right-$Left)-($Right2-$Left2))/2
CT MOVE(Area;$Chart;$Locate;$Top2)

  `タイトルを横方向に中央揃えする
CT GET BOUNDARY(Area;$Title;$Left2;$Top2;$Right2;$Bottom2)
$Locate:=(($Right-$Left)-($Right2-$Left2))/2
CT MOVE(Area;$Title;$Locate;$Top2)

  `チャートをタイトルから10ポイント分下げる
CT GET BOUNDARY(Area;$Chart;$Left;$Top;$Right;$Bottom)
CT MOVE(Area;$Chart;$Left;$Top+10)

  `チャートとタイトルに灰色の矩形で枠組みをする
$Rect:=CT Draw rectangle(Area;$Left-2;$Top2-2;$Right+2;$Bottom+2+10;0)
CT SET FILL ATTRIBUTES(Area;$Rect;3;CT Index to color(13))

  `すべのオブジェクトを縦方向に中央揃えする
CT GET AREA BOUNDARY(Area;1;$Left;$Top;$Right;$Bottom)
```

```
CT GET BOUNDARY(Area;-1;$Left2;$Top2;$Right2;$Bottom2)
$Locate:=(($Bottom-$Top)-($Bottom2-$Top2))/2
CT MOVE(Area;-1;$Left2;$Locate)

  `矩形を背面にする
CT SELECT(Area;-1;0) `すべての選択を解除する
CT SELECT(Area;$Rect;1) `矩形を選択する
CT DO COMMAND(Area;24002) `背面にする
CT SELECT(Area;-1;0) `すべの選択を解除する

  `すべのオブジェクトの選択を解除する
CT SELECT(Area;-1;0)
```

CTエリア

- CT AREA TO AREA
- CT AREA TO FIELD
- CT Area to picture
- CT DELETE OFFSCREEN AREA
- CT FIELD TO AREA
- CT GET AREA BOUNDARY
- CT NEW DOCUMENT
- CT New offscreen area
- CT OPEN DOCUMENT
- CT PICTURE TO AREA
- CT SAVE DOCUMENT

CT AREA TO AREA

CT AREA TO AREA (source ; destination ; copyCode)

引数	型		説明
source	倍長整数	➡	コピー元となる4D Chartエリア
destination	倍長整数	⬅	コピーの受け取り先となる4D Chartエリア
copyCode	整数	➡	コピーする項目: 1 = 設定内容、2 = オブジェクト、3 = 両方

説明

*CT AREA TO AREA*コマンドは、4D Chartエリア*source*の内容を4D Chart エリア*destination*にコピーします。

転送する内容は、引数*copyCode*をもとにしています。

- *copyCode*が1の場合、表示オプション等のドキュメント設定が転送されます。
- *copyCode*が2の場合、*source*内のすべてのオブジェクトが*destination*に転送されます。
- *copyCode*が3の場合、オブジェクトとドキュメント設定の両方が*destination*に転送されます。

ドキュメント設定は転送されると、*destination*のドキュメント設定を置き換えます。オブジェクトが転送されるときには、*destination*内のオブジェクトに付加されます。*CT AREA TO AREA*コマンドは特にオフスクリーンエリアを操作するときに便利です。

例題

以下の例は、4D ChartエリアSalesChartの内容を新規オフスクリーンエリアにコピーします。

```
vOffscreen:=CT New offscreen area
CT AREA TO AREA(SalesChart;vOffscreen;3)
```

CT AREA TO FIELD

CT AREA TO FIELD (area ; scope ; numTable ; numField ; saveOption)

引数	型		説明
area	倍長整数	⇒	4D Chartエリア
scope	倍長整数	⇒	コマンドのスコープ: -2 = ドキュメント、-1 = すべて、0 = 選択されたオブジェクト、>0 = オブジェクトID
numTable	整数	⇒	テーブル番号
numField	整数	⇒	フィールド番号
saveOption	整数	⇒	エリアの内容を保存する際の方法: 1 = ピクチャのみ、2 = 日付のみ、3 = ピクチャと日付、-1 = 変更なし

説明

*CT AREA TO FIELD*コマンドは、引数*area* の内容を*table*と*field*で指定されたBLOBまたはピクチャフィールドにコピーします。

*CT AREA TO FIELD*コマンドは、リレートしているテーブルのフィールドにオブジェクトを格納したいときや、特定のオブジェクトだけを格納したいときに便利です。*CT AREA TO FIELD*は単にオブジェクトを*field*に割り当てるだけです。*table*のレコードは保存する必要があります。

*scope*はコピー内容を制御します。

オプション引数の*saveOption*は4D Chartエリア内のドキュメントの保存方法を決定します。

- *saveOption*が1の場合、ピクチャ (PICT) だけが保存されます。この設定をすると、オブジェクトを個別に操作することはできなくなります。
- *saveOption*が2の場合、4D Chartエリアのオブジェクトに関連するデータだけが保存されます。イメージは保存されたデータの情報を使用して後で再構築されます。この保存オプションは最速の方法であり、メモリの使用量が最も少ないものです。選択した保存方法に対するメモリが十分ないときには、別の方法を選択できるダイアログボックスが表示されます。
- *saveOption*が3の場合、イメージの再構築に使用されたピクチャと内部データの両方が保存されます。これはドキュメントを保存する際の通常の方法です。

例題

以下の例はarea内のオブジェクトを格納するリレートされたレコードを作成します。

```
  `レコードを作成しオブジェクトを格納する
CREATE RECORD([Objects])
  `リレート値を割り当てる
[Objects]Key:=[Charts]Name
  `オブジェクトのIDを取得する
$Temp:=CT Get ID(Area;-1;3)
  `オブジェクトをレコードにコピーする
CT AREA TO FIELD(Area;$Temp;3;2;1)
  `レコードを保存する
SAVE RECORD([Objects])
```

CT Area to picture

CT Area to picture (area ; scope) -> 戻り値

引数	型		説明
area	倍長整数	⇒	4D Chartエリア
scope	倍長整数	⇒	コマンドのスコープ: -2 = ドキュメント、-1 = すべて、0 = 選択されたオブジェクト、>0 = オブジェクトID
戻り値	ピクチャー	⇐	エリア内のオブジェクトの4Dピクチャ

説明

*CT Area to picture*コマンドは、引数*area*中のオブジェクトを4Dピクチャにして返します。

ピクチャに含まれるオブジェクトは、引数*scope*によって制御されます。

- *scope*が-2の場合、ドキュメント全体がコピーされます。これには表示オプション等のドキュメント設定も含まれます。
- *scope*が-1の場合、ドキュメント設定を除く*area*内のすべてのオブジェクトがコピーされます。
- *scope*が0の場合、選択されたオブジェクトだけがコピーされます。
- *scope*が0よりも大きい場合、それが特定オブジェクトIDと同じである必要があり、そのオブジェクトだけがコピーされます。

例題

以下の例は、新しいオフスクリーンエリアを開いて既存の配列から棒グラフを作成し、グラフをピクチャ変数に格納してから最後にオフスクリーンエリアを削除します。

```
Area:=CT New offscreen area
vChart:=CT Chart arrays(Area;2;2;aCategory;aSeries;aValues)
vPict:=CT Area to picture(Area;vChart)
CT DELETE OFFSCREEN AREA(Area)
```

CT DELETE OFFSCREEN AREA

CT DELETE OFFSCREEN AREA (area)

引数	型		説明
area	倍長整数	➡	4D Chart エリア

説明

*CT DELETE OFFSCREEN AREA*コマンドは、*CT New offscreen area*で作成された4D Chartオフスクリーンエリアを廃棄し、使用されたメモリを解放します。

引数*area*はフォーム上やウインドウ内のエリアではなくオフスクリーンエリアである必要があります。オフスクリーンエリアの使用を完了したときには常に*CT DELETE OFFSCREEN AREA*コマンドを実行してください。

例題

以下の例は*CT New offscreen area*とそれに対応する*CT DELETE OFFSCREEN AREA*をペアに実行する状況を示しています。

```
  `新しいオフスクリーンエリアを作成する
$NewArea :=CT New offscreen area
  `ここで何らかの処理を行う
  `オフスクリーンエリアを消去する
CT DELETE OFFSCREEN AREA($NewArea)
```

CT FIELD TO AREA

CT FIELD TO AREA (area ; numTable ; numField)

引数	型		説明
area	倍長整数	➡	4D Chart エリア
numTable	整数	➡	テーブル番号
numField	整数	➡	フィールド番号

説明

*CT FIELD TO AREA*コマンドは、引数*table*と*field*で指定されたBLOBまたはピクチャフィールドに含まれるドキュメントを*area*に配置します。

フィールド値は*table*のカレントレコードから取得されます。

*field*はBLOBまたはピクチャ型です。*field*には以前に保存された4D Chartドキュメントやピクチャが含まれています。*area*の内容は*field*で内容を置き換えられます。*field*が空の場合、このコマンドは何も行いません。

例題

以下の入力フォームのオブジェクトメソッドは、2番目のテーブルの5番目のフィールドに含まれる4D Chartドキュメントを開きます。

```
If(Form event=On Load)
   CT FIELD TO AREA(Area;2;5)
End If
```

CT GET AREA BOUNDARY

CT GET AREA BOUNDARY (area ; boundaryCode ; left ; top ; right ; bottom)

引数	型		説明
area	倍長整数	⇒	4D Chartエリア
boundaryCode	整数	⇒	境界コード: 0 = ドキュメント全体の境界、1 = 切り取られた境界
left	実数	⇐	エリアの左の境界を受け取る
top	実数	⇐	エリアの上の境界を受け取る
right	実数	⇐	エリアの右の境界を受け取る
bottom	実数	⇐	エリアの下の境界を受け取る

説明

*CT GET AREA BOUNDARY*コマンドは、引数*left*、*top*、*right*、*bottom*の各変数に*area*矩形の座標を返します。

*boundaryCode*が0の場合、*CT GET AREA BOUNDARY*コマンドはドキュメント全体の境界を返します。
*boundaryCode*が1の場合は、*CT GET AREA BOUNDARY*コマンドはフォーム上の4DChartエリアまたは4DChartプラグインウインドウの現在のサイズの境界を返します。

例題

以下の例は、既存のチャートエリアで複数の線を組み合せたジオメトリックオブジェクトを作成し、エリアの境界座標を取得してエリア内のオブジェクトを中央に配置します。

```
For($i;0;360;5)
   vLine:=CT Draw line(Area;50*Cos($i);50*Sin($i);0;0;0)
End for
CT GET AREA BOUNDARY(Area;1;$left;$top;$right;$bottom)
CT MOVE(Area;-1;(($right-$left)/2)-50;(($bottom-$top)/2)-50)
```

CT NEW DOCUMENT

CT NEW DOCUMENT (area)

引数	型		説明
area	倍長整数	⇨	4D Chart エリア

説明

*CT NEW DOCUMENT*コマンドは、引数*area*内のドキュメントの内容をクリアします。*CT NEW DOCUMENT*コマンドは確認ダイアログボックスが提示されない以外は、ファイルメニューから新規メニューを選択することと同じです。*CT NEW DOCUMENT*コマンドはすべてのオブジェクトと、ドキュメントサイズやルーラの目盛り指定等のすべてのドキュメント設定をクリアします。

警告: このコマンドを使用すると、*area*内の現在のドキュメントは保存されません。現在のドキュメントを保存する場合には、*CT NEW DOCUMENT*コマンドを実行する前に*CT SAVE DOCUMENT*コマンドを実行する必要があります。

例題

以下の例は、area内のドキュメントをクリアします。

```
CT NEW DOCUMENT(Area)
```

CT New offscreen area

CT New offscreen area -> 戻り値

引数	型		説明
戻り値	倍長整数		4D Chartオフスクリーンエリアの ID

説明

*CT New offscreen area*コマンドは4D Chartオフスクリーンエリアを作成し、そのエリアのIDを返します。*CT New offscreen area*コマンドから返される値は、4D Chartエリアを必要とする任意の4D Chartコマンドで使用できます。

例題

以下の例は、レコードを検索し、オフスクリーンエリアを作成し、レコードからエリアにドキュメントをコピーしてから、そのエリアを印刷します。

```
  `レコードを検索する
QUERY([Table3];[Table3]Field1 ="Level1")
  `新しいオフスクリーンエリアを作成する
$Offscreen :=CT New offscreen area
  `フィールドに格納されたドキュメントをコピーする
CT FIELD TO AREA($Offscreen;3;2)
  `エリアを印刷する
CT PRINT($Offscreen;0)
  `オフスクリーンエリアを削除する
CT DELETE OFFSCREEN AREA($Offscreen)
```

CT OPEN DOCUMENT

CT OPEN DOCUMENT (area ; document ; mode)

引数	型		説明
area	倍長整数	➡	4D Chartエリア
document	文字	➡	ドキュメントの名前、パスは最大255文字
mode	整数	➡	ドキュメント置き換えまたはドキュメント追加: 0 = 置き換え、1 = ドキュメントに追加する

説明

*CT OPEN DOCUMENT*コマンドは*document*を開き、その内容を*area*に 配置します。

*document*が空の文字列の場合、*CT OPEN DOCUMENT*コマンドはユーザがドキュメントの選択をできる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。

*document*に文字列が入っている場合には、指定されたドキュメントを開きま す。*document*が存在しない場合、*area*の内容は変更されず、*CT Error*コ マンドがシステムエラーコードを返します。

4D Chartはデータベースストラクチャを含むフォルダに*document*が あることを想定しています。データベースフォルダの外のドキュメントを開く場合、完全なパス名を指定してください。パス名に関する詳細は、*4D Language Reference*を参照してください。*document*が 既に開かれてていると、*CT Error*コマンドがシステムエラーコードを返します。

オプション引数の*mode*はドキュメントの開き方を制御します。*mode*は*document*が空の 文字列ではなく、かつ4D Chartドキュメントではないときにだけ使用されます。*mode*が 0の場合または指定されていない場合、*document*は*area*の内容を置き換えます。*mode*が 1の場合、*document*は*area*の 現在の内容に組み入れられます。

例題

以下の例はClient Typeフィールドの値に基づいて、異なるドキュメントを開きます。

```
Case of
  `業種が"販売店"の場合
    :([Client]Client type="Distributor")
  `"販売店"ドキュメントを開く
      CT OPEN DOCUMENT(Area;"Distributor") 
  `業種が"建設"の場合
    :([Client]Client type="Constructor")  
  `"建設"ドキュメントを開く
      CT OPEN DOCUMENT(Area;"Constructor")   
  `業種が"エンドユーザ"の場合
    :([Client]Client type="FinalClient")        
  `"エンドユーザ"ドキュメントを開く
      CT OPEN DOCUMENT(Area;"FinalClient")   
End case
```

CT PICTURE TO AREA

CT PICTURE TO AREA (area ; picture)

引数	型		説明
area	倍長整数	➡	4D Chartエリア
picture	ピクチャー	➡	ピクチャ

説明

*CT PICTURE TO AREA*コマンドは、引数*picture*に含まれるドキュメントを*area*に配置します。

*picture*は正しい4Dピクチャ式である必要があります。*area*の内容は*picture*で置き換えられます。*picture*が空の場合、このコマンドは何も行いません。

例題

以下のオブジェクトメソッドは、ピクチャフィールドから4D Chartエリアにグラフをコピーします。

```
$Name:=Request("Enter the name of the chart to load.")
If(OK=1)
   QUERY([Charts];[Charts]Label:=$Name)
   If(Records in selection([Charts])>0)
      CT PICTURE TO AREA(Area;[Charts]MyChart)
   End if
End if
```

CT SAVE DOCUMENT

CT SAVE DOCUMENT (area ; document ; type ; scope)

引数	型		説明
area	倍長整数	➡	4D Chartエリア
document	テキスト	➡	ドキュメントの名前 (パス付き)
type	文字	➡	ドキュメントのタイプ
scope	倍長整数	➡	コマンドのスコープ: 0 = すべてのオブジェクト、1 = 選択されたオブジェクト

説明

*CT SAVE DOCUMENT*コマンドは、引数*area*の内容を*document*に保存します。

*document*が空の文字列の場合、*CT SAVE DOCUMENT*コマンドはドキュメント名、タイプ、スコープをユーザが指定できる標準のファイル保存ダイアログボックスを表示します。*document*が空の文字列ではない場合、*CT SAVE DOCUMENT*コマンドは*type*タイプで*document*を保存します。

*document*が存在しない場合、 *CT SAVE DOCUMENT*コマンドがそのドキュメントを作成します。*document*が存在する場合、*CT SAVE DOCUMENT*コマンドはそのドキュメントを上書きします。

*type*が空の文字列の場合、標準の4D Chartドキュメントが作成されます。PICTとしてドキュメントを保存するには、*type*が"PICT"である必要があります。

オプション引数の*scope*は、*document*に保存するものを制御します。*document*が空文字列ではなく、かつドキュメントをPICTとして保存するときにだけ*scope*を使用してください。

デフォルトで、*document*はデータベースストラクチャを含むフォルダに保存されます。データベースフォルダ以外にドキュメントを保存する場合には完全なパス名を指定します。パス名に関する詳細は、*4D*ランゲージリファレンスを参照してください。

例題

以下の例は、4D Chartドキュメントを会社名と年度を名前とするドキュメントとして保存します。

```
  `年度を要求する
$Year :=Request("For what year?")
  `リクエストダイアログが受け入れられたら
If(OK=1)
  `ドキュメント名を連結する
   $SaveName :=[Company]Name+" "+$Year
  `ドキュメントを保存する
   CT SAVE DOCUMENT(Area;$SaveName;"")
End if
```

CTエリアコントロール

- CT DO COMMAND
- CT Error
- CT EVENT FILTER
- CT EXPERT COMMAND
- CT EXPERT MODE
- CT GET AREA PROPERTY
- CT Get display
- CT GET DOCUMENT SIZE
- CT GET PROPERTIES
- CT Last event
- CT MENU STATUS
- CT ON ERROR
- CT ON EVENT
- CT ON MENU
- CT SET AREA PROPERTY
- CT SET DISPLAY
- CT SET DOCUMENT SIZE
- CT SET ENTERABLE
- CT SET PROPERTIES

CT DO COMMAND

CT DO COMMAND (area ; command)

引数	型		説明
area	倍長整数	➡	4D Chartエリア
command	倍長整数	➡	コマンド番号

説明

*CT DO COMMAND*コマンドは、引数*command*によって指定されるメニューコマンドを実行します。ユーザが4D Chartメニューから選択したのと同じように、メニューコマンドは実行されます。コマンドを使用すると、ランゲージで同等のものがないアクションでも実行できます。*area*はオフスクリーンのエリアです。

*command*に指定できる値は、**コマンドコード**にリストされています。4D Chartの将来のバージョンでメニューが変更されたり、位置が変わってもこれらの数値は変わりません。

例題

以下の例は指定されたエリアのすべてのオブジェクトを選択し、複製します。

```
  `編集メニューからすべを選択を指定するのと同じ
CT DO COMMAND(Area;2009)
  `編集メニューから複製を選択するのと同じ
CT DO COMMAND(Area;2007)
```

CT Error

CT Error (message) -> 戻り値

引数	型		説明
message	文字	⇦	エラーメッセージ
戻り値	整数	⇨	4D Chartによって実行された最後の操作のステータス: 0 = 最後の処理はエラーを発生させていない、>0 = 最後の処理がエラーを発生させた

説明

*CT Error*コマンドは、4D Chartによって実行された最後の処理のステータスを表す番号を返します。

*CT Error*コマンドが0を返した場合、最後の処理はエラーを発生させていません。*CT Error*コマンドが0以外の番号を返した場合、最後の処理中にエラーが発生しています。

複数のエリアが同じフォーム上でアクティブな場合、エリアに関係なく、*CT Error*コマンドは最後のエラーを返します。

エラーコードの完全なリストは、**4D Chart エラーコード**を参照してください。

オプション引数*message*が*CT Error*へ渡されると、呼び出し後にエラーのテキストを含むテキスト変数となります。

例題

以下の例は、以前のコマンドでエラーが発生したかどうかをチェックします。

```
If(CT Error#0)
  `最後の処理がエラーを起こした
End if
```

CT EVENT FILTER

CT EVENT FILTER (area ; filter)

引数	型		説明
area	倍長整数	➡	4D Chartエリア
filter	倍長整数	➡	処理するイベント

説明

*CT EVENT FILTER*コマンドは、*area*のオブジェクトメソッドを実行させるイベント、または実行するイベントメソッドを指定します。

デフォルトで、4D Chartエリアに割り当てられたオブジェクトメソッドは、ユーザがエリア以外のオブジェクトを選択したときに実行されます。*CT EVENT FILTER*コマンドを使用して、メソッドを実行する他のイベントを指定することができます。更に、*CT ON EVENT*コマンドでインストールされたメソッドも実行できます。

*filter*にはイベントコードの加算値で、使用するイベントを指定します。下記にイベントコードを示します:

値	イベント
-1	すべてのイベント
0	イベントなし
1	エリア作成
2	エリア削除
4	エリアのアクティブ化 (クリックまたは前面になった)
8	エリアの非アクティブ化 (エリアがアクティブでなくなった)
16	オブジェクト作成 (作成、ペースト、複製)
32	オブジェクト削除 (削除、カット、クリア)
64	Ctrl+Click (Windows) または Command-click (Macintosh), オブジェクト上でなくてもよい
128	オブジェクトが移動された (整列、移動など)
256	オブジェクトのサイズが変更された (矢印キー、ドラッグなど)
1024	選択されたオブジェクトの変更
2048	ダブルクリック
4096	オブジェクトが作り直された

*area*に対して-1を指定すると、イベントフィルタはフォーム上、およびプラグインウインドウ内で新たに作成されるすべての4D Chartエリアに対するデフォルトフィルタになります。

例題

以下の例は、チャートエリアのオブジェクトメソッドによってトラップされたデフォルトイベントのリストに対して、**Ctrl+click** (Windows) / **Command+click** (MacOS) と **ダブルクリック**を追加します。

```
CT EVENT FILTER(Area;64+2048)
```

CT EXPERT COMMAND

CT EXPERT COMMAND (area ; command ; status)

引数	型		説明
area	倍長整数	⇨	4D Chartエリア
command	倍長整数	⇨	コマンド番号
status	整数	⇨	エキスパートモードのメニューのステータス: -1 = カレント値を返す、0 = 使用可、1 = 使用不可

説明

*CT EXPERT COMMAND*コマンドは4D Chartエクスパートモードのメニューを有効または無効にします。

*status*が0の場合、*command*で指定されるメニューはエクスパートモードで使用可能になります。
*status*が0より大きい場合、そのメニューは使用不可になります。
*status*が、値が-1である変数の場合、*CT EXPERT COMMAND*はメニューの現在のステータスを*status*に返します (0 = 使用可、1 = 使用不可) 。

*command*に指定できる値は、**コマンドコード**を参照してください。

メニューが*CT EXPERT COMMAND*コマンドで使用不可である場合でも、*CT DO COMMAND*を呼び出すことで実行できます。

例題

以下の例は**データベース**メニューの**フィールド貼り付け**メニューを使用不可にします。

```
CT EXPERT COMMAND(Area;6001;1)
CT EXPERT MODE(Area;1)
```

CT EXPERT MODE

CT EXPERT MODE (area ; mode)

引数	型		説明
area	倍長整数	⇒	4D Chart
mode	整数	⇒	エキスパートモードを切り替える: -1 = カレント値を返す、0 = オフ、1 = オン

説明

*CT EXPERT MODE*コマンドは、エキスパートモードをオンまたはオフにします。4D Chartがエキスパートモードのとき、4D Chart上の特定の項目が使用できなくなっている可能性があります。エキスパートモードは*CT EXPERT COMMAND*コマンドで設定されます。

*mode*が1の場合、エキスパートモードが起動します。エキスパートモードが起動されると、事前に*CT EXPERT COMMAND*で指定されたメニューは使用不可になります。
*mode*が0の場合、エキスパートモードはオフになります。
*mode*が-1の場合、*CT EXPERT MODE*は*mode*にそのモード (0 = オフ、1 = オン)を戻します。

例題

以下の例は、**ファイル**メニューの**フルウインドウ**メニューコマンドを使用不可にします。

```
CT EXPERT COMMAND(Area;1012;1)
CT EXPERT MODE(Area;1)
```

CT GET AREA PROPERTY

CT GET AREA PROPERTY (area ; property ; value)

引数	型		説明
area	倍長整数	⇒	4D Chart エリア
property	整数	⇒	プロパティの番号
value	整数	⇐	プロパティの値

説明

*CT GET AREA PROPERTY*コマンドは、4D Chart *area*の*property*の現 在値を*value*に取得するために使用します。

以下のプロパティを読み込みできます:

property	value	意味
0 = クライアント/サーバモードでテンプレートを保存	0	クライアント側
	1	サーバ側
1= クライアント/サーバモードでテンプレートをロード	0	クライアント側
	1	サーバ側

デフォルトで、テーンプレートはサーバマシン上で保存され読み込まれます。

CT Get display

CT Get display (area ; item) -> 戻り値

引数	型		説明
area	倍長整数	➡	4D Chartエリア
item	整数	➡	情報を取得する項目
戻り値	整数	⮌	0 = 指定された項目が表示されていない、1 = 項目が表示されている

説明

*CT Get display*を使用すると、4D Chartウインドウの特定の機能が表示されるかどうかわかります。

指定された項目が表示されないのであれば、*CT Get display*は0を返しま す。表示されるのであれば1を返します。

メ ニューバー、チャートツールパレット、オブジェクトツールパレット、スクロールバー、そしてルーラは、**CT SET DISPLAY**コマンドを使用して、ユーザモードまたはプログラムによって隠す、あるいは表示することができます。

以 下は、引数*item*のコードです。

コード	項目
1	メニューバー
2	チャートツール
3	オブジェクトツール
6	スクロールバー
9	ルーラ

例題

以下の例は、メニューバーが使用不可かどうかを確認し、使用不可でない場合は使用不可にします。

```
If(CT Get display(Area;1)=1)
   CT SET DISPLAY(Area;1;0)
End if
```

CT GET DOCUMENT SIZE

CT GET DOCUMENT SIZE (area ; width ; height)

引数	型		説明
area	倍長整数	⇒	4D Chartエリア
width	実数	⇐	ドキュメントの幅を受け取る (ポイント単位)
height	実数	⇐	ドキュメントの高さを受け取る (ポイント単位)

説明

*CT GET DOCUMENT SIZE*コマンドを使用してドキュメントエリアのサイズを取得します。4D Chartドキュメントは最大3500x3500ポイントまでを測定できます。

*width*はドキュメントエリアの幅のポイント数です。

*height*はドキュメントエリアの高さのポイント数です。

例題

以下の例では現在のドキュメントサイズを変更する前に、*CT GET DOCUMENT SIZE*コマンドを使用してそのサイズを取得します。

```
CT GET DOCUMENT SIZE(Area;$Width;$Height)
If($Width<2208)
   CT SET DOCUMENT SIZE(Area;2208;730)
End if
```

CT GET PROPERTIES

CT GET PROPERTIES (area ; printOrder ; changeAlert ; hotlinkType ; saveAlert)

引数	型		説明
area	倍長整数	➡	4D Chartエリア
printOrder	整数	⬅	プリント順: 0 = 行順、1 = 列順
changeAlert	整数	⬅	グラフタイプ変更メッセージ設定: 0 = メッセージなし、1 = メッセージあり
hotlinkType	整数	⬅	廃止
saveAlert	整数	⬅	保存メッセージ設定: 0 = メッセージなし、1 = メッセージあり

説明

*CT GET PROPERTIES*コマンドは、指定された4D Chartエリアに設定されているプロパティ情報を取得します。

*printOrder*は、ドキュメントのどのページが印刷されるのかの順序です。プリント順はドキュメントが印刷される順序だけに影響し、ページの方向には影響しません。

*changeAlert*はグラフのタイプを変更しようとしたときに警告ボックスが表示されるかどうかの設定です。ユーザはキャンセルするか変更を続行するかのオプションを選択できます。

*hotlinkType*はサポートされていませんので値を返しません。

*saveAlert*は、変更を保存していない4D Chartドキュメントをクローズするときに、警告ボックスをユーザに表示するかどうかの設定です。

- *saveAlert*が1の場合、4D Chartは変更を保存していない4D Chartドキュメントをユーザが閉じると、通常の警告ボックスを表示します。警告ボックスには変更を保存する、変更を保存しない、クローズせずにドキュメントに戻るオプションがあります。
- *saveAlert*が0の場合、4D Chart自身は変更を保存せず、ユーザに警告ボックスを表示しません。変更を保存するかどうかは開発者の責任になります。例外はフォーム上の、ピクチャフィールドに保存される4D Chartエリアです。このエリアの内容はピクチャフィールドに自動的に保存されます。

例題

以下は$POrder、$CAlert、$SAlertの変数にプリント順、グラフタイプ変更メッセージといったエリアのプロパティを返す例です。

```
CT GET PROPERTIES(Area;$POrder;$CAlert;$HType;$SAlert)
```

CT Last event

CT Last event (area) -> 戻り値

引数	型		説明
area	倍長整数	➡	4D Chartエリア
戻り値	倍長整数	⮌	エリアで実行した最後のイベントのコード

説明

*CT Last event*は、*area*で発生した最後のイベントのコードを返します。

*CT Last event*は4D Chart エリアのオブジェクトメソッド、または *CT ON EVENT*コマンドでインストールされたイベントメソッドで使用できます。*CT Last event*はオブジェクトまたはプロジェクトメソッドを実行させたイベントを特定します。*CT EVENT FILTER*コマンドと共に使用した場合、ユーザのアクションに基づいたアクションを実行することができます。

以下の表はイベントコードのリストです:

値	イベント
-1	すべてのイベント
0	イベントなし
1	エリア作成
2	エリア削除
4	エリアのアクティブ化 (クリックされた、または前面になった)
8	エリアの非アクティブ化 (エリアがアクティブでなくなった)
16	オブジェクト作成 (作成、ペースト、複製)
32	オブジェクト削除 (削除、カット、クリア)
64	Command-クリック (オブジェクトに対するものである必要はない)
128	オブジェクトが移動された (整列、移動など)
256	オブジェクトのサイズが変更された (矢印キー、ドラッグなど)
1024	選択済みでのオブジェクトでの変更
2048	ダブルクリック
4096	オブジェクトが形状が変更された

例題

以下の例で、*CT Last event*は*CT ON EVENT*でインストールされたメソッドで使用され、ダブルクリックを特定します。*CT ON EVENT*でインストールされたメソッドで使用されています。ユーザがグラフをダブルクリックすると、カスタムダイアログボックスが表示されるため、グラフに対して変更が行えます。

```
If(CT Last event(Area)=2048)  `ダブルクリックの場合
   If(CT Get object type(Area;0)=5) `グラフの場合
 `カスタム変更チャートダイアログボックスを表示する
      CHANGE CHART(Area;CT Get ID(Area;0;1))
   End if
End if
```

CT MENU STATUS

CT MENU STATUS (area ; command ; checked ; available ; name)

引数	型		説明
area	倍長整数	⇒	4D Chartエリア
command	倍長整数	⇒	コマンド番号
checked	整数	⇐	メニューがチェックされているか？ 0=チェックなし、1 = チェックあり
available	整数	⇐	メニューが使用可能かどうか？: 0 = 使用不可、1 = 使用可能
name	文字	⇐	メニューアイテムの名前を受け取る

説明

*CT MENU STATUS*コマンドは、*area*内で*command*で指定されるメニューに関する情報を*checked*、*available*および*name*変数に返します。

*command*に指定可能な値はにリストされています。

*available*が0の場合そのメニューは使用不可です。*available*が1の場合そのメニューは使用可能です。

*checked*が0の場合そのメニューはチェックされていません。*checked*が1の場合そのメニューはチェックされています。

*name*はメニューの名前です

例題

以下の例は、エリアが**参照表示**モードまたは**値表示**モードかどうかを確認するためにメニューをチェックします。エリアが**参照表示**モードの場合は、**値表示**モードをオンにします。

```
CT MENU STATUS(Area;6006;$Checked;$Available;$Name)
If($Name="参照表示")
   CT DO COMMAND(Area;6006)
End if
```

CT ON ERROR

CT ON ERROR (method)

引数	型		説明
method	文字	➡	実行するメソッド

説明

*CT ON ERROR*コマンドは、4D Chartエラー管理メソッドとして*method*をインストールします。エラー処理メソッドをインストールすると、4D Chartは4D Chartエラーが発生したときに*method*をコールします。

*method*が空の文字列の場合、メソッドはコールされず、エラー処理は4D Chartに返ります。

4D Chartは*method*を呼び出すときにエラー管理に使用できる3つのパラメータ ($1、$2、$3) を返します。

引数	型	説明
$1	倍長整数	倍長整数エラーが発生した4D Chartエリアを表す。エラーが特定の4D Chartエリアに対するものではない場合には、$1は0になる。
$2	倍長整数	倍長整数エラー番号を保持する。CT Errorの呼び出しと同じ。
$3	テキスト	エラーのテキストが含まれる。CT Errorの呼び出しと同じ。

データベースをコンパイルする予定がある場合は、以下のようにこれらのパラメータの型を宣言してください:

```
C_LONGINT($1;$2)
C_TEXT($3)
```

例題

以下の例は、エラー処理メソッドをインストールします。

```
CT ON ERROR("CHART ERROR")
```

以下は***CHART ERROR***メソッドです。このメソッドは$1をテストして、エラーがarea内で発生したかどうかを判断し、以下にエラー番号とメッセージを含む警告ボックスを提示します。

```
C_LONGINT($1;$2)
C_TEXT($3)

If($1=Area)
   ALERT("An error occurred in the 4D Chart area 'Area'.")
End if
ALERT("Error number "+String($2)+Char(13)+$3)
```

CT ON EVENT

CT ON EVENT (method)

引数	型		説明
method	文字	➡	Method to execute

説明

*CT ON EVENT*コマンドは、事前に指定されたイベントが発生した際に*method*メソッドを実行します。*method*を実行させるイベントは*CT EVENT FILTER*コマンドで指定されます。

*method*が空の文字列の場合メソッドは実行されません。イベントが発生するエリアにオブジェクトメソッドとイベントメソッドがある場合、オブジェクトメソッドが最後に実行されます。4D Chartエリアにはオブジェクトメソッドがないため、*CT ON EVENT*はプラグインウインドウの4D Chartエリアを使用する際、特に便利です。

4D Chartが*method*をコールすると、イベント管理用に使用できる4つのパラメータ ($1、$2、$3 と $4) を返します。

引数	型	説明
$1	倍長整数	イベントが発生した4D Chartエリアを表す。
$2	倍長整数	イベントコードを保持する。CT Last event へのコールと同じ。
$3	倍長整数	エリアが存在するフォームのテーブル番号。$3が-1だと、エリアはプラグインウインドウ内。
$4	倍長整数	エリアが自動保存されているフィールドの番号。$4が0だと、エリアは自動保存されません。

データベースをコンパイルする予定がある場合、以下のパラメータのタイプを指定する必要があります:

```
C_LONGINT($1;$2;$3;$4)
```

例題

以下の例は、イベントメソッドのインストールを示しています。プラグインウインドウを開き、イベントとして **Ctrl**+ クリック (Macintoshでは**Command**- クリック) を指定します。そして、イベントメソッド ***EventProc*** をインストールします。

```
 `プラグインウインドウを開く
vArea:=Open external window(20;50;400;350;0;"Chart";"_4D Chart")
 `EventProcメソッドをインストールする
CT ON EVENT("EventProc")
 `Ctrl+click でメソッドをコールする
CT EVENT FILTER(vArea;64)
```

CT ON MENU

CT ON MENU (area ; method)

引数	型		説明
area	倍長整数	➡	4D Chartエリア
method	文字	➡	コールするメソッドの名前

説明

ユーザモードまたはカスタムモードのいずれかでメニューコマンドがアクティブにされる度に、*CT ON MENU*コマンドは*method*を実行します。またメニューコマンドが*method*でコールされる限りにおいては、*CT DO COMMAND*コマンドを使用して呼び出すこともできます。

- コールされたメソッドは、3つのパラメータを返します:

引数	説明
$1	4D ChartエリアのIDを含む倍長整数
$2	メニュー番号を含む倍長整数
$3	モディファイアキーが押されたときの番号を含む倍長整数

- $3パラメータは以下のモディファイアキー (またはモディファイアキーの組み合わせ) の1つに対応します:

値	モディファイアキー
0	なし
1	Ctrlキー (Windows) または Commandキー (Macintosh)
2	Shiftキー
4	Altキー (Windows) または Optionキー (Macintosh)
8	Controlキー (Macintosh)

モディファイアキーの組み合わせが押されると、値は加算され、パラメータとして渡されます。例えば、値10は、メニューを選択しているときに、 **Shift** と **Control**キーの両方を押したことを示します。

データベースをコンパイルする予定がある場合、以下のようにこれらのパラメータの型を指定する必要があります:

```
C_LONGINT($1;$2;$3)
```

例題

以下の例は、***MenuProc***イベントメソッドを起動します。

```
CT ON MENU(Area;"MenuProc")
```

MenuProcメソッドは、メニューコマンドへのユーザのアクセスを制御します。**テンプレートとして保存**または**プロパティ**のいずれかのメニューが選択されると、ダイアログボックスが表示され、メニュー選択は無効になります。他のすべてのメニューは中断なく実行されます。

以下は、***MenuProc***メソッドのコードです。

```
C_LONGINT($1;$2;$3)
Case of
   :($2=1006)   `テンプレートとして保存
      ALERT("You cannot save templates.")
   :($2=2011)   `プロパティ
```

```
      ALERT("You do not have access to Properties.")
   Else
      CT DO COMMAND(vArea;$2)
End case
```

CT SET AREA PROPERTY

CT SET AREA PROPERTY (area ; property ; value)

引数	型		説明
area	倍長整数	➡	4D Chartエリア
property	整数	➡	プロパティ番号
value	整数	➡	プロパティの値

説明

*CT SET AREA PROPERTY*コマンドは、現在のセッションに対して、4D Chart *area*で*property*の*value*を変更します。

-1を*area*に渡すと、*CT SET AREA PROPERTY*コマンドはセッション中に後で読み込まれるすべての4D Chartエリアに適用されます。この場合このコマンドを**On Startupデータベースメソッド**で使用することをお勧めします。

プロパティは以下のように変更可能です:

property	value	意味
0 = クライアント/サーバモードでテンプレートを保存	0	クライアント側
	1	サーバ側
1= クライアント/サーバモードでテンプレートをロード	0	クライアント側
	1	サーバ側

デフォルトで、テーンプレートはサーバマシン上で保存され読み込まれます。

CT SET DISPLAY

CT SET DISPLAY (area ; item ; displayCode)

引数	型		説明
area	倍長整数	➡	4D Chartエリア
item	整数	➡	表示または隠す項目 (コード参照)
displayCode	整数	➡	項目の表示: 0 = 隠す、1 = 表示する、2 = 切り替え

説明

*CT SET DISPLAY*コマンドは、指定した項目が4D Chartウインドウで表示されるのか隠されるのかを設定します。

メニューバー、チャートツールパレット、オブジェクトツールパレット、スクロールバー、そしてルーラは*CT SET DISPLAY*コマンドを使用して隠す、あるいは表示することができます。

以下は引数*item*のコードです:

コード	項目
1	メニューバー
2	チャートツール
3	オブジェクトツール
6	スクロールバー
9	ルーラ

例題

以下の例は、4D Chart メニューバー、チャートツールパレット、オブジェクトツールパレット、そしてルーラを隠します。

```
CT SET DISPLAY(Area;1;0)
CT SET DISPLAY(Area;2;0)
CT SET DISPLAY(Area;3;0)
CT SET DISPLAY(Area;9;0)
```

CT SET DOCUMENT SIZE

CT SET DOCUMENT SIZE (area ; width ; height)

引数	型		説明
area	倍長整数	⇒	4D Chartエリア
width	実数	⇒	ドキュメントの幅 (ポイント単位)、-1 = 変更なし
height	実数	⇒	ドキュメントの高さ (ポイント単位)、-1 = 変更なし

説明

*CT SET DOCUMENT SIZE*コマンドは、ドキュメントエリアのサイズを設定します。4D Chartドキュメントのデフォルトサイズは588 x 768ピクセルです。このサイズは4D Chart プラグインエリア、あるいはプラグインウインドウのサイズによって異なります。4D Chartドキュメントは3500 x 3500ポイントまでを設定できます。

*width*はドキュメントエリアの幅のポイント数です。

*height*はドキュメントエリアの高さのポイント数です。

例題

以下の例は、On Startup データベースメソッドに設定されており、2208 x 1460 ポイントのすべての新しいドキュメントにデフォルトのドキュメントサイズを設定します。

```
CT SET DOCUMENT SIZE(-1;2208;1460)
```

CT SET ENTERABLE

CT SET ENTERABLE (area ; mode ; buttonMode)

引数	型		説明
area	倍長整数	➡	4D Chartエリア
mode	整数	➡	入力可または入力不可: 0 = 入力不可、1 = 入力可
buttonMode	整数	➡	0 = エリアが縮小サイズである場合ボタンとして表示 (デフォルト)、1 = ボタンへ切り替えない

説明

*CT SET ENTERABLE*コマンドは、*area*内のドキュメントへのアクセスを制御します。

*mode*が1の場合、*area*が使用可能になり、通常通りに動作します。

*mode*が0の場合、*area*は使用不可になります。

使用不可になったエリアに対してユーザは操作はできませんが、ランゲージで操作できます。エリアが使用不可になると、ユーザはそのエリアをスクロールして選択したオブジェクトをクリップボードコピーできます。ユーザは選択したものを変更する、あるいは4D Chartメニューまたはツールパレットを使用することもできません。

オプション引数*buttonMode*は、元のサイズを縮小した際、4D Chartエリアの表示をコントロールします (高さは150ピクセル以下、幅は300ピクセル以下) 。

- 0を*buttonMode*に渡すと、エリアはフォームエディタとユーザモードでボタンとして表示されます。ユーザがそのボタンをクリックすると、4D Chartがフルページモードへと切り替わります。これはデフォルト動作です。
- 1を*buttonMode*に渡すと、エリアはフォームエディタでボタンとして表示されます (元のサイズが縮小されている場合) 。ただし、ユーザモードでは表示されません。この場合、コンテンツの一部だけが表示される可能性があります。

例題

以下はareaを入力不可にするフォームメソッドです。

```
If(Form event=On Load)
   CT SET ENTERABLE(Area;0) `エリアを入力不可にする
End if
```

CT SET PROPERTIES

CT SET PROPERTIES (area ; printOrder ; changeAlert ; hotlinkType ; saveAlert)

引数	型		説明
area	倍長整数	➡	4D Chartエリア
printOrder	整数	➡	プリント順: 0 = 行順、1 = 列順、-1 = 変更なし
changeAlert	整数	➡	グラフタイプ変更メッセージ: 0 = メッセージなし、1 = メッセージあり、-1 = 変更なし
hotlinkType	整数	➡	廃止 (-1を渡してください)
saveAlert	整数	➡	ドキュメントを閉じたときの警告ボックス: 0 = メッセージなし、1 = メッセージあり、-1 = 変更なし

説明

*CT SET PROPERTIES*コマンドは、指定された4D Chartエリアに設定されているプロパティ情報を取得します。

*printOrder*はドキュメントのどのページが印刷されるのかの順序です。プリント順はドキュメントが印刷される順序だけに影響し、ページの方向には影響しません。

*changeAlert*はグラフのタイプを変更しようとしたときに警告ボックスが表示されるかどうかを指定します。ユーザは、キャンセルするか変更を続行するかのオプションを選択できます。

*hotlinkType*はサポートされていません (-1を渡します) 。

*saveAlert*は、変更を保存していない4D Chartドキュメントをクローズするときに警告ボックスをユーザに表示するかどうかを指定します。

- *saveAlert*が1の場合、4D Chartは変更を保存していない4D Chartドキュメントをユーザがクローズすると通常の警告ボックスを表示します。警告ボックスには、変更を保存する、変更を保存しない、クローズせずにドキュメントに戻るというオプションがあります。
- *saveAlert*が0の場合、4D Chartは変更を保存しないか、またはユーザに警告ボックスを表示しません。変更を保存するかどうかは開発者が指定します。例外は、ピクチャフィールドに保存されるフォーム上の4D Chartエリアです。これらのエリアの内容はピクチャフィールドに自動的に保存されます。

例題

以下の例は、他のプロパティは変更せずに、グラフタイプの変更に対して警告を表示しないように設定します。

```
CT SET PROPERTIES(Area;-1;1;-1;-1)
```

CTオブジェクト

- CT ALIGN
- CT Array to polygon
- CT Count
- CT Draw line
- CT Draw oval
- CT Draw rectangle
- CT Draw text
- CT GET BOUNDARY
- CT GET FILL ATTRIBUTES
- CT GET HIGHLIGHT
- CT Get ID
- CT GET LINE ATTRIBUTES
- CT Get object type
- CT Get refnum
- CT GET TEXT ATTRIBUTES
- CT INSERT EXPRESSION
- CT INSERT FIELD
- CT MOVE
- CT Place picture
- CT SELECT
- CT SET FILL ATTRIBUTES
- CT SET FILLS ATTRIBUTES
- CT SET HIGHLIGHT
- CT SET LINE ATTRIBUTES
- CT SET LINES ATTRIBUTES
- CT SET REFNUM
- CT SET TEXT ATTRIBUTES
- CT SIZE

CT ALIGN

CT ALIGN (area ; scope ; horizontal ; vertical)

引数	型		説明
area	倍長整数	➡	4D Chartエリア
scope	倍長整数	➡	1 = すべて、0 = 選択されたオブジェクト
horizontal	整数	➡	0 = なし、1 = 左、2 = 中、3 = 右
vertical	整数	➡	0 = なし、1 = 上、2 = 中、3 = 下

説明

*CT ALIGN*コマンドは、*area*中の*scope*で指定されたオブジェクトを整列させます。

- *scope*が-1の場合、*CT ALIGN*コマンドはドキュメント内の全オブジェクトを整列させます。
- *scope*が0の場合、*CT ALIGN*コマンドは選択されたオブジェクトを整列させます。

*scope*で記述されたオブジェクトは*horizontal*と*vertical*に従って配列されます。

- 以下の表は*horizontal*の値とその効果を表しています:

値	整列
0	なし
1	左揃え
2	中揃え
3	右揃え

- 以下の表は引数 *vertical*の値とその効果を表しています:

値	整列
0	なし
1	上揃え
2	中揃え
3	下揃え

例題

以下の例は、選択したオブジェクトを縦横両方の中央に整列させます。

```
CT ALIGN(Area;0;2;2)
```

CT Array to polygon

CT Array to polygon (area ; arrayH ; arrayV) -> 戻り値

引数	型		説明
area	倍長整数	➡	4D Chartエリア
arrayH	実数配列	➡	頂点の横方向の値の配列
arrayV	実数配列	➡	頂点の縦方向の値の配列
戻り値	倍長整数	⮌	新しいオブジェクトのオブジェクトID

説明

*CT Array to polygon*は*arrayH*と*arrayV*の配列をもとにした新しい多角形を*area*に作成し、その新しいオブジェクトのIDを返します。

*arrayH*と*arrayV*は多角形の各頂点の位置を記述しています。この2つの配列にはポイント単位で実数、倍長整数、整数のいずれかのデータタイプを指定できます。多角形が正常に作成されるようにそれぞれの配列には少なくとも3つの要素が必要です。配列の要素が同じ数ではない場合には、大きい方の配列の余分の要素は無視されます。閉じた多角形を作成するには、各配列の最後の値と最初の値を一致させる必要があります。

例題

以下の例は、2つの配列に値を入れ、そこから多角形を作成します。次にその多角形を移動し、サイズを変更します:

```
$Vertices:=Num(Request("Enter number of vertices:"))
If(OK=1) `配列を宣言する
   ARRAY REAL(aVerticeH;$Vertices)
   ARRAY REAL(aVerticeV;$Vertices)
   For($i;1;$Vertices) `配列に値を入れる
      aVerticeH{$i}:=Sin($i)
      aVerticeV{$i}:=Cos($i)
   End for
 `多角形を描く
   $Poly:=CT Array to polygon(Area;aVerticeH;aVerticeV)
 `多角形を座標(10,10) に移動する
   CT MOVE(Area;$Poly;10;10)
   CT SIZE(Area;$Poly;200;200) `多角形のサイズを200×200に変更する
End if
```

CT Count

CT Count (area ; scope) -> 戻り値

引数	型		説明
area	倍長整数	⇒	4D Chart エリア
scope	倍長整数	⇒	-1 = すべて、0 = 選択されたオブジェクト、>0 = グループID
戻り値	整数	⇐	エリア内のオブジェクトの数

説明

*CT Count*は、*area*中*scope*で指定されたオブジェクトの数を返します。

scope が-1の場合、*CT Count*はグループ内にはないドキュメント中のオブジェクトの数を返します。グループは単一のオブジェクトになります。
scope が0の場合、*CT Count*はグループ内にはない現在選択されているオブジェクトの数を返します。グループは単一のオブジェクトになります。
scope が0よりも大きな場合、それはグループ用のIDである必要があり、*CT Count*はそのグループ内部のオブジェクト数を返します。この構文ではグループを解除せずにグループ内のオブジェクトについての情報を取得できます。ネストしているグループは、再度*CT Count*をコールして調査できます。

例題

以下の例は、現在選択されているオブジェクトの数を表示する警告ボックスを表示します。

```
$Count :=CT Count(Area;0)
ALERT("You have selected "+String($Count)+" object(s).")
```

CT Draw line

CT Draw line (area ; left ; top ; right ; bottom ; arrowhead) -> 戻り値

引数	型		説明
area	倍長整数	➡	4D Chartエリア
left	実数	➡	水平方向の開始位置 (ポイント単位)
top	実数	➡	垂直方向の開始位置 (ポイント単位)
right	実数	➡	水平方向の終了位置 (ポイント単位)
bottom	実数	➡	垂直方向の終了位置 (ポイント単位)
arrowhead	整数	➡	矢印位置のコード (表参照)
戻り値	倍長整数	⬅	新しいオブジェクトのオブジェクトID

説明

*CT Draw line*は*area*に新しい線オブジェクトを作成し、その新しいオブジェクトのオブジェクトIDを返します。オブジェクトは*left*、*top*、*right*、*bottom*の座標に応じて配置されます。

以下は*arrowhead*のコードです:

コード	位置
-1	デフォルト値
0	なし
1	始点
2	終点
3	両端

例題

以下のメソッドは、チャートエリアに矢印付きの線を描画します。

```
$Line:=CT Draw line(Area;10;10;50;50;3)
```

CT Draw oval

CT Draw oval (area ; left ; top ; right ; bottom) -> 戻り値

引数	型		説明
area	倍長整数	➡	4D Chartエリア
left	実数	➡	水平方向の開始位置 (ポイント単位)
top	実数	➡	垂直方向の開始位置 (ポイント単位)
right	実数	➡	水平方向の終了位置 (ポイント単位)
bottom	実数	➡	垂直方向の終了位置 (ポイント単位)
戻り値	倍長整数	⬅	新しいオブジェクトのオブジェクトID

説明

*CT Draw oval*は*area*に新しい楕円オブジェクトを作成し、その新しいオブジェクトのオブジェクトIDを返します。オブジェクトは*left*、*top*、*right*、*bottom*の座標に応じて配置されます。

例題

以下のメソッドは、円をチャートエリアに描画します:

```
$Oval:=CT Draw oval(Area;5;5;100;100)
```

CT Draw rectangle

CT Draw rectangle (area ; left ; top ; right ; bottom ; round) -> 戻り値

引数	型		説明
area	倍長整数	➡	4D Chartエリア
left	実数	➡	水平方向の開始位置 (ポイント単位)
top	実数	➡	垂直方向の開始位置 (ポイント単位)
right	実数	➡	水平方向の終了位置 (ポイント単位)
bottom	実数	➡	垂直方向の終了位置 (ポイント単位)
round	実数	➡	角の丸みの度合い (ポイント単位)
戻り値	倍長整数	⬅	新しいオブジェクトのオブジェクトID

説明

*CT Draw rectangle*は*area*に新しい矩形オブジェクトを作成し、その新しいオブジェクトのオブジェクトIDを返します。オブジェクトは*left*、*top*、*right*、*bottom*の座標に応じて配置されます。

引数*round*は新しい矩形の角の丸みの度合いを調整します。*round*が0ならば角は丸くなりません。

例題

以下のメソッドは、丸みのある矩形をチャートエリアに描画します:

```
$Rect:=CT Draw rectangle(Area;5;5;200;200;5)
```

CT Draw text

CT Draw text (area ; left ; top ; right ; bottom ; text) -> 戻り値

引数	型		説明
area	倍長整数	➡	4D Chart エリア
left	実数	➡	水平方向の開始位置 (ポイント単位)
top	実数	➡	垂直方向の開始位置 (ポイント単位)
right	実数	➡	水平方向の終了位置 (ポイント単位)
bottom	実数	➡	垂直方向の終了位置 (ポイント単位)
text	テキスト	➡	新しいテキストオブジェクトのテキスト
戻り値	倍長整数	⬅	新しいオブジェクトのオブジェクトID

説明

*CT Draw text*は*area*に新しいテキストオブジェクトを作成し、その新しいオブジェクトのオブジェクトIDを返します。オブジェクトは*left*、*top*、*right*、*bottom*の座標に応じて配置されます。

例題

以下のメソッドはチャートエリアの上部左隅にテキスト“Hello World”を記述します。

```
$Text:=CT Draw text(Area;0;0;300;10;"Hello World")
```

CT GET BOUNDARY

CT GET BOUNDARY (area ; scope ; left ; top ; right ; bottom)

引数	型		説明
area	倍長整数	⇒	4D Chart エリア
scope	倍長整数	⇒	コマンドのスコープ: -1 = すべて、0 = 選択されたオブジェクト、>0 = オブジェクトID
left	実数	⇐	開始点の横方向の位置を受け取る (ポイント単位)
top	実数	⇐	開始点の縦方向の位置を受け取る (ポイント単位)
right	実数	⇐	終了点の横方向の位置を受け取る (ポイント単位)
bottom	実数	⇐	終了点の縦方向の位置を受け取る (ポイント単位)

説明

*CT GET BOUNDARY*コマンドは*area*内*scope*で記述されたオブジェクトの境界を*left*、*top*、*right*、*bottom*変数に返します。

境界とはオブジェクトを含む最も小さな矩形領域の座標です。

- *scope*が-1の場合、*CT GET BOUNDARY*コマンドはドキュメント内のすべてのオブジェクトに対する境界を返します。
- *scope*が0の場合、*CT GET BOUNDARY*コマンドは選択されたオブジェクトの境界を返します。
- *scope*が0よりも大きな場合、それは特定オブジェクトのIDと同じである必要があり、そのオブジェクトの境界が返されます。オブジェクトが存在しない場合には、*CT GET BOUNDARY*コマンドは各座標に対して-32000を返します。

例題

以下のコードは選択したオブジェクトの境界を$left、$top、$right、$bottom変数に返します。

```
CT GET BOUNDARY(Area;0;$left;$top;$right;$bottom)
```

CT GET FILL ATTRIBUTES

CT GET FILL ATTRIBUTES (area ; scope ; pattern ; color)

引数	型		説明
area	倍長整数	⇒	4D Chart エリア
scope	倍長整数	⇒	コマンドのスコープ: -2 = デフォルト、-1 = すべて、0 = 選択されたオブジェクト、>0 = オブジェクトID
pattern	整数	⇐	パターンを受け取る (0から36まで)
color	倍長整数	⇐	カラーの値を受け取る

説明

*CT GET FILL ATTRIBUTES*コマンドは、*area*内*scope*で記述されるオブジェクトに対する塗りつぶし属性を変数に返します。塗りつぶし属性はオブジェクトの内部によって決定されます。

*pattern*はパレット上のパターン番号です。以下は*pattern*に対応するコードです。

*color*はオブジェクトの色を受け取る倍長整数です。*CT Index to color*や*CT RGB to color*を使用することで *color*の値を指定できます。

Note: このコマンドは、このテーマにある描画ツールや描画関数を使用してドキュメントに追加されたオブジェクトの属性を取得するために使用してください。系列カラムなどのチャートオブジェクト属性を取得するには、チャートテーマで説明するコマンドを使用してください。

例題

以下の例は、選択されたオブジェクトの塗りつぶし属性を$Pattern変数と$Color変数に返します:

```
CT GET FILL ATTRIBUTES(Area;0;$Pattern;$Color)
```

CT GET HIGHLIGHT

CT GET HIGHLIGHT (area ; first ; last)

引数	型		説明
area	倍長整数	⇒	4D Chart エリア
first	整数	⇐	先頭の文字の位置から1を引いたものを受け取る
last	整数	⇐	最終の文字の位置を受け取る

説明

*CT GET HIGHLIGHT*コマンドは、*area*内で反転表示されているテキストの文字位置を*first*と*last*に返します。

*first*は反転表示された先頭の文字の位置から1を引いたものであり、*last*は反転表示された最終の文字です。*first*と *last*が同じ場合には、反転表示されている文字はないので、挿入点は*first*と*first*+1の間になります。

反転表示されるテキストを持つのは一度に1つのオブジェクトだけなので、scopeは必要ありません。*area*に反転表示されているテキストが存在しない場合、*CT GET HIGHLIGHT*コマンドは*first*と*last*に-32000を返します。

Example

以下の例は、反転表示されたテキストの位置を返し、テキストが選択されていない場合にはユーザに警告を表示します。

```
CT GET HIGHLIGHT(Area;$First;$Last)
If(CT Error=46)
   ALERT("There is no text highlighted.")
End if
```

CT Get ID

CT Get ID (area ; scope ; index) -> 戻り値

引数	型		説明
area	倍長整数	➡	4D Chart エリア
scope	倍長整数	➡	関数のスコープ: -1 = すべて、0 = 選択されたオブジェクト、>0 = グループID
index	倍長整数	➡	スコープでのオブジェクトの数値
戻り値	倍長整数	⬅	オブジェクトのユニークのオブジェクトID

説明

*CT Get ID*は、*area*内*scope*と*index*で記述されたオブジェクトに対する一意なIDを返します。この数値は他の多くの4D Chartコマンドで使用されるオブジェクトIDです。

オブジェクトIDを取得するにはまず最初に対象のオブジェクトセットを指定し、次にそのセット内のオブジェクトの順序を指定します。オブジェクトは後ろから前への順になっています。一番後ろのオブジェクトにはインデックス1がついています。

- *scope*が-1の場合、*index*はドキュメント全体のオブジェクトの順序を指します。
- *scope* が0の場合、*index*は現在選択されているオブジェクト内のオブジェクトの順序を指します。
- *scope*が0よりも大きな場合には、それがグループに対するIDである必要があり、*index* はそのグループ内部のオブジェクトの順序を指します。この最後の構文ではグループを解除せずにグループ内のオブジェクトを扱うことができます。

例題

以下の例は、選択したオブジェクトのIDを取り出す方法を示しています。

```
vID:=CT Get ID(Area;0;1) `最初に選択したオブジェクトのIDを取得する
```

CT GET LINE ATTRIBUTES

CT GET LINE ATTRIBUTES (area ; scope ; pattern ; color ; lineWidth)

引数	型		説明
area	倍長整数	⇒	4D Chart area
scope	倍長整数	⇒	Scope of the command -2 = Default -1 = All 0 = Selected objects >0 = Object ID
pattern	整数	⇐	Receives pattern index
color	倍長整数	⇐	Receives color value
lineWidth	実数	⇐	Receives line width (in points)

説明

CT GET LINE ATTRIBUTES コマンドは、*area* 内 *scope* で記述されたオブジェクトに対する線属性を変数に返します。線以外のオブジェクトには、線属性がそのオブジェクトの境界線に適用されます。

引数 *pattern* は、パレット上のパターン番号です。以下は引数 *pattern* に対応するコードです。

引数 *color* は、オブジェクトの色を指定する倍長整数です。*CT Index to color* 関数や*CT RGB to color* 関数を使用することで *color* の値を指定できます。

引数 *lineWidth* は、ポイント単位で計測される線の太さです。

Note: このコマンドは線ツールや *CT Draw line* 関数を使用してドキュメントに追加された線の属性を取得するために使用してください。グリッド線などのチャート線属性を取得するには、チャートテーマで説明するコマンドを使用してください。

例題

以下の例は、選択したオブジェクトの線属性を$Pattern、$Color、$Width変数に返します。

```
CT GET LINE ATTRIBUTES(Area;0;$Pattern;$Color;$Width)
```

CT Get object type

CT Get object type (area ; scope) -> 戻り値

引数	型		説明
area	倍長整数	➡	4D Chart area
scope	倍長整数	➡	Scope of the function -1 = All 0 = Selected objects >0 = Object ID
戻り値	整数	↩	Object type of the objects in area

説明

CT Get object type 関数は、*area*内*scope*で記述されたオブジェクトのオブジェクトタイプを返します。

オブジェクトのタイプは整数コードによって記述され、一度オブジェクトが作成されると変更はできません。

- *scope* が-1の場合には、*CT Get object type* 関数はドキュメント内のすべてのオブジェクトのオブジェクトタイプを返します。オブジェクトのタイプがすべて同じではないときには、*CT Get object type* 関数は-32000を返します。
- *scope* が0の場合には、*CT Get object type* 関数は選択されたオブジェクトのオブジェクトタイプを返します。オブジェクトのタイプがすべて同じではないときには、*CT Get object type* 関数は-32000を返します。
- *scope* が0よりも大きな場合には、それが特定オブジェクトのIDと同じである必要があり、そのオブジェクトのタイプが返されます。オブジェクトが存在しない場合には、*CT Get object type* 関数は-32000を返します。

以下の表は、すべてのオブジェクトコードです。

コード	オブジェクトタイプ
1	テキスト
2	使用せず
3	ピクチャ
4	使用せず
5	チャート
6	矩形
7	多角形
8	長方形
9	使用せず
10	線
11	グループ

例題

以下のコマンドは、選択されたオブジェクトのIDを$ID変数に返します。

```
$ID:=CT Get object type(Area;0)
```

CT Get refnum

CT Get refnum (area ; scope) -> 戻り値

引数	型		説明
area	倍長整数	⇒	4D Chart エリア
scope	倍長整数	⇒	関数のスコープ: -2 = デフォルト、-1 = すべて、0 = 選択されたオブジェクト、>0 = オブジェクトID
戻り値	倍長整数	⇐	エリア内のオブジェクトの参照番号

説明

CT Get refnum 関数は、*area*内*scope*で記述されたオブジェクトの参照番号を返します。参照番号はオブジェクトに対応する倍長整数であり、一意である必要性はありません。参照番号はプロシージャだけに従って操作されます。参照番号は、ユーザがオブジェクトに割り当てるものです。これに対して、オブジェクトIDは4D Chartによって割り当てられるものです。

- *scope* が-2の場合には、*CT Get refnum*はデフォルトの参照番号を返します。
- *scope* が-1の場合には、*CT Get refnum*はドキュメント内の全オブジェクトに対する参照番号を返します。オブジェクトの参照番号がすべて同じではないときには、*CT Get refnum*は、-32000を返します。
- *scope* が0の場合には、*CT Get refnum*は選択されたオブジェクトの参照番号を返します。オブジェクトの参照番号がすべて同じではないときには、*CT Get refnum*は、-32000を返します。
- *scope* が0よりも大きな場合には、それが特定オブジェクトのIDと同じである必要があり、そのオブジェクトの参照番号が返されます。オブジェクトが存在しない場合には、*CT Get refnum*は-32000を返します。

例題

以下の例は、エリアを含むフォーム上のボタン用オブジェクトメソッドです。オブジェクトメソッドは、実行されると1つのオブジェクトだけが選択されているかどうかをチェックし、対応するレコードを[Parts]テーブルで検索し、その説明を表示します。

```
QUERY([Parts];[Parts]RefNum=CT Get refnum(Area;0))
ALERT("This object is a "+[Parts]Description)
```

CT GET TEXT ATTRIBUTES

CT GET TEXT ATTRIBUTES (area ; scope ; fontID ; fontSize ; style ; color ; justification)

引数	型		説明
area	倍長整数	⇒	4D Chart エリア
scope	倍長整数	⇒	コマンドのスコープ -3 = 選択されたテキスト -2 = デフォルト -1 = すべて 0 = 選択されたオブジェクト >0 = オブジェクトID
fontID	整数	⇐	フォントIDを受け取る
fontSize	整数	⇐	フォントサイズを受け取る (ポイント単位)
style	整数	⇐	フォントスタイルを受け取る
color	倍長整数	⇐	テキストカラーを受け取る
justification	整数	⇐	テキストの位置揃えを受け取る 0 = 左 1 = 中央 2 = 右

説明

CT GET TEXT ATTRIBUTES コマンドは、*area*内*scope*で記述されたテキストのテキストの属性を引数 *fontID* 、*size* 、*color* 、 *justification* に返します。

引数 *fontID* は、システム内にあるフォントのIDです。フォントのID番号は *CT Font number* 関数を使用することによって取得できます。

引数 *fontSize* は、反転表示されたテキストやテキストオブジェクトのポイント単位のサイズです。

引数 *style* は、複数のスタイル番号の加算の結果を混合した番号です。以下の表はスタイル番号を示しています。

値	スタイル
0	標準
1	太字 (ボルード)
2	斜体 (イタリック)
4	下線 (アンダーライン)
8	アウトライン
16	シャドウ

引数 *color* は、オブジェクトの色を指定する倍長整数です。*CT Index to color* 関数や *CT RGB to color* 関数を使用することで *color* の値を指定できます。

引数 *justification* は、テキストの位置揃えです。

Note: このコマンドはテキストツールや *CT Draw text* 関数を使用してドキュメントに追加されたテキストの属性を取得するために使用してください。軸ラベルなどのチャートテキストの属性を取得するには、チャートテーマで説明するコマンドを使用してください。

例題

以下の例は、選択したテキストオブジェクトの属性を$Font、$Size、$Style、$Color、$Justify変数に返します。

```
CT GET TEXT ATTRIBUTES(Area;0;$Font;$Size;$Style;$Color;$Justify)
```

CT INSERT EXPRESSION

CT INSERT EXPRESSION (area ; scope ; first ; last ; expression ; format)

引数	型		説明
area	倍長整数	➡	4D Chart エリア
scope	倍長整数	➡	-1 = 先頭のオブジェクト 0 = セレクション内の先頭のオブジェクト >0 = オブジェクトID
first	整数	➡	先頭の文字の位置から1を引いたもの
last	整数	➡	最終の文字の位置
expression	文字	➡	式
format	文字	➡	式のフォーマット

説明

CT INSERT EXPRESSION コマンドは、*area*内*scope*で指定されたテキストオブジェクトに*expression* への参照を挿入します。

- *scope*が-1の場合には、*CT INSERT EXPRESSION* コマンドはドキュメントの最初のオブジェクトに参照を挿入します。
- *scope*が0の場合には、*CT INSERT EXPRESSION* コマンドは最初に選択されたオブジェクトに参照を挿入します。
- *scope*が0よりも大きな場合には、それが特定テキストオブジェクトのIDと同じである必要があり、参照はそのテキストオブジェクト内部に挿入されます。オブジェクトが存在しない場合には、 *CT INSERT EXPRESSION* コマンドは何も行いません。

*scope*で記述されたオブジェクトがテキストオブジェクトではない場合には、*CT INSERT EXPRESSION* コマンドは何も行いません。

引数 *first* と *last* は、参照が挿入される場所を決定します。*first* は置き換えられる先頭の文字の位置から1少ないものであり、*last* は置き換えられる最終の文字位置です。*first* が *last* と同じ場合には、文字は置き換えられず、参照は *first* と *first* +1の間に挿入されます。*last* がテキストオブジェクトにある文字の数よりも大きい場合には、*CT INSERT EXPRESSION* コマンドはテキストオブジェクトの first から最終の文字まで文字を置き換えます。

引数 *expression* は値を返す正しい4D式であるテキストです。*expression* は、フィールド、変数、4D関数、ユーザ定義関数（プロジェクトメソッド）、プラグイン、ステートメントのいずれかへの参照になります。

以下の表は、それぞれの式タイプの例を示しています。

例	型
[Drawings]Object	フィールド
vCriteria	変数
Current Date	4D 関数
GetNum	ユーザ定義関数 (プロジェクトメソッド)
CT Count	4D Chart 関数
3 * "Hello"	ステートメント

オプション引数の *format* は、参照用の表示フォーマットです。このオプションは、フォーマットダイアログボックスからフォーマットを選択するのと同じです。フォーマットはその番号またはその名前から参照されます。フォーマットは、フォーマットダイアログボックスのリストに表示される順に番号付けされています。

format が1桁か2桁の文字列の場合には、*field* に適用されたフォーマットはリストからのものです。*format* が1桁か2桁の文字列ではない場合には、リスト内の各フォーマットのテキスト値と比較されます。リスト内の値のどれかと一致した場合には、そのフォーマットが適用されます。つまり、最初の日付フォーマットは "19"または "Short" のどちらかで参照できるということです。

format がフォーマットのリストにない場合は、カスタム数値フォーマットとして解釈されます。format が参照の結果の値としては不適切である場合には、無効になります。例えば、数値に対して日付フォーマットを使用した場合には、その数値はフォーマットされずに表示されます。

例題

以下の例は、新しいテキストオブジェクトを作成し、それに4D関数の**Current date**への参照を入れ、Long日付フォーマットを使用してフォーマットします。

```
$ID :=CT Draw text(Area;0.5;0.5;3.5;1;"Today's date is: ")
CT INSERT EXPRESSION(Area;$ID;32000;32000;"Current date";"Long")
```

CT INSERT FIELD

CT INSERT FIELD (area ; scope ; first ; last ; numTable ; numField ; format)

引数	型		説明
area	倍長整数	➡	4D Chart エリア
scope	倍長整数	➡	-1 = ドキュメント内の最初のオブジェクト 0 = セレクション内の最初のオブジェクト >0 = オブジェクトID
first	整数	➡	先頭の文字の位置から1を引いたもの
last	整数	➡	最終の文字の位置
numTable	整数	➡	参照のテーブル番号
numField	整数	➡	参照のフィールド番号
format	文字	➡	参照のフォーマット

説明

CT INSERT FIELD コマンドは、*area*内*scope*で指定されたテキストオブジェクトにフィールド参照を挿入します。

- *scope* が-1の場合には、*CT INSERT FIELD*はドキュメントの最初のオブジェクトに参照を挿入します。
- *scope* が0の場合には、*CT INSERT FIELD*は最初に選択されたオブジェクトに参照を挿入します。
- *scope* が0よりも大きな場合には、それが特定テキストオブジェクトのIDと同じである必要があり、参照はそのテキストオブジェクト内部に挿入されます。オブジェクトが存在しない場合には、*CT INSERT FIELD*は何も行いません。

scope で記述されたオブジェクトがテキストオブジェクトではない場合には、*CT INSERT FIELD*は何も行いません。

引数 *first* と *last* は、参照が挿入される場所を決定します。*first* は置き換えられる先頭の文字の位置から1少ないものであり、*last* は置き換えられる最終の文字位置です。*first* が *last* と同じ場合には、文字は置き換えられず、参照は *first* と *first* +1の間に挿入されます。*last* がテキストオブジェクトにある文字の数よりも大きい場合には、*CT INSERT FIELD*はテキストオブジェクトの *first* から最終の文字まで文字を置き換えます。

table と *field* は参照されるフィールドを決定します。*table* はテーブルの番号であり、*field* はフィールドの番号です。テーブルとフィールドは作成された順に番号付けされています。

オプション引数の *format* は、参照用の表示フォーマットです。このオプションは、フォーマットダイアログボックスからフォーマットを選択するのと同じです。フォーマットはその番号またはその名前から参照されます。フォーマットは、フォーマットダイアログボックスのリストに表示される順に番号付けされています。

format が1桁か2桁の文字列の場合には、*field* に適用されたフォーマットはリストからのものです。*format* が1桁か2桁の文字列ではない場合には、リスト内の各フォーマットのテキスト値と比較されます。リスト内の値のどれかと一致した場合には、そのフォーマットが適用されます。つまり、最初の日付フォーマットは "19" または "Short" のどちらかで参照できるということです。

format がフォーマットのリストにない場合は、カスタム数値フォーマットとして解釈されます。*format* が参照の結果の値としては不適切である場合には、無効になります。例えば、数値に対して日付フォーマットを使用した場合には、その数値はフォーマットされずに表示されます。

例題 1

以下の例は、IDが1であるテキストオブジェクトに最初のテーブルの最初のフィールドへの参照を挿入し、オブジェクトにある任意のテキストを置き換えてから、リストの11番目のフォーマットに従ってフォーマットを行います。

```
CT INSERT FIELD(Area;1;0;32000;1;1;"11")
```

例題 2

4Dの**Field**関数と**Table**関数を使用すると、フィールドやテーブルの番号を判断できます。これによって、コードは読みやすくなります。例えば、前述の例で使用されたフィールドが[Customer]Nameである場合、コードは以下のようになります:

```
CT INSERT FIELD(Area;1;0;32000;Table(->[Customers]);Field(->Name);"11")
```

CT MOVE

CT MOVE (area ; scope ; newLeft ; newTop)

引数	型		説明
area	倍長整数	⇒	4D Chart エリア
scope	倍長整数	⇒	コマンドのスコープ -1 = すべて 0 = 選択されたオブジェクト >0 = オブジェクトID
newLeft	実数	⇒	新しい左端の座標
newTop	実数	⇒	新しい上端の座標

説明

CT MOVE コマンドは、*area*内*scope*で指定されたオブジェクトの位置を変更します。

- *scope* が-1の場合には、*CT MOVE*はドキュメント内の全オブジェクトの位置を変更します。
- *scope* が0の場合には、*CT MOVE*は選択されたオブジェクトの位置を変更します。
- *scope* が0よりも大きな場合には、それが特定オブジェクトのIDと同じである必要があり、そのオブジェクトの位置を変更します。

scope で記述されたオブジェクトは、現在の原点からのオフセットとして指定される引数 *newLeft* と *newTop* に従って移動されます。

例題

以下の例は、選択されたオブジェクトをチャートエリアの左上隅に移動します。

```
CT MOVE(Area;0;0;0)
```

CT Place picture

CT Place picture (area ; picture ; left ; top) -> 戻り値

引数	型		説明
area	倍長整数	➡	4D Chart エリア
picture	ピクチャー	➡	配置する4D ピクチャ
left	実数	➡	左端からポイント単位で表した左座標
top	実数	➡	上端からポイント単位で表した上座標
戻り値	倍長整数	⮌	ピクチャのオブジェクトID

説明

CT Place picture は *left* と *top* によって決定される点で *picture* を *area* に貼り付け、そのピクチャのオブジェクトIDを返します。

ピクチャは、正しい4Dのピクチャ式である必要があります。

例題

以下の例は、指定された企業のエリアにピクチャフィールドの [Logos]Logo の内容を貼り付けます。

```
MyRequest:=Request("Which company's logo do you want?")
If(OK=1)
   QUERY([Logos];[Logos]Company=MyRequest)
   If(Records in selection([Logos]>0)
      $NewPict:=CT Place picture(Area;[Logos]Logo;10;10)
   Else
      ALERT("This company does not exist.")
   End if
End if
```

CT SELECT

CT SELECT (area ; scope ; action)

引数	型		説明
area	倍長整数	➡	4D Chart エリア
scope	倍長整数	➡	コマンドのスコープ -1 = すべて 0 = 選択されたオブジェクト >0 = オブジェクトID
action	整数	➡	オブジェクトの選択または選択解除 0 = 選択解除 1 = 選択 2 = 切り替え

説明

CT SELECT コマンドは、*area*内*scope*で指定されたオブジェクトの選択または選択解除を行います。

- *scope* が-1の場合には、*CT SELECT*はドキュメント内の全オブジェクトを対象にします。
- *scope* が0の場合には、*CT SELECT*は選択されたオブジェクトを対象にします。
- *scope* が0よりも大きな場合には、それが特定オブジェクトのIDと同じである必要があり、そのオブジェクトが対象になります。オブジェクトが存在しない場合には、*CT SELECT*は何も行いません。

scope で記述されたオブジェクトは、引数 *action* に従って選択または選択解除されます。*action* が0の場合には、*scope* で記述されたオブジェクトの選択は解除されます。*action* が1の場合には、*scope* で記述されたオブジェクトが選択されます。*action* が2の場合には、オブジェクトの現在の状態が切り替わります。つまり、選択されたものは選択が解除され、選択解除されているものは選択されます。

scope 以外のオブジェクトは、 *CT SELECT*の対象にはなりません。つまり、エリアで既に選択されており、*scope* で指定されないオブジェクトは、選択されたままになります。

例題

以下の例は、ドキュメントの全オブジェクトの選択を解除してから、ID番号が1であるオブジェクトを選択します。

```
CT SELECT(Area;-1;0)
CT SELECT(Area;1;1)
```

CT SET FILL ATTRIBUTES

CT SET FILL ATTRIBUTES (area ; scope ; pattern ; color)

引数	型		説明
area	倍長整数	➡	4D Chart エリア
scope	倍長整数	➡	コマンドのスコープ -2 = デフォルト -1 = すべて 0 = 選択されたオブジェクト >0 = オブジェクトID
pattern	整数	➡	パターン (0から36まで) -1 = 変更なし
color	倍長整数	➡	カラーの値 -1 = 変更なし

説明

CT SET FILL ATTRIBUTES コマンドは、*area*内*scope*で指定されたオブジェクトに対する塗りつぶし属性を変更します。塗りつぶし属性はオブジェクトの内部によって決定されます。

引数 *pattern* は、パレット上のパターン番号です。以下は引数 *pattern* に対応するコードです。

引数 *color* は、オブジェクトの色を指定する倍長整数です。*CT Index to color* 関数や*CT RGB to color* 関数を使用することで *color* の値を指定できます。

Note: このコマンドは、このテーマにある描画ツールや描画関数を使用してドキュメントに追加されたオブジェクトの属性を取得するために使用してください。系列カラムなどのチャートオブジェクト属性を取得するには、チャートテーマで説明するコマンドを使用してください。

例題

以下の例は、選択されたオブジェクトの塗りつぶし属性を黄色の塗りつぶしに設定します。

```
CT SET FILL ATTRIBUTES(Area;0;3;CT Index to color(2))
```

CT SET FILLS ATTRIBUTES

CT SET FILLS ATTRIBUTES (area ; objects ; patterns ; colors)

引数	型		説明
area	倍長整数	➡	4D Chart エリア
objects	倍長整数配列	➡	オブジェクトID番号のリスト
patterns	整数配列	➡	パターン番号のリスト
colors	倍長整数配列	➡	カラーの値のリスト

説明

CT SET FILLS ATTRIBUTES コマンドは、オブジェクトのリストに適用されることを除き、*CT SET FILL ATTRIBUTES* コマンドと同じように動作します。引数 *objects* には、塗りつぶし属性を設定したいオブジェクトのID番号のリストを格納する倍長整数の配列を渡します。

引数 *patterns* と *colors* は、対応している属性を格納している配列です。

詳細は、*CT SET FILL ATTRIBUTES* コマンドを参照してください。

例題

この例題では、vct という名前の4D Chart エリアを含むフォームがあると想定します。特定な線とパターンを使用して、100個の矩形を同時に作成します。*CT SET LINE ATTRIBUTES* コマンドと *CT SET FILL ATTRIBUTES* コマンドを100回呼び出す代わりに、1度の呼び出しで、配列を埋め、矩形の属性を定義します。

以下は、フォーム用のメソッドです。

```
If(Form event=On Load)
   ARRAY LONGINT($ids;100)
   ARRAY INTEGER($pat;100)
   ARRAY INTEGER($pat2;100)
   ARRAY LONGINT($color;100)
   ARRAY LONGINT($color2;100)
   ARRAY LONGINT($ln;100) `または ARRAY REAL ($ln;100)
   CT SELECT(vct;-1;1)
   CT DO COMMAND(vct;2006)
   For($i;1;100)
      $ids{$i}:=CT Draw rectangle(vCT;40+($i*10);40;40+(($i+1)*10)-2;60;0)
      $pat{$i}:=1+($i%30)
      $pat2{$i}:=1+($i%15)
      $color{$i}:=CT Index to color($i)
      $color2{$i}:=CT Index to color(100-$i)
      $ln{$i}:=1+$i%4
   End for
   CT SET FILL ATTRIBUTES(vct;$ids;$pat;$color)
   CT SET LINE ATTRIBUTES(vct;$ids;$pat2;$color2;$ln)
End if
```

CT SET HIGHLIGHT

CT SET HIGHLIGHT (area ; scope ; first ; last)

引数	型		説明
area	倍長整数	➡	4D Chart エリア
scope	倍長整数	➡	-1 = ドキュメント内の最初のオブジェクト 0 = セレクション内の最初のオブジェクト >0 = オブジェクトID
first	整数	➡	先頭の文字の位置から1を引いたもの
last	整数	➡	最終の文字の位置

説明

CT SET HIGHLIGHT コマンドは、*area*内*scope*で指定されたテキストオブジェクト内の文字を反転表示します。
scope が-1の場合には、*CT SET HIGHLIGHT*はドキュメントの最初のオブジェクトにある文字を反転表示します。
scope が0の場合には、*CT SET HIGHLIGHT*は最初に選択されたオブジェクトの文字を反転表示します。
scope が0よりも大きな場合は、それが特定テキストオブジェクトのIDと同じである必要があり、そのテキストオブジェクト内部の文字が反転表示されます。オブジェクトが存在しない場合は、*CT SET HIGHLIGHT*は何も行いません。*CT SET HIGHLIGHT*では、*scope* で記述されたオブジェクトが *area* で選択される唯一のオブジェクトです。

scope で記述されたオブジェクトがテキストオブジェクトではない場合には、*CT SET HIGHLIGHT*は何も行いません。

引数 *first* と *last* は、どの文字を反転表示するのかを決定します。*first* は反転表示される先頭の文字位置から1を引いたものです。*last* は反転表示される最終の文字の位置です。*first* と *last* が同じ場合には、選択されている文字はないので、挿入点は *first* と *first* +1の間になります。*last* がテキストオブジェクトの文字数よりも大きい場合には、*CT SET HIGHLIGHT*はテキストオブジェクトの最終まで文字を反転表示します。

*CT SET HIGHLIGHT*は参照の一部だけの反転表示は行いません。参照の任意の部分が反転表示されると、*CT SET HIGHLIGHT*は参照全体を含めるように反転表示を調整します。

例題

以下の例は、選択されたテキストオブジェクトのテキストを取得し、"4D" という名前を検索します。"4D" が見つかった場合には、反転表示されて太字 (ボールド) になります。

```
$Find :=Position("4D";$Text)
If($Find #0)
   CT SET HIGHLIGHT(Area;0;$Find -1;$Find +12)
   CT SET TEXT ATTRIBUTES(Area;-3;-1;-1;1;-1;-1)
End if
```

CT SET LINE ATTRIBUTES

CT SET LINE ATTRIBUTES (area ; scope ; pattern ; color ; lineWidth)

引数	型		説明
area	倍長整数	⇒	4D Chart エリア
scope	倍長整数	⇒	コマンドのスコープ -2 = デフォルト -1 = すべて 0 = 選択されたオブジェクト >0 = オブジェクトID
pattern	整数	⇒	パターン (0から36まで) -1 = 変更なし
color	倍長整数	⇒	カラーの値 -1 = 変更なし
lineWidth	実数	⇒	線の幅 (ポイント単位) -1 = 変更なし

説明

CT SET LINE ATTRIBUTES コマンドは、*area*内*scope*で指定されたオブジェクトに対する線属性を変更します。線以外のオブジェクトには、線属性がそのオブジェクトの境界線に適用されます。

引数 *pattern* は、パレット上のパターン番号です。以下は引数 *pattern* に対応するコードです。

引数 *color* は、オブジェクトの色を指定する倍長整数です。*CT Index to color* 関数や*CT RGB to color* 関数を使用することで *color* の値を指定できます。

引数 *lineWidth* は、ポイント単位で計測される線の太さです。

Note: このコマンドは線ツールや *CT Draw line* 関数を使用してドキュメントに追加された線の属性を取得するために使用してください。グリッド線などのチャート線属性を取得するには、チャートテーマで説明するコマンドを使用してください。

例題

以下の例は、選択されたオブジェクトの線属性を実線、青、3ポイントに設定します。

```
CT SET LINE ATTRIBUTES(Area;0;3;CT Index to color(6);3)
```

CT SET LINES ATTRIBUTES

CT SET LINES ATTRIBUTES (area ; objects ; patterns ; colors ; lineWidths)

引数	型		説明
area	倍長整数	➡	4D Chart エリア
objects	倍長整数配列	➡	オブジェクトID番号のリスト
patterns	整数配列	➡	パターン番号のリスト
colors	倍長整数配列	➡	カラーの値のリスト
lineWidths	実数配列	➡	線の太さのリスト (ポイント単位)

説明

CT SET LINES ATTRIBUTES コマンドの動作は、オブジェクトのリストに適用されていることを除き、*CT SET LINE ATTRIBUTES* コマンドと似ています。引数 *objects* には、属性を設定したいオブジェクトのID番号のリストを格納する倍長整数の配列を渡します。

CT SET LINES ATTRIBUTES コマンドは、*area* と *objects* によって指定された線の属性を設定します。引数 *patterns*、*colors*、*lineWidths* は、それに対応する属性を格納している配列です。

詳細は、*CT SET LINE ATTRIBUTES* コマンドを参照してください。

例題

CT SET FILL ATTRIBUTES コマンドの例題を参照してください。

CT SET REFNUM

CT SET REFNUM (area ; scope ; refNum)

引数	型		説明
area	倍長整数	➡	4D Chart エリア
scope	倍長整数	➡	コマンドのスコープ -2 = デフォルト -1 = すべて 0 = 選択されたオブジェクト >0 = オブジェクトID
refNum	倍長整数	➡	参照番号

説明

CT SET REFNUM コマンドは、引数 *refNum* を *area*内*scope*で指定されたオブジェクト用の参照番号にします。参照番号は、それがオブジェクトを識別する方法であり、一意ではないという点でオブジェクト名と似ています。参照番号はオブジェクトID番号ではありません。オブジェクトID番号はドキュメント内の各オブジェクトに対して4D Chartで割り当てられた一意の番号です。

- *scope* が-2の場合は、*CT SET REFNUM*がデフォルトの参照番号を設定します。これは、任意の新しいオブジェクトに使用される参照番号です。
- *scope* が-1の場合は、*CT SET REFNUM*はドキュメント内の全オブジェクトに対する参照番号を設定します。
- *scope* が0の場合は、*CT SET REFNUM*は選択されたオブジェクトに対する参照番号を設定します。
- *scope* が0よりも大きな場合は、それが特定オブジェクトのIDと同じである必要があります。オブジェクトが存在しない場合は、*CT SET REFNUM*は何も行いません。

参照番号はオブジェクトに割り当てられる一意ではない倍長整数値です。参照番号はメソッドからのみ操作できます。参照番号のデフォルト値は0です。

例題

以下の例は、選択したオブジェクトの参照番号をvNumber変数に含まれる値に変更します。

```
CT SET REFNUM(Area;0;vNumber)
```

CT SET TEXT ATTRIBUTES

CT SET TEXT ATTRIBUTES (area ; scope ; fontID ; fontSize ; style ; color ; justification)

引数	型		説明
area	倍長整数	⇒	4D Chart エリア
scope	倍長整数	⇒	コマンドのスコープ -3 = 選択されたテキスト -2 = デフォルト -1 = すべて 0 = 選択されたオブジェクト >0 = オブジェクトID
fontID	整数	⇒	フォントID
fontSize	整数	⇒	フォントサイズ (ポイント単位)
style	整数	⇒	フォントスタイル
color	倍長整数	⇒	テキストカラー
justification	整数	⇒	テキストの位置揃え 0 = 左 1 = 中央 2 = 右

説明

CT SET TEXT ATTRIBUTES コマンドは、*area*内*scope*で指定されたテキストにフォント、フォントサイズ、フォントスタイル、カラー、位置揃えを設定します。

引数 *fontID* は、システム内にあるフォントのIDです。フォントのID番号は *CT Font number* 関数を使用することによって取得できます。

引数 *fontSize* は、反転表示されたテキストやテキストオブジェクトのポイント単位のサイズです。

引数 *style* は、複数のスタイル番号の加算の結果を混合した番号です。以下の表はスタイル番号を示しています。

値	スタイル
0	標準
1	太字 (ボールド)
2	斜体 (イタリック)
4	下線 (アンダーライン)
8	アウトライン
16	シャドウ

引数 *color* は、オブジェクトの色を指定する倍長整数です。*CT Index to color* 関数や*CT RGB to color* 関数を使用することで *color* の値を指定できます。

引数 *justification* は、テキストの位置揃えです。

Note: このコマンドはテキストツールや *CT Draw text* 関数を使用してドキュメントに追加されたテキストの属性を取得するために使用してください。軸ラベルなどのチャートテキストの属性を取得するには、チャートテーマで説明するコマンドを使用してください。

例題

以下の例は、選択したテキストをTimes、14ポイント、太字、斜体、緑、中央揃えに設定します。

```
CT SET TEXT ATTRIBUTES(Area;0;CT Font number("Times");14;3;CT Index to color(10);1)
```

CT SIZE

CT SIZE (area ; scope ; width ; height)

引数	型		説明
area	倍長整数	➡	4D Chart エリア
scope	倍長整数	➡	コマンドのスコープ -1 = すべて 0 = 選択されたオブジェクト >0 = オブジェクトID
width	実数	➡	新しい幅 (ポイント単位) -1 = 変更なし
height	実数	➡	新しい高さ (ポイント単位) -1 = 変更なし

説明

CT SIZE コマンドは、*area*内*scope*で指定されたオブジェクトのサイズを変更します。オブジェクトのサイズを変更するときに、オブジェクトの左上隅は固定されています。

- *scope* が-1の場合には、*CT SIZE*はドキュメント内の全オブジェクトのサイズを変更します。
- *scope* が0の場合には、*CT SIZE*は選択されたオブジェクトのサイズを変更します。
- *scope* が0よりも大きな場合には、それは特定オブジェクトのIDである必要があり、そのオブジェクトのサイズが変更されます。

scope で記述されたオブジェクトは、ポイント単位で指定される引数 *width* と *height* に従ってサイズが変更されます。

例題

以下の例は、IDが5であるオブジェクトのサイズを変更します。

```
CT SIZE(Area;5;10;10)
```

CTチャート

- CT Chart arrays
- CT Chart data
- CT Chart selection
- CT EXPLODE PIE
- CT GET 3D VIEW
- CT GET AXIS ATTRIBUTES
- CT GET CHART COORDINATES
- CT GET CHART FILL ATTRIBUTES
- CT GET CHART LINE ATTRIBUTES
- CT GET CHART OPTIONS
- CT GET CHART PART
- CT Get chart picture
- CT GET CHART TEXT ATTRIBUTES
- CT Get chart type
- CT GET DATE SCALE
- CT GET DEPTH
- CT GET LABEL ATTRIBUTES
- CT GET LEGEND ATTRIBUTES
- CT Get legend text
- CT GET REAL SCALE
- CT GET TIPS ATTRIBUTES
- CT GET TITLE ATTRIBUTES
- CT GET VALUE ATTRIBUTES
- CT GET X DATE SCALE
- CT GET X REAL SCALE
- CT SET 3D VIEW
- CT SET AXIS ATTRIBUTES
- CT SET CHART COORDINATES
- CT SET CHART FILL ATTRIBUTES
- CT SET CHART LINE ATTRIBUTES
- CT SET CHART OPTIONS
- CT SET CHART PICTURE
- CT SET CHART TEXT ATTRIBUTES
- CT SET CHART TYPE
- CT SET DATE SCALE
- CT SET DEPTH
- CT SET LABEL ATTRIBUTES
- CT SET LEGEND ATTRIBUTES
- CT SET LEGEND TEXT
- CT SET REAL SCALE
- CT SET TIPS ATTRIBUTES
- CT SET TITLE ATTRIBUTES
- CT SET VALUE ATTRIBUTES
- CT SET X DATE SCALE
- CT SET X REAL SCALE
- CT SHOW GRID LINES
- CT UPDATE CHART

CT Chart arrays

CT Chart arrays (area ; type ; size ; categoryArray ; seriesArray ; valuesArray) -> 戻り値

引数	型		説明
area	倍長整数	⇒	4D Chart エリア
type	整数	⇒	グラフのタイプ (以下のコードを参照)
size	整数	⇒	グラフの初期サイズオプション 1 = 可変 2 = ウインドウと連動 (自動-変数) 3 = グラフと連動 (自動-ドキュメント)
categoryArray	配列	⇒	項目配列
seriesArray	配列	⇒	系列配列
valuesArray	配列	⇒	数値配列
戻り値	倍長整数	⇐	オブジェクトID番号

説明

CT Chart arrays 関数は、指定された配列に基づいてグラフを作成し、そのグラフのオブジェクトIDを返します。このコマンドは2次元グラフまたは3次元グラフのどちらを作成するときにも使用できます。

以下の表は、引数 *type* のコードを示しています。

コード	チャートタイプ
1	面
2	棒
3	ピクチャ
4	線
5	散布図
6	円
7	ポーラー
8	2D XY
100	3D 棒
101	3D 線
102	3D 面
103	3D 等高線
104	3D 三角形
105	3D ピン

引数 *size* は、グラフが生成されたときにそのグラフがどの程度のスペースを取るのか、ウインドウのサイズ変更をしたときにグラフのサイズはどのように変更するのかを決定します。

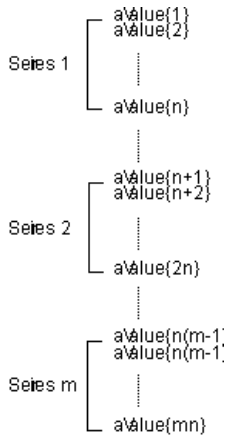
- 1を渡した場合 (サイズの変数) 、グラフは4D Chart エリアかまたはウインドウいっぱいになります。ハンドルを使って修正するまでこのサイズは保たれます。
- 2を渡した場合 (ウインドウと連動したサイズ) 、グラフは4D Chart エリアかまたはプラグインウインドウいっぱいになります。ウインドウサイズを変更した後は、グラフは自動的に変えられます。しかしながら、一度ハンドルを使ってグラフサイズを変更すると、この変更はそれ以上実行されません。
- 3を渡した場合 (文章と連動したサイズ) 、グラフはページセットアップダイアログボックスで選択したページの次元に合わせて変えられます。ハンドルを使ってグラフサイズを修正するまでこのサイズは保たれます。

categoryArray には、X軸項目があります。

seriesArray には、系列が含まれます。2次元グラフでは、系列は項目軸上に表示されます。3次元グラフでは、系列は数値軸上に表示されます。

valuesArray は、数値軸上でグラフ化されるすべての値を含む1次元の配列です。*categoryArray* と *seriesArray* の各要素の値が存在するように *valuesArray* には値が入っている必要があります。つまり、n 項目と m 系列がある場合には、*valuesArray* には n*m 個の要素があることになります。

以下の図は、*valuesArray* に入れられるものの順序を示しています。n は項目の合計数を表します。m は系列の合計数を表します。



例えば、以下のデータと数値配列の結果を参照してください。

学校名 (項目)	**年度 (系列)**	**生徒数(数値)**	**aValues**
Sunnyoaks	1990	1000	aValues{1}:=1000
Sunnyoaks	1992	1250	aValues{2}:=600
Sunnyoaks	1994	800	aValues{3}:=1250
Valley	1990	600	aValues{4}:=975
Valley	1992	975	aValues{5}:=800
Valley	1994	1100	aValues{6}:=1100

CT Chart data

CT Chart data (area ; type ; size ; groupCategory ; groupSeries ; numTable ; categoryField ; seriesField ; valuesField) -> 戻り値

引数	型		説明
area	倍長整数	⇒	4D Chart エリア
type	整数	⇒	グラフのタイプ (以下のコードを参照)
size	整数	⇒	グラフの初期サイズオプション 1 = 可変 2 = ウインドウと連動 (自動-ウインドウ) 3 = グラフと連動 (自動-ドキュメント)
groupCategory	整数	⇒	項目データをグループ化する? 0 = なし 1 = グループ化
groupSeries	整数	⇒	系列データをグループ化する? 0 = なし 1 = グループ化
numTable	整数	⇒	データをグラフ化するテーブルの番号
categoryField	整数	⇒	項目軸上にプロットするフィールドの番号
seriesField	整数	⇒	系列としてプロットするフィールドの番号
valuesField	整数	⇒	数値軸上にプロットするフィールドの番号
戻り値	倍長整数	⇐	チャートのオブジェクトID

説明

CT Chart data 関数は、引数 *table* のカレントセレクションのグラフを作成します。この関数はチャートのオブジェクトIDを返します。

以下の表は、引数 *type* のコードを示しています。

コード	チャートタイプ
1	面
2	棒
3	ピクチャ
4	線
5	散布図
6	円
7	ポーラー
8	2D XY
100	3D 棒
101	3D 線
102	3D 面
103	3D 等高線
104	3D 三角形
105	3D ピン

引数 *size* は、グラフが生成されたときにそのグラフがどの程度のスペースを取るのか、ウインドウのサイズ変更をしたときにグラフのサイズはどのように変更するのかを決定します。

- 1を渡した場合 (サイズの値) 、グラフは4D Chart エリアかまたはウインドウいっぱいになります。ハンドルを使って修正するまでこのサイズは保たれます。
- 2を渡した場合 (ウインドウと連動したサイズ) 、グラフは4D Chart エリアかまたはプラグインウインドウいっぱいになります。ウインドウサイズを変更した後は、グラフは自動的に変えられます。しかしながら、一度ハンドルを使ってグラフサイズを変更すると、この変更はそれ以上実行されません。
- 3を渡した場合 (文章と関連したサイズ) 、グラフはページセットアップダイアログボックスで選択したページの次元に合わせて変えられます。ハンドルを使ってグラフサイズを修正するまでこのサイズは保たれます。

引数 *GroupCategory* は、項目軸のデータがグループ化されるのかどうかを指定します。

- *groupCategory* が1の場合には、各項目は一意になり、重複した項目の値があればまとめられます。
- *groupCategory* が0の場合には、各項目の値は別々にグラフ化されます。

引数 *groupSeries* は、系列軸上のデータをグループ化するかどうかを指定します。

- *groupSeries* が1の場合には、各シリーズは一意になり、重複したシリーズの値があればまとめられます。
- *groupSeries* が0の場合には、各シリーズの値は別々にグラフ化されます。

引数 *table* は、データをグラフ化するテーブルの番号です。テーブルへのポインタを**Table**関数の引数として渡すことによって、テーブルの番号を検索することができます。

引数 *categoryField* は、グラフ化する項目軸上のフィールド番号です。

引数 *seriesField* は、グラフ化する系列軸上のフィールド番号です。2次元グラフでは、系列は項目軸上に表示されます。3次元グラフでは、系列は系列軸上に表示されます。

引数 *valuesField* は、グラフ化する数値軸上のフィールド番号です。

フィールドへのポインタを**Field**関数の引数として渡すと、フィールドの番号を検索することができます。

CT Chart selection

CT Chart selection (area ; type ; size ; groupCategory ; numTable ; categoryField ; series/valuesFields) -> 戻り値

引数	型		説明
area	倍長整数	➡	4D Chart エリア
type	整数	➡	グラフのタイプ (以下のコードを参照)
size	整数	➡	グラフの初期サイズオプション 1 = 可変 2 = ウインドウと連動 (自動-ウインドウ) 3 = ドキュメントと連動 (自動-ドキュメント)
groupCategory	整数	➡	整数 項目データをグループ化する 0 = なし 1 = グループ化
numTable	整数	➡	グラフのテーブル番号
categoryField	整数	➡	項目軸上にプロットするフィールドの番号
series/valuesFields	整数配列	➡	フィールド番号の配列
戻り値	倍長整数	↩	チャートのオブジェクトID

説明

CT Chart selection 関数は、引数 *table* 番号のカレントセレクションのグラフを作成します。この関数は、チャートのオブジェクトIDを返します。

以下の表は、引数 *type* のコードを示しています。

コード	チャートタイプ
1	面
2	棒
3	ピクチャ
4	線
5	散布図
6	円
7	ポーラー
8	2D XY
100	3D 棒
101	3D 線
102	3D 面
103	3D 等高線
104	3D 三角形
105	3D ピン

引数 *size* は、グラフが生成されたときにそのグラフがどの程度のスペースを取るのか、ウインドウのサイズ変更をしたときにグラフのサイズはどのように変更するのかを決定します。

- 1を渡した場合 (サイズの値) 、グラフは4D Chart エリアかまたはウインドウいっぱいになります。ハンドルを使って修正するまでこのサイズは保たれます。
- 2を渡した場合 (ウインドウと連動したサイズ) 、グラフは4D Chart エリアかまたはプラグインウインドウいっぱいになります。ウインドウサイズを変更した後は、グラフは自動的に変えられます。しかしながら、一度ハンドルを使ってグラフサイズを変更すると、この変更はそれ以上実行されません。
- 3を渡した場合 (文章と関連したサイズ) 、グラフはページセットアップダイアログボックスで選択したページの次元に合わせて変えられます。ハンドルを使ってグラフサイズを修正するまでこのサイズは保たれます。

引数 *GroupCategory* は、項目軸のデータがグループ化されるのかどうかを指定します。

- *groupCategory* が1の場合には、各項目は一意になり、重複した項目の値があればまとめられます。
- *groupCategory* が0の場合には、各項目の値は別々にグラフ化されます。

Note: 系列をグループ化する必要はありません。これは、系列はフィールド名なため一意だからです。

引数 *table* は、データをグラフ化するテーブルの番号です。テーブルへのポインタを**Table**関数の引数として渡すことによって、テーブルの番号を検索することができます。

categoryField は、グラフ化する項目軸上のフィールドの番号です。フィールドへのポインタを**Field**関数の引数として渡すことによってフィールドの番号を検索することができます。

series/*valuesFields* は系列と数値としてグラフ化するフィールドのフィールド番号配列です。フィールド名は系列になります。フィールドに格納された値は数値軸上でグラフ化されます。2次元グラフでは、系列は項目軸上に表示されます。3次元グラフでは、系列は数値軸上に表示されます。

CT EXPLODE PIE

CT EXPLODE PIE (area ; object ; category ; percentage)

引数	型		説明
area	倍長整数	⇒	4D Chart エリア
object	倍長整数	⇒	オブジェクトID
category	整数	⇒	切り離す円グラフのパイウェッジの項目番号 (0 = すべてのパイウェッジ)
percentage	整数	⇒	半径の長さのパーセンテージ (0から1000まで)

説明

CT EXPLODE PIE コマンドは、円グラフの中心から指定されたパイウェッジを移動します。

引数 *category* は、移動させるパイウェッジの項目番号です。*category* が0の場合には、すべてのパイウェッジが移動されます。

引数 *percentage* は、半径の長さのパーセンテージとして指定された、パイウェッジの移動距離です。このため、円グラフのサイズが変更されると、移動距離は新しい半径に従って変わります。

例題

以下の例は$ChartIDに指定された円グラフの最初の3つの部分を切り離します。各パイウェッジは円の半径の5%の距離で外側に移動されます。

```
For($i;1;3)
   CT EXPLODE PIE(Area;$ChartID;$i;5)
End for
```

CT GET 3D VIEW

CT GET 3D VIEW (area ; object ; rotation ; elevation)

引数	型		説明
area	倍長整数	⇒	4D Chart エリア
object	倍長整数	⇒	オブジェクトID
rotation	実数	⇐	度数単位で回転を受け取る (0から90まで)
elevation	実数	⇐	度数単位で仰角を受け取る (0から90まで)

説明

CT GET 3D VIEW コマンドは、引数 *area* と *object* で指定されたグラフの回転と仰角を返します。このコマンドは、3次元グラフだけで有効です。

引数 *rotation* は、Z軸を中心としたグラフの回転です。

引数 *elevation* は、X軸を中心としたグラフの回転です。

例題

以下の例は、$ChartIDに指定されたチャートの回転と仰角を$Rotationと$Elevation変数に返します。

```
CT GET 3D VIEW(Area;$ChartID;$Rotation;$Elevation)
```

CT GET AXIS ATTRIBUTES

CT GET AXIS ATTRIBUTES (area ; object ; axis ; minorTick ; majorTick ; location ; reverse)

引数	型		説明
area	倍長整数	⇒	4D Chart エリア
object	倍長整数	⇒	オブジェクトID
axis	整数	⇒	グラフ軸 0 = 項目 1 = 系列 2 = 数値
minorTick	整数	⇐	補助目盛りマークの種類を受け取る 0 = なし 1 = 内側 2 = 外側 3 = 交差
majorTick	整数	⇐	目盛りマークの種類を受け取る 0 = なし 1 = 内側 2 = 外側 3 = 交差
location	実数	⇐	軸の位置を受け取る (軸が配置される位置)
reverse	整数	⇐	逆順を受け取る 0 = 反転なし 1 = 反転

説明

CT GET AXIS ATTRIBUTES コマンドは、引数 *minorTick*、*majorTick*、*location*、*reverse*の変数に*area*、*object*、*axis*で指定されたグラフ軸の属性を返します。このコマンドは、2次元グラフだけに適用されます。

minorTick と *majorTick* とは、*axis* 上の目盛りのことです。目盛りオプションは、各軸用の軸ダイアログボックスに設定するか、*CT SET AXIS ATTRIBUTES* コマンドを使用して設定できます。

location とは、軸が別の軸と交差するときの数値のことです。*axis* が横軸の場合には、*location* は縦軸の一番下からの増分数になります。*axis* が縦軸の場合には、*location* は横軸の左側からの増分数になります。

引数 *reverse* が1の場合には、軸上にグラフ化される項目の順序は逆になります。*reverse* が0の場合には、項目は本来の順序のままになります。

例題

以下の例は、$ChartIDに指定されたチャートの項目軸属性を$MajorTick、$MinorTick、$Location、$Reverse変数に返します。

```
$ChartID:=CT Get ID(Area;0;1)
CT GET AXIS ATTRIBUTES(Area;$ChartID;0;$MinorTick;$MajorTick;$Location;$Reverse)
```

CT GET CHART COORDINATES

CT GET CHART COORDINATES (area ; object ; left ; top ; right ; bottom)

引数	型		説明
area	倍長整数	⇒	4D Chart エリア
object	倍長整数	⇒	オブジェクトID
left	整数	⇐	チャートの開始の水平位置（ポイント単位）
top	整数	⇐	チャートの開始の垂直位置（ポイント単位）
right	整数	⇐	チャートの終了の水平位置（ポイント単位）
bottom	整数	⇐	チャートの終了の垂直位置（ポイント単位）

説明

CT GET CHART COORDINATES コマンドは、*area*内の *object* で作成されたチャートの矩形の座標を変数 *left*、*top*、*right*、*bottom* に返します。

引数 *left* は、グラフの左隅からスクリーンの左隅までの距離を示します。

引数 *top* は、グラフの頂上からスクリーンの頂上までの距離を示します。

引数 *right* は、グラフの右隅からスクリーンの右隅までの距離を示します。

引数 *bottom* は、グラフの下からスクリーンの下までの距離を示します。

CT GET CHART FILL ATTRIBUTES

CT GET CHART FILL ATTRIBUTES (area ; object ; partType ; partSpecifics ; pattern ; color)

引数	型		説明
area	倍長整数	⇒	4D Chart エリア
object	倍長整数	⇒	オブジェクトID
partType	整数	⇒	属性を取得するオブジェクトのタイプ
partSpecifics	倍長整数	⇒	属性を取得するオブジェクトの特定の部分
pattern	整数	⇐	パターン番号を受け取る (1から36まで)
color	倍長整数	⇐	カラーの値を受け取る

説明

CT GET CHART FILL ATTRIBUTES コマンドは指定されたチャートオブジェクトの塗りつぶし属性を、引数*area*、*object*、*partType*、*partSpecifics* に返します。

partType と *partSpecifics* は、属性を取得するグラフの部分を指定します。これらの引数のコードは、**引数コード**に掲載されています。

引数 *pattern* は、パターンパレット上で利用可能なパターンの1つを指定する1から36までの整数です。引数 *pattern* のコードは、**引数コード**に掲載されています。

引数 *color* は、オブジェクトの色を指定する倍長整数です。*CT Index to color* 関数や*CT RGB to color* 関数を使用することで引数 *color* の値を指定できます。

Note: 描画ツールや描画関数を使用してドキュメントに追加されたオブジェクトの属性を取得するには、**CTオブジェクト**テーマに記載されているコマンドを使用してください。

例題

以下の例は、$ChartIDに指定されたチャートの最初の系列の塗りつぶし属性を$Pattern変数と$Color変数に返します。

```
CT GET CHART FILL ATTRIBUTES(Area;$ChartID;8;100;$Pattern;$Color)
```

CT GET CHART LINE ATTRIBUTES

CT GET CHART LINE ATTRIBUTES (area ; object ; partType ; partSpecifics ; pattern ; color ; lineWidth)

引数	型		説明
area	倍長整数	⇒	4D Chart エリア
object	倍長整数	⇒	オブジェクトID
partType	整数	⇒	属性を取得するオブジェクトのタイプ
partSpecifics	倍長整数	⇒	所属を取得するオブジェクトの特定の部分
pattern	整数	⇐	パターン番号を受け取る (1から36まで)
color	倍長整数	⇐	カラーの値を受け取る (0以上)
lineWidth	実数	⇐	ポイント単位で線の太さを受け取る (0以上)

説明

CT GET CHART LINE ATTRIBUTES コマンドは、指定された線の属性を引数 *pattern*、*color*、*lineWidth*に返します。

属性を検索する対象の線は、引数 *area*、*object*、*partType*、*partSpecifics* によって指定されます。

partType と *partSpecifics* は、属性を取得するグラフの部分を指定します。これらの引数のコードは**引数コード**に掲載されています。

引数 *pattern* は、パターンパレット上で利用可能なパターンの1つを指定する1から36までの整数です。引数 *pattern* のコードは**引数コード**に掲載されています。

引数 *color* は、オブジェクトの色を指定する倍長整数です。*CT Index to color* 関数や *CT RGB to color* 関数を使用することで引数 *color* の値を指定できます。

引数 *lineWidth* は、ポイント単位で計測される線の太さです。

Note: 線ツールや*CT Draw line* 関数を使用してドキュメントに追加された線の属性を取得するには、**CTオブジェクト** テーマに記載されているコマンドを使用してください。

例題

以下の例は、$ChartIDに指定されたチャートの矩形の線属性のプロットを$Pattern、$Color、$Line変数に返します。

```
CT GET CHART LINE ATTRIBUTES(Area;$ChartID;1;0;$Pattern;$Color;$Line)
```

CT GET CHART OPTIONS

CT GET CHART OPTIONS (area ; object ; options)

引数	型		説明
area	倍長整数	⇨	4D Chart エリア
object	倍長整数	⇨	オブジェクトID
options	整数配列	⇦	オプションコードを受けるための変数を含む配列

説明

CT GET CHART OPTIONS コマンドは、選択されたグラフのオプションを返します。これらのオプションは、各チャートタイプに対して**オプション**ダイアログボックスでユーザが設定できるオプションと同じです。

各グラフタイプのコードについては**引数コード**を参照してください。

例題

以下の例は、$ChartIDに指定されたチャートのオプションをaOptionsに返します。

```
ARRAY INTEGER(aOptions;0)
CT GET CHART OPTIONS(Area;$ChartID;aOptions)
```

CT GET CHART PART

CT GET CHART PART (area ; object ; partType ; partSpecifics)

引数	型		説明
area	倍長整数	⇒	4D Chart エリア
object	倍長整数	⇒	オブジェクトID
partType	整数	⇐	エリアで選択されたオブジェクトのタイプを受け取る
partSpecifics	倍長整数	⇐	選択されたオブジェクトの特定の部分を受け取る

説明

CT GET CHART PART コマンドは、引数 *area* と *object* で指定されたグラフで現在選択されているチャートオブジェクトのコードを *partType* と *partSpecifics* 変数に返します。

引数 *partType* と *partSpecifics* は、ユーザが選択しているグラフの部分を示します。*partType* と *partSpecifics* のコードは**引数コード**に掲載されています。

例題

以下の例は、選択された系列用の部分コードを取得し、その系列の塗りつぶしパターンを無地、塗りつぶしカラーを緑に設定します。

```
CT GET CHART PART(Area;$ChartID;$Type;$Specifics)
If($Type=8) `それは系列か?
   CT SET CHART FILL ATTRIBUTES(Area;$ChartID;$Type;$Specifics;3;CT Index to color(10))
End if
```

CT Get chart picture

CT Get chart picture (area ; object ; partType ; partSpecifics) -> 戻り値

引数	型		説明
area	倍長整数	⇒	4D Chart エリア
object	倍長整数	⇒	オブジェクトID
partType	整数	⇒	ピクチャを取得するオブジェクトのタイプ
partSpecifics	倍長整数	⇒	ピクチャを取得するオブジェクトの特定の部分
戻り値	ピクチャー	⇐	ピクチャチャートの指定された系列に表示されているピクチャ

説明

CT Get chart picture 関数は、ピクチャチャートの指定されたシリーズに表示されているピクチャを返します。ピクチャはピクチャ変数内に返されます。

引数 *partType* は8である必要があり、これはチャート内の系列です。

引数 *partSpecifics* は、系列の数値に100を乗算したものと同じである必要があります。

例題

以下の例はピクチャグラフの最初の系列からピクチャをコピーし、それをクリップボードに配置します。

```
$Pict:=CT Get chart picture(Area;$ChartID;8;100)
CT PICTURE TO CLIPBOARD($Pict)
```

CT GET CHART TEXT ATTRIBUTES

CT GET CHART TEXT ATTRIBUTES (area ; object ; partType ; partSpecifics ; fontID ; fontSize ; style ; color)

引数	型		説明
area	倍長整数	⇒	4D Chart エリア
object	倍長整数	⇒	オブジェクトID
partType	整数	⇒	属性を取得するオブジェクトのタイプ
partSpecifics	倍長整数	⇒	属性を取得するオブジェクトの特定の部分
fontID	整数	⇐	フォントIDを受け取る
fontSize	整数	⇐	フォントサイズを受け取る
style	整数	⇐	フォントスタイルを受け取る
color	倍長整数	⇐	カラーの値を受け取る

説明

CT GET CHART TEXT ATTRIBUTES コマンドは、引数 *area*、*object*、*partType*、*partSpecifics* で指定されたチャートテキストの属性を取得します。

partType と *partSpecifics* は、属性を取得するグラフの部分を指定します。これらの引数のコードは**引数コード**に掲載されています。

引数 *fontID* は、システム内にあるフォントのIDです。フォントのID番号は*CT Font number* 関数を使用することによって取得できます。

引数 *fontSize* は、反転表示されたテキストやテキストオブジェクトのポイント単位のサイズです。

引数 *style* は、複数のスタイル番号の加算の結果を混合した番号です。以下の表はスタイル番号を示しています。

値	スタイル
0	標準
1	太字 (ボールド)
2	斜体 (イタリック)
4	下線 (アンダーライン)
8	アウトライン (Macintosh のみ)
16	シャドウ (Macintosh のみ)

引数 *color* は、オブジェクトの色を指定する倍長整数です。*CT Index to color* 関数や *CT RGB to color* 関数を使用することで引数 *color* の値を指定できます。

Note: テキストツールや *CT Draw text* 関数を使用してドキュメントに追加されたオブジェクトの属性を取得するには、**CT オブジェクト**テーマに記載されているコマンドを使用してください。

例題

この例題では、項目軸のテキストの属性がカスタマイズされているかを確認します。カスタマイズされていれば、それをリセットします。

```
CT GET CHART TEXT ATTRIBUTES(Area;$ChartID;8;100;$FontID;$FontSize;
$Style;$Color)
If(($FontSize#10)|($FontID#CT Font number("Geneva"))|($Color#CT Index to color(10)))
   CT SET CHART TEXT ATTRIBUTES(Area;$ChartID;5;0;CT Font number("Geneva");10;1;CT Index to
```

```
color(10))
 End if
```

CT Get chart type

CT Get chart type (area ; object) -> 戻り値

引数	型		説明
area	倍長整数	➡	4D Chart エリア
object	倍長整数	➡	オブジェクトID
戻り値	整数	↩	チャートタイプ

説明

CT Get chart type 関数は、引数 *area* と *object* で指定されたチャートのタイプを返します。

以下の表は、チャートタイプのコードです。

コード	チャートタイプ
1	面
2	棒
3	ピクチャ
4	線
5	散布図
6	円
7	ポーラー
8	2D XY
100	3D 棒
101	3D 線
102	3D 面
103	3D 等高線
104	3D 三角形
105	3D ピン

例題

以下の例で、*CT Get chart type*は$ChartIDに指定されているチャートのチャートタイプを取得するために使用されます。カラムチャートではない場合*CT SET CHART TYPE*コマンドはチャートをカラムチャートに変更します。

```
If(CT Get chart type(Area;$ChartID)#2)
   CT SET CHART TYPE(Area;$ChartID;2)
End if
```

CT GET DATE SCALE

CT GET DATE SCALE (area ; object ; minAuto ; maxAuto ; majIncrAuto ; minIncrAuto ; minimum ; maximum ; majIncrType ; majIncr ; minIncrType ; minIncr)

引数	型		説明
area	倍長整数	⇒	4D Chart エリア
object	倍長整数	⇒	オブジェクトID
minAuto	整数	⇐	デフォルトの最小値を使用しているか? 0 = 使用しない 1 = 使用する
maxAuto	整数	⇐	デフォルトの最大値を使用しているか? 0 = 使用しない 1 = 使用する
majIncrAuto	整数	⇐	デフォルトの主増分を使用しているか? 0 = 使用しない 1 = 使用する
minIncrAuto	整数	⇐	デフォルトの補助増分を使用しているか? 0 = 使用しない 1 = 使用する
minimum	日付	⇐	最小値を受け取る
maximum	日付	⇐	最大値を受け取る
majIncrType	整数	⇐	主増分タイプは何か? 1 = 日 2 = 週 3 = 月 4 = 年
majIncr	整数	⇐	主増分を受け取る
minIncrType	整数	⇐	補助増分タイプは何か? 1 = 日 2 = 週 3 = 月 4 = 年
minIncr	整数	⇐	補助増分を受け取る

説明

CT GET DATE SCALE コマンドは、デフォルト値が使用されるのかどうか、数値軸目盛り用に設定されている補助値が何であるのかを返します。*CT GET DATE SCALE* コマンドは、値が日付のときに使われます。

引数 *minAuto* と *maxAuto* は、グラフがデフォルトの最小値と最大値を現在使用しているかどうかを示します。

引数 *majIncrAuto* と *minIncrAuto* は、グラフがデフォルトの主増分と補助増分を現在使用しているかどうかを示します。

引数 *minimum* と *maximum* は、軸ダイアログボックスでユーザが設定するか、または設計者が *CT SET DATE SCALE* コマンドを使用するかによって設定される最小値と最大値です。

引数 *majIncrType* と *minIncrType* は、*majorIncr* と *majorIncr* が指定される単位用のコードです。

引数 *majorIncr* と *minorIncr* は、軸ダイアログボックスでユーザが設定するか、または設計者が *CT SET DATE SCALE* コマンドを使用するかによって設定される主増分と補助増分です。

例題

以下の例は、$ChartIDに指定されたチャートの目盛りデータを返します。

```
 CT GET DATE
SCALE(Area;$ChartID;$MinA;$MaxA;$MajA;$MinA;$Minimum;$Maximum;$MajType;$MajorInc;$MinType;
 $MinorInc)
```

CT GET DEPTH

CT GET DEPTH (area ; object ; horizontal ; vertical)

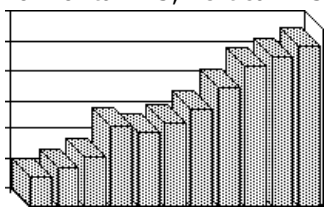
引数	型		説明
area	倍長整数	⇒	4D Chart エリア
object	倍長整数	⇒	オブジェクトID
horizontal	整数	⇐	ポイント単位で横方向のオフセットを受け取る
vertical	整数	⇐	ポイント単位で縦方向にオフセットを受け取る

説明

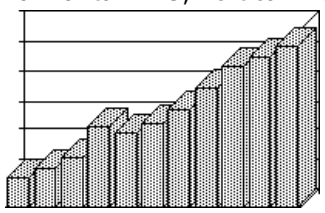
CT GET DEPTH コマンドは、引数 *area*、*object* で指定されたグラフの横方向と縦方向のオフセット (奥行き) を設定します。このコマンドは、2次元グラフだけで有効です。

引数 *horizontal* は、ポイント単位で測定した横方向のオフセットです。正の値はオフセットが右方向になることを示し、負の値はオフセットが左方向になることを示します。

horizontal = 5, vertical = 5



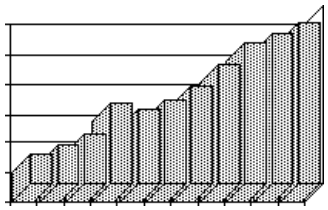
horizontal = -5, vertical = 5



引数 *vertical* は、ポイント単位で測定した縦方向のオフセットです。正の値はX軸から奥に入る距離を示し、負の値はX軸から前に出る距離を示します。

水平または垂直オフセットをゼロで設定するには次元で3D効果を取り除きます。

horizontal = 5, vertical = - 5



horizontal または *vertical* オフセットをゼロで設定と次元の3D効果が取り除かれます。

例題

以下の例は、$ChartIDに指定されたチャートの奥行きをチェックし、ユーザによって変更されていなかった場合には奥行きを設定します。

```
CT GET DEPTH(vArea;$ChartID;$Horiz;$Vert)
If($Horiz=0) & ($Vert=0)
```

```
   CT SET DEPTH(vArea;$ChartID;10;10)
End if
```

CT GET LABEL ATTRIBUTES

CT GET LABEL ATTRIBUTES (area ; object ; axis ; position ; orientation ; format ; frequency)

引数	型		説明
area	倍長整数	⇒	4D Chart エリア
object	倍長整数	⇒	オブジェクトID
axis	整数	⇒	グラフ軸 0 = 項目 1 = 系列 2 = 数値
position	整数	⇐	ラベルの位置を受け取る 0 = なし 1 = 上 2 = 左 3 = 下 4 = 右
orientation	整数	⇐	ラベルの方向を受け取る 0 = 通常 1 = 垂直 2 = 右回転 3 = 左回転 4 = 互い違い 5 = ワードラップ
format	文字	⇐	ラベルフォーマットを受け取る
frequency	整数	⇐	ラベル表示を受け取る

説明

CT GET LABEL ATTRIBUTES コマンドは、引数 *area*、*object*、*axis* で指定された軸ラベルの属性を *position*、*orientation*、*format* 変数に返します。

引数 *position* は、グラフに相対的な軸のラベル位置です。

引数 *orientation* は、各ラベルの方向です。以下のラベル方向はそれぞれの軸で使用できます。

ORIENTATION					
Normal	Vertical	Rotated Left	Rotated Right	Staggered	Wrap
Label	L a b e l	Label	Label	Label1 Label2 Label3	Lab el

引数 *format* は、ラベルテキストの表示フォーマットです。表示フォーマットが "通常" のときには *format* 変数に空の文字列 ("") が返ります。表示フォーマットで使用される特殊文字に関する詳細は、*4D Design Reference* を参照してください。

オプションの引数 *frequency* はラベルが表示する系列や軸の項目の頻度を受け取ります。この引数はラベルが表示されたグラフのみの項目／系列の数を返します。デフォルトでは、この引数は1が返ります（すべてのラベルが表示されている場合）。軸が軸の値の場合は、*frequency* は-32000が返ります。引数 *frequency* の設定の詳しい説明は *CT SET LABEL ATTRIBUTES* コマンドの記述を参照してください。

Note: 軸ラベルのテキスト属性を取得するには、*CT GET CHART TEXT ATTRIBUTES* コマンドを使用してください。

例題

以下の例は、$ChartIDに指定されたチャートの項目軸ラベル属性を$Position、$Orient、$Format変数に返します。

```
CT GET LABEL ATTRIBUTES(Area;$ChartID;0;$Position;$Orient;$Format)
```

CT GET LEGEND ATTRIBUTES

CT GET LEGEND ATTRIBUTES (area ; object ; display ; orientation ; reverseOrder ; reverseKey ; location ; horizOffset ; vertOffset)

引数	型		説明
area	倍長整数	⇒	4D Chart エリア
object	倍長整数	⇒	オブジェクトID
display	整数	⇐	凡例を表示するか? 0 = 表示しない 1 = 表示する
orientation	整数	⇐	凡例内の系列の方向を受け取る 0 = 水平 1 = 垂直
reverseOrder	整数	⇐	順番を逆にするか? 0 = 逆順にしない 1 = 逆順にする
reverseKey	整数	⇐	キーとテキストは逆になっているか? 0 = 反転しない 1 = 反転する
location	整数	⇐	位置コードを受け取る
horizOffset	整数	⇐	プロットの左側からの水平方向のオフセットをポイント単位で受け取る
vertOffset	整数	⇐	プロットの上側からの垂直方向のオフセットをポイント単位で受け取る

説明

CT GET LEGEND ATTRIBUTES コマンドは、引数 *display*、*orientation*、*reverseOrder*、*reverseKey*、*location*、*horizOffset*、*vertOffset*の変数に引数 *area*と*object* で指定された凡例の属性を返します。

引数 *display* は、凡例が表示されるかどうかを示します。

引数 orientation は、系列が凡例内で縦方向に表示されるか、横方向に表示されるのかを示します。以下の図は、縦方向の凡例と横方向の凡例を示しています。

引数 *reverseOrder* は、凡例内の系列の順序が逆順になるかどうかを示します。

引数 *reverseKey* は、系列の一意なパターンと色を説明する系列ラベルとキーが逆になるかどうかを示します。デフォルトでは、キーがラベルの左側に表示されます。

以下の表は、引数 *location* のコードを示しています。

コード	位置
0	凡例の位置は、組み込み位置オプションの1つではありません。
1	左上
2	左下
3	右上
4	右下
5	左
6	右
7	上
8	下

引数 *horizOffset* と *vertOffset* は、*location* が組み込み凡例位置にはないとき (*location* = 0) に使用されます。*horizOffset*

は、グラフの左側から凡例の左側までをポイント単位で表します。*vertOffset* は、グラフの上の端から凡例の上の端までをポイント単位で表します。

Note: 凡例テキストのテキスト属性を取得するには、*CT GET CHART TEXT ATTRIBUTES* コマンドを使用してください。

例題

以下の例は、$ChartIDに指定されたチャートの凡例テキスト属性を取得します。

```
 CT GET LEGEND
ATTRIBUTES(Area;$ChartID;$Display;$Orient;$ReverseO;$ReverseK;$Location;$HorizOff;$VertOff)
```

CT Get legend text

CT Get legend text (area ; object ; legendItem) -> 戻り値

引数	型		説明
area	倍長整数	⇒	4D Chart エリア
object	倍長整数	⇒	オブジェクトID
legendItem	整数	⇒	凡例項目番号
戻り値	テキスト	⇐	指定された凡例項目のテキスト

説明

CT Get legend text コマンドは、指定された凡例項目にテキストを返します。

引数 *legendItem* は、凡例内での系列の番号 (あるいは、円グラフでの項目) です。ただし、凡例が逆順になった場合には、*legendItem* は逆順ではなく正順を反映します。

例題

以下の例は、$ChartIDに指定されたチャート内の最初の系列の凡例テキストを返します。

```
$Text:=CT Get legend text(Area;$ChartID;1)
```

CT GET REAL SCALE

CT GET REAL SCALE (area ; object ; minAuto ; maxAuto ; majIncrAuto ; minIncrAuto ; minimum ; maximum ; majorIncr ; minorIncr)

引数	型		説明
area	倍長整数	⇒	4D Chart エリア
object	倍長整数	⇒	オブジェクトID
minAuto	整数	⇐	デフォルトの最小値を使用しているか? 0 = 使用しない 1 = 使用する
maxAuto	整数	⇐	デフォルトの最大値を使用しているか? 0 = 使用しない 1 = 使用する
majIncrAuto	整数	⇐	デフォルトの主増分を使用しているか? 0 = 使用しない 1 = 使用する
minIncrAuto	整数	⇐	デフォルトの補助増分を使用しているか? 0 = 使用しない 1 = 使用する
minimum	実数	⇐	最小値を受け取る
maximum	実数	⇐	最大値を受け取る
majorIncr	実数	⇐	主増分を受け取る
minorIncr	実数	⇐	補助増分を受け取る

説明

CT GET REAL SCALE コマンドは、デフォルト値が使用されるのかどうか、数値軸目盛用に設定されている補助値が何であるのかを返します。*CT GET REAL SCALE* コマンドは、値が実数、整数、または倍長整数のときに使用されます。

引数*minAuto* と *maxAuto* は、グラフがデフォルトの最小値と最大値を現在使用しているかどうかを示します。

引数 *majIncrAuto* と *minIncrAuto* は、グラフがデフォルトの主増分と補助増分を現在使用しているかどうかを示します。

引数 *minimum* と *maximum* は、軸ダイアログボックスでユーザが設定するか、または設計者が *CT SET REAL SCALE* コマンドを使用するかによって設定される最小値と最大値です。

引数 *majorIncr* と *minorIncr* は軸ダイアログボックスでユーザが設定するか、または設計者が *CT SET REAL SCALE* コマンドを使用するかによって設定される主増分と補助増分です。

例題

以下の例は、$ChartIDに指定されたチャートの目盛りデータを返します。

```
CT GET REAL SCALE(Area;$ChartID;$MinA;$MaxA;$MajA;$MinA;$Minimum;$Maximum;$MajorInc;$MinorInc)
```

CT GET TIPS ATTRIBUTES

CT GET TIPS ATTRIBUTES (area ; object ; axis ; toolBar ; status ; contents ; format ; formatX ; method)

引数	型		説明
area	倍長整数	⇒	4D Chart エリア
object	倍長整数	⇒	オブジェクトID
axis	整数	⇐	選択した軸 1 = 項目 2 = 系列 4 = 数値
toolBar	整数	⇐	廃止 (使用しないでください)
status	整数	⇐	モードを表示 0 = ヒントなし 1 = 常にヒントを表示 2 = 要求時にのみヒントを表示
contents	整数	⇐	ヒントの内容 0 = 値のみ 1 = パーセンテージのみ 2 = 値とパーセンテージ
format	文字	⇐	Ｚ軸値の表示フォーマット
formatX	文字	⇐	Ｘ軸値の表示フォーマット
method	文字	⇐	実行するメソッドの名前

説明

CT GET TIPS ATTRIBUTES コマンドは引数 *axis*、*toolbar*、*status*、*contents*、*format*、*formatX*、*method* 変数に、*area* と *object* で指定したヒントの属性を返します。

引数 *axis* はヒントが利用されている軸を示します。これは追加したいくつかの軸の数からの結果の数で構成しています。軸の数は：

数値	軸
1	項目軸
2	系列軸
4	値軸

引数 *toolbar* は、現在サポートされておらず、有効な値を返しません。

引数 *status* はヒントの設定の表示を示します。ヒントは常に実行されているか、要求時に実行するか (Windows版では**Ctrl**、Macintosh版では**command**キーが押された時) 、または実行しないようにできます。

引数 *contents* でタイプ情報を表示されているか確認することができます。情報は値、パーセンテージまたは両方のいずれかです。

引数 *format* はＺ軸のヒントの値のフォーマットを表示します。フォーマットが "通常" の場合は、空の文字列 ("") が *format* に返されます。フォーマット表示に使用する特殊文字に関する詳細は、4D Design Referenceマニュアルを参照してください。

引数 *formatX* は、Ｘ軸 (XYグラフのみ) に適用されることを除外すれば、*format* と同じです。

引数 *method* は、ヒントが表示されるのと同時に実行されているメソッドの名前です。

CT GET TITLE ATTRIBUTES

CT GET TITLE ATTRIBUTES (area ; object ; axis ; position ; orientation ; title)

引数	型		説明
area	倍長整数	⇒	4D Chart エリア
object	倍長整数	⇒	オブジェクトID
axis	整数	⇒	属性を設定するための軸 0 = 項目 1 = 系列 2 = 数値
position	整数	⇐	タイトルの位置を取得する 0 = なし 1 = 上 2 = 左 3 = 下 4 = 右
orientation	整数	⇐	タイトルの方向を取得する 0 = 通常 1 = 垂直 2 = 右回転 3 = 左回転
title	文字	⇐	タイトルのテキストを取得する

説明

CT GET TITLE ATTRIBUTES コマンドは、引数 *area*、*object*、*axis* で指定された軸タイトルの位置、方向、テキストを返します。

引数 *position* は、グラフに相対的なタイトルの位置です。

引数 *orientation* は、タイトルの方向です。

引数 *title* は、タイトルのテキストです。タイトルの長さは255文字以内です。

Note: タイトルのテキスト属性を取得するには、*CT GET CHART TEXT ATTRIBUTES* コマンドを使用してください。

例題

以下の例は$Position、$Orient、$Title変数に項目軸タイトル属性を返します。

```
CT GET TITLE ATTRIBUTES(Area;$ChartID;0;$Position;$Orient;$Title)
```

CT GET VALUE ATTRIBUTES

CT GET VALUE ATTRIBUTES (area ; object ; position ; display ; orientation ; format)

引数	型		説明
area	倍長整数	⇒	4D Chart エリア
object	倍長整数	⇒	オブジェクトID
position	整数	⇐	位置の値を受け取る -1 = 変更なし 0 = なし 1 = 外部の上 2 = 外部の下 3 = 内部の上 4 = 内部の中央 5 = 内部の下 6 = 軸の上 8 =軸の左 9 = 軸の右 10 = 下
display	整数	⇐	タイプの情報を受け取る 1 = 値 2 = パーセンテージ 3 = 項目 4 = 値とパーセンテージ 5 = 項目とパーセンテージ
orientation	整数	⇐	方向の値を受け取る -1 = 変更なし 0 = 標準 1 = 垂直 2 = 右の上 3 = 左の上
format	文字	⇐	値のフォーマット

説明

CT GET VALUE ATTRIBUTES コマンドは *area* と *object* で指定したグラフの値の属性を返します。

position は位置の値を指定した整数です。

display は値として表示している情報のタイプを指定した整数です。

orientation は方向の値を指定した整数です。

CT GET X DATE SCALE

CT GET X DATE SCALE (area ; object ; minAuto ; maxAuto ; majIncrAuto ; minIncrAuto ; minimum ; maximum ; majIncrType ; majIncr ; minIncrType ; minorIncr)

引数	型		説明
area	倍長整数	⇒	4D Chart エリア
object	倍長整数	⇒	オブジェクトID
minAuto	整数	⇐	デフォルトの最小値を使用しているか? 0 = 使用しない 1 = 使用する
maxAuto	整数	⇐	デフォルトの最大値を使用している? 0 = 使用しない 1 = 使用する
majIncrAuto	整数	⇐	デフォルトの主増分を使用しているか? 0 = 使用しないNo 1 = 使用する
minIncrAuto	整数	⇐	デフォルトの補助増分を使用しているか? 0 = 使用しない 1 = 使用する
minimum	日付	⇐	最小値を受け取る
maximum	日付	⇐	最大値を受け取る
majIncrType	整数	⇐	主増分タイプは何か? 1 = 日 2 = 週 3 = 月 4 = 年
majIncr	整数	⇐	主増分を受け取る
minIncrType	整数	⇐	補助増分タイプは何か? 1 = 日 2 = 週 3 = 月 4 = 年
minorIncr	整数	⇐	補助増分タイプを受け取る

説明

CT GET X DATE SCALE コマンドは、デフォルト値が使用されるのかどうか、XYチャートのX軸目盛り用に設定されている補助値が何であるのかを返します。同じチャートタイプの値軸 (Z軸) での *CT GET DATE SCALE* コマンドは、値が日付のときに使われます。

引数 *minAuto* と *maxAuto* は、グラフがデフォルトの最小値と最大値を現在使用しているかどうかを示します。

引数 *majIncrAuto* と *minIncrAuto* は、グラフがデフォルトの主増分と補助増分を現在使用しているかどうかを示します。

引数 *minimum* と *maximum* は、軸ダイアログボックスでユーザが設定するか、または設計者が *CT SET X DATE SCALE* コマンドを使用するかによって設定される最小値と最大値です。

引数 *majIncrType* と *minIncrType* は、 *majorIncr* と *minorIncr* が指定される単位用のコードです。

引数 *majorIncr* と *minorIncr* は、軸ダイアログボックスでユーザが設定するか、または設計者が *CT SET X DATE SCALE* コマンドを使用するかによって設定される主増分と補助増分です。

CT GET X REAL SCALE

CT GET X REAL SCALE (area ; object ; minAuto ; maxAuto ; majIncrAuto ; minIncrAuto ; minimum ; maximum ; majorIncr ; minorIncr)

引数	型		説明
area	倍長整数	⇒	4D Chart エリア
object	倍長整数	⇒	オブジェクトID
minAuto	整数	⇐	デフォルトの最小値を使用しているか? 0 =使用しない 1 = 使用する
maxAuto	整数	⇐	デフォルトの最大値を使用しているか? 0 = 使用しない 1 = 使用する
majIncrAuto	整数	⇐	デフォルトの主増分を使用しているか? 0 = 使用しない 1 = 使用する
minIncrAuto	整数	⇐	デフォルトの補助増分を使用していか? 0 = 使用しない 1 = 使用する
minimum	実数	⇐	最小値を受け取る
maximum	実数	⇐	最大値を受け取る
majorIncr	実数	⇐	主増分を受け取る
minorIncr	実数	⇐	補助増分を受け取る

説明

CT GET X REAL SCALE コマンドは、デフォルト値が使用されるのか、XYチャートでX軸目盛用に設定されている補助値が何であるのかを返します。グラフタイプと同じ軸の値 (Z軸) では *CT GET REAL SCALE* コマンドは、値が実数、整数、または倍長整数のときに使用されます。

引数 *minAuto* と *maxAuto* は、グラフがデフォルトの最小値と最大値を現在使用しているかどうかを示します。

引数 *majIncrAuto* と *minIncrAuto* は、グラフがデフォルトの主増分と補助増分を現在使用しているかどうかを示します。

引数 *minimum* と *maximum* は、軸ダイアログボックスでユーザが設定するか、または設計者が *CT SET X REAL SCALE* コマンドを使用するかによって設定される最小値と最大値です。

引数 *majorIncr* と *minorIncr* は、軸ダイアログボックスでユーザが設定するか、または設計者が *CT SET X REAL SCALE* コマンドを使用するかによって設定される主増分と補助増分です。

例題

以下の例は、$ChartIDに指定されたチャートの目盛りデータを返します。

```
 CT GET X REAL
SCALE(Area;$ChartID;$MinA;$MaxA;$MajA;$MinA;$Minimum;$Maximum;$MajorInc;$MinorInc)
```

CT SET 3D VIEW

CT SET 3D VIEW (area ; object ; rotation ; elevation)

引数	型		説明
area	倍長整数	➡	4D Chart エリア
object	倍長整数	➡	オブジェクトID
rotation	実数	➡	度数単位の回転 (0から90まで、それ以外の値のは何も行わない)
elevation	実数	➡	度数単位の仰角 (0から90まで、それ以外の値のは何も行わない)

説明

CT SET 3D VIEW コマンドは、引数 *area* と *object* で指定されたグラフの回転と仰角を設定します。このコマンドは、3次元グラフだけで有効です。

引数 *rotation* は、Z軸を中心としたグラフの回転です。*rotation* は0から90までの値である必要があります。

引数 *elevation* は、X軸を中心としたグラフの回転です。*elevation* は0から90までの値である必要があります。

例題

以下の例は、$ChartIDに指定されているチャートの回転と仰角を30度に設定します。

```
CT SET 3D VIEW(Area;$ChartID;30;30)
```

CT SET AXIS ATTRIBUTES

CT SET AXIS ATTRIBUTES (area ; object ; axis ; minorTick ; majorTick ; location ; reverse)

引数	型		説明
area	倍長整数	➡	4D Chart エリア
object	倍長整数	➡	オブジェクトID
axis	整数	➡	グラフ軸 0 = 項目 1 = 系列 2 = 数値
minorTick	整数	➡	補助目盛りマークの種類 -1 = 変更なし 0 = なし 1 = 内側 2 = 外側 3 = 交差
majorTick	整数	➡	主目盛りマークの種類 -1 = 変更なし 0 = なし 1 = 内側 2 = 外側 3 = 交差
location	実数	➡	軸の位置 (軸が配置される位置)
reverse	整数	➡	逆順 0 = 反転なし 1 = 反転 -1 = 変更

説明

CT SET AXIS ATTRIBUTES コマンドは、*area*、*object*、*axis* で指定されたグラフ軸の属性を設定します。このコマンドは、2次元グラフだけに適用されます。

minorTick と *majorTick* とは、*axis* 上の目盛りのことです。

location とは、軸が別の軸と交差するときの数値のことです。*axis* が横軸の場合には、*location* は縦軸の一番下からの増分数になります。*axis* が縦軸の場合には、*location* は横軸の左側からの増分数になります。

引数 *reverse* が1の場合には、軸上にグラフ化される項目の順序は逆になります。*reverse* が0の場合には、項目は本来の順序のままになります。

例題

以下の例は、$ChartIDに指定されたチャートの項目軸属性を変更します。

```
CT SET AXIS ATTRIBUTES(vArea;$ChartID;0;0;3;160;1)
```

CT SET CHART COORDINATES

CT SET CHART COORDINATES (area ; object ; left ; top ; right ; bottom)

引数	型		説明
area	倍長整数	➡	4D Chart エリア
object	倍長整数	➡	オブジェクトID
left	整数	➡	チャートの開始の水平位置（ポイント単位）
top	整数	➡	チャート開始の垂直位置（ポイント単位）
right	整数	➡	チャートの終了の水平位置（ポイント単位）
bottom	整数	➡	チャートの終了の垂直位置（ポイント単位）

説明

CT SET CHART COORDINATES コマンドは、引数 *left*、*top*、*right*、*bottom* として渡した値を使用して、4D Chart内のグラフの位置を修正します。

引数 *left* は、グラフの左隅からスクリーンの左隅までの距離を示します。

引数 *top* は、グラフの頂上からスクリーンの頂上までの距離を示します。

引数 *right* は、グラフの右隅からスクリーンの右隅までの距離を示します。

引数 *bottom* はグラフの下からスクリーンの下までの距離を示します。

CT SET CHART FILL ATTRIBUTES

CT SET CHART FILL ATTRIBUTES (area ; object ; partType ; partSpecifics ; pattern ; color)

引数	型		説明
area	倍長整数	➡	4D Chart エリア
object	倍長整数	➡	オブジェクトID
partType	整数	➡	属性を設定するオブジェクトのタイプ
partSpecifics	倍長整数	➡	属性を設定するオブジェクトの特定の部分
pattern	整数	➡	パターン番号 (1から36まで) -1 = 変更なし
color	倍長整数	➡	カラーの値 -1 = 変更なし

説明

CT SET CHART FILL ATTRIBUTES コマンドは、引数 *area*、*object*、*partType*、*partSpecifics* で指定されたチャートオブジェクトの塗りつぶし属性を取得します。

partType と *partSpecifics* は、属性を取得するグラフの部分を指定します。これらの引数のコードは**引数コード**に掲載されています。

引数 *pattern* は、パターンパレット上で利用可能なパターンの1つを指定する1から36までの整数です。この引数用のコードは**引数コード**に掲載されています。

引数 *color* は、オブジェクトの色を指定する倍長整数です。*CT Index to color* 関数や*CT RGB to color* 関数を使用することで引数 *color* の値を指定できます。これらの関数に関する詳細は**CTユーティリティ**に記載されているコマンドを参照してください。

Note: 描画ツールや描画関数を使用してドキュメントに追加されたオブジェクトの属性を取得するには**CTオブジェクト**テーマに記載されているコマンドを使用してください。

例題

以下の例は、$ChartIDに指定されたチャートの最初の系列の塗りつぶし属性を変更します。色は赤に、パターンは塗りつぶしに設定されます。

```
CT SET CHART FILL ATTRIBUTES(Area;$ChartID;8;100;3;CT Index to color(4))
```

CT SET CHART LINE ATTRIBUTES

CT SET CHART LINE ATTRIBUTES (area ; object ; partType ; partSpecifics ; pattern ; color ; lineWidth)

引数	型		説明
area	倍長整数	⇒	4D Chart エリア
object	倍長整数	⇒	オブジェクトID
partType	整数	⇒	属性を取得するオブジェクトのタイプ
partSpecifics	倍長整数	⇒	属性を取得するオブジェクトの特定な部分
pattern	整数	⇒	パターン番号 (1から36まで) -1 = 変更なし
color	倍長整数	⇒	カラーの値 -1 = 変更なし
lineWidth	実数	⇒	ポイント単位で線の太さ (0以上) -1 = 変更なし

説明

CT SET CHART LINE ATTRIBUTES コマンドは、引数 *area*、*object*、*partType*、*partSpecifics* で指定された線の属性を設定します。

partType と *partSpecifics* は、属性を取得するグラフの部分を指定します。これらの引数のコードは**引数コード引数コード**に掲載されています。

引数 *pattern* は、パターンパレット上で利用可能なパターンの1つを指定する1から36までの整数です。この引数のコードは**引数コード**に掲載されています。

引数 *color* は、オブジェクトの色を指定する倍長整数です。*CT Index to color* 関数や*CT RGB to color* 関数を使用することで引数 *color* の値を指定できます。これらの関数の詳細については**CTユーティリティ**テーマに記載されているコマンドを参照してください。

引数 *lineWidth* は、ポイント単位で計測される線の太さです。

Note: 線ツールや*CT Draw line* 関数を使用してドキュメントに追加された線の属性を取得するには**CTオブジェクト**テーマに記載されているコマンドを使用してください。

例題

以下の例は、$ChartIDに指定されたチャートに対して、矩形の線属性のプロットを変更します。色は緑に変更され、線の幅は3ポイントに、パターンは塗りつぶしに設定されます。

```
CT SET CHART LINE ATTRIBUTES(Area;$ChartID;1;0;3;CT Index to color(10);3)
```

CT SET CHART OPTIONS

CT SET CHART OPTIONS (area ; object ; options)

引数	型		説明
area	倍長整数	⇒	4D Chart エリア
object	倍長整数	⇒	オブジェクトID
options	整数配列	⇒	オプションコードを含む配列

説明

CT SET CHART OPTIONS コマンドは、選択されたグラフのオプションを設定します。これらのオプションは、各チャートタイプに対してオプションダイアログボックスでユーザが設定できるオプションと同じです。

各グラフタイプの *options* コードについては**引数コード**を参照してください。

例題

以下の例は、$ChartIDで指定されたカラムチャートのチャートオプションを横方向、スタック比例、100パーセントオーバーラップ、25パーセントのギャップに設定します。

```
ARRAY INTEGER(aOptions;4)
aOptions{1}:=1
aOptions{2}:=2
aOptions{3}:=100
aOptions{4}:=25
CT SET CHART OPTIONS(Area;$ChartID;aOptions)
```

CT SET CHART PICTURE

CT SET CHART PICTURE (area ; object ; partType ; partSpecifics ; picture)

引数	型		説明
area	倍長整数	➡	4D Chart エリア
object	倍長整数	➡	オブジェクトID
partType	整数	➡	属性を設定するオブジェクトのタイプ
partSpecifics	倍長整数	➡	属性を設定するオブジェクトの特定の部分
picture	ピクチャー	➡	ピクチャグラフの系列に貼り付けるピクチャ

説明

CT SET CHART PICTURE コマンドは、指定された系列に引数 *picture* を貼り付けます。

引数 *object* は、ピクチャグラフである必要があります。

引数 *partType* は8である必要があり、これはグラフ内の系列です。

引数 *partSpecifics* は、系列の数値に100を乗算したものと同じである必要があります。

例題

以下の例は、グラフがピクチャグラフである場合には、クリップボードから選択したピクチャをグラフの最初の系列にコピーします。

```
$ChartID:=CT Get ID(Area;0;1)
If(CT Get chart type(Area;$ChartID)=3)
   $Pict:=CT Clipboard to picture
   CT SET CHART PICTURE(Area;$ChartID;8;100;$Pict)
End if
```

CT SET CHART TEXT ATTRIBUTES

CT SET CHART TEXT ATTRIBUTES (area ; object ; partType ; partSpecifics ; fontID ; fontSize ; style ; color)

引数	型		説明
area	倍長整数	➡	4D Chart エリア
object	倍長整数	➡	オブジェクトID
partType	整数	➡	属性を取得するオブジェクトのタイプ
partSpecifics	倍長整数	➡	属性を取得するオブジェクトの特定の部分
fontID	整数	➡	フォントID -1 = 変更なし
fontSize	整数	➡	フォントサイズ -1 = 変更なし
style	整数	➡	フォントスタイルのコード -1 = 変更なし
color	倍長整数	➡	カラーの値 -1 = 変更なし

説明

CT SET CHART TEXT ATTRIBUTES コマンドは、引数 *area*、*object*、*partType*、*partSpecifics* で指定されたチャートテキストの属性を取得します。

partType と *partSpecifics* は、属性を取得するグラフの部分を指定します。これらの引数のコードは**引数コード**に掲載されています。

引数 *fontID* は、システム内にあるフォントのIDです。フォントのID番号は*CT Font number* 関数を使用することによって取得できます。

引数 *fontSize* は、反転表示されたテキストやテキストオブジェクトのポイント単位のサイズです。

引数 *style* は、複数のスタイル番号の加算の結果を混合した番号です。以下の表はスタイル番号を示しています。

値	スタイル
0	標準
1	太字 (ボールド)
2	斜体 (イタリック)
4	下線 (アンダーライン)
8	アウトライン (Macintosh のみ)
16	シャドウ (Macintosh のみ)

引数 *color* は、オブジェクトの色を指定する倍長整数です。*CT Index to color* 関数や*CT RGB to color* 関数を使用することで引数 *color* の値を指定できます。

CT GET CHART TEXT ATTRIBUTES コマンドの例題を参照してください。

Note: テキストツールや*CT Draw text* 関数を使用してドキュメントに追加されたオブジェクトの属性を取得するには**CTオブジェクト**テーマに記載されているコマンドを使用してください。

例題

*CT GET CHART TEXT ATTRIBUTES*コマンドの例題参照

CT SET CHART TYPE

CT SET CHART TYPE (area ; object ; type)

引数	型		説明
area	倍長整数	➡	4D Chart エリア
object	倍長整数	➡	オブジェクトID
type	整数	➡	チャートタイプ

説明

CT SET CHART TYPE コマンドは、指定されたチャートのタイプを *type* に変更します。

引数 *type* は、2次元グラフに対しては2Dグラフタイプ、3次元グラフに対しては3Dグラフタイプである必要があります。

以下の表は、チャートタイプのコードです。

コード	チャートタイプ
1	面
2	棒
3	ピクチャ
4	線
5	散布図
6	円
7	ポーラー
8	2D XY
100	3D 棒
101	3D 線
102	3D 面
103	3D 等高線
104	3D 三角形
105	3D ピン

例題

*CT Get chart type*の例題参照

CT SET DATE SCALE

CT SET DATE SCALE (area ; object ; minAuto ; maxAuto ; majIncrAuto ; minIncrAuto ; minimum ; maximum ; majIncrType ; majorIncr ; minIncrType ; minorIncr)

引数	型		説明
area	倍長整数	⇒	4D Chart オブジェクト
object	倍長整数	⇒	オブジェクトID
minAuto	整数	⇒	デフォルトの最小値を使用しているか? -1 = 変更なし 0 = 使用しない 1 = 使用する
maxAuto	整数	⇒	デフォルトの最大値を使用しているか? -1 = 変更なし 0 = 使用しない 1 = 使用する
majIncrAuto	整数	⇒	デフォルトの主増分を使用しているか? -1 = 変更なし 0 = 使用しない 1 = 使用する
minIncrAuto	整数	⇒	デフォルトの補助増分を使用しているか? -1 = 変更なし 0 = 使用しない 1 = 使用する
minimum	日付	⇒	最小値
maximum	日付	⇒	最大値
majIncrType	整数	⇒	デフォルトの主増分タイプを使用しているか? -1 = 変更なし 1 = 日 2 = 週 3 = 月 4 = 年
majorIncr	整数	⇒	主増分
minIncrType	整数	⇒	デフォルトの補助増分タイプを使用している? -1 = 変更なし 1 = 日 2 = 週 3 = 月 4 = 年
minorIncr	整数	⇒	補助増分

説明

CT SET DATE SCALE コマンドは、デフォルト値を使用するかどうかの指定、数値軸目盛り用の代替値の設定に使用します。*CT SET DATE SCALE* コマンドは、値が日付のときに使われます。

引数 *minAuto* と *maxAuto* は、グラフがデフォルトの最小値と最大値を現在使用しているかどうかを示します。

引数 *majIncrAuto* と *minIncrAuto* は、デフォルトの最小値と最大値を使用するかどうかを示します。

引数 *minimum* と *maximum* は、代替最小値と代替最大値です。

引数 *majIncrType* と *minIncrType* は、*majorIncr* と*minorIncr* が指定される単位用のコードです。

引数 *majorIncr* と *minorIncr* は、代替主増分と代替補助増分です。

例題

以下の例は、データベースからチャートを作成し、目盛り値を設定します。

```
ARRAY INTEGER(aYFields;2)
aYFields{1}:=2
aYFields{2}:=3
$ChartID:=CT Chart selection(Area;2;1;1;Table(->[Customer]);Field(->[Customer]Customer
Type);aYFields)
CT SET DATE SCALE(Area;$ChartID;0;0;0;0;0;!01/01/90!;!12/30/95!;4;1;3;1)
```

CT SET DEPTH

CT SET DEPTH (area ; object ; horizontal ; vertical)

引数	型		説明
area	倍長整数	⇒	4D Chart エリア
object	倍長整数	⇒	オブジェクトID
horizontal	整数	⇒	ポイント単位の横方向のオフセット (-32000よりも大きい必要がある)
vertical	整数	⇒	ポイント単位の縦方向のオフセット (-32000よりも大きい必要がある)

説明

CT SET DEPTH コマンドは、引数 *area*と*object* で指定されたグラフの横方向と縦方向のオフセット (奥行き) を設定します。このコマンドは2次元グラフだけで有効です。

引数 *horizontal* は、ポイント単位で測定した横方向のオフセットです。正の値はオフセットが右方向になることを示し、負の値はオフセットが左方向になることを示します。

引数 *vertical* は、ポイント単位で測定した縦方向のオフセットです。正の値はX軸から奥に入る距離を示し、負の値はX軸から前に出る距離を示します。

横方向と縦方向の奥行きの実例については、*CT GET DEPTH* コマンドを参照してください。

例題

CT GET DEPTH コマンドの例を参照

CT SET LABEL ATTRIBUTES

CT SET LABEL ATTRIBUTES (area ; object ; axis ; position ; orientation ; format ; frequency)

引数	型		説明
area	倍長整数	➡	4D Chart エリア
object	倍長整数	➡	オブジェクトID
axis	整数	➡	グラフ軸 0 = 項目 1 = 系列 2 = 数値
position	整数	➡	ラベルの位置 -1 = 変更なし 0 = なし 1 = 上 2 = 左 3 = 下 4 = 右
orientation	整数	➡	ラベルの方向 -1 = 変更なし 0 = 通常 1 = 垂直 2 = 右回転 3 = 左回転 4 = 互い違い 5 = ワードラップ
format	文字	➡	ラベルフォーマット
frequency	整数	➡	ラベル表示の頻度 (1-255)

説明

CT SET LABEL ATTRIBUTES コマンドは、引数 *area*、*object*、*axis* で指定された軸ラベルの属性を *position*、*orientation*、*format* 変数に返します。

引数 *position* は、グラフに相対的な軸のラベル位置です。

引数 *orientation* は、各ラベルの方向です。各方向オプションを示す表については、*CT GET LABEL ATTRIBUTES* を参照してください。

引数 *format* は、ラベルテキストの表示フォーマットです。表示フォーマットが "通常" のときには *format* 変数に空の文字列 ("") が返ります。表示フォーマットで使用される特殊文字に関する詳細は、*4D Design Reference* を参照してください。

オプションの引数 *frequency* により、項目軸または系列軸に対してラベル表示の頻度を設定することができます。多くの異なる値が項目軸や系列軸上にあり、グラフ上に十分なスペースがないため、すべてのラベルを印刷できない場合、この引数はとても役に立ちます。引数 *frequency* を1以上の値に設定することにより、項目軸あるいは系列軸に対して指定した印刷のパーセンテージをコントロールすることができます。例えば、引数 *frequency* に3を設定すると、ラベルが3番目ごとにプリントされます。
引数 *frequency* は1から255までの範囲内である必要があります。引数 *frequency* が0 だと、デフォルト値の1 (すべてのラベルを印刷) が使用されます。
指定した軸が数値軸の場合、引数 *frequency* は、何も行いません。また、引数 *frequency* は、極線グラフや円グラフに対しては使用されません。最後のラベルは通常印刷されます。
引数 *frequency* に-1が設定された場合は4D Chartは自動的にグラフの次元により頻度の表示ラベルを設定します。

Note: 軸ラベルのテキスト属性を取得するには、*CT SET CHART TEXT ATTRIBUTES* コマンドを使用してください。

例題

以下の例は、$ChartIDに指定されるチャートの項目軸ラベル属性を変更します。ラベル位置は下、フォーマットは通常、方向は右回転に設定されます。

```
CT SET LABEL ATTRIBUTES(Area;$ChartID;0;3;2;"")
```

CT SET LEGEND ATTRIBUTES

CT SET LEGEND ATTRIBUTES (area ; object ; display ; orientation ; reverseOrder ; reverseKey ; location ; horizOffset ; vertOffset)

引数	型		説明
area	倍長整数	➡	4D Chart エリア
object	倍長整数	➡	オブジェクトID
display	整数	➡	凡例を表示するか? 0 = 表示しない 1 = 表示する
orientation	整数	➡	凡例内の系列の方向 0 = 水平 1 = 垂直
reverseOrder	整数	➡	順番を逆にするか? -1 = 変更しない 0 = 逆順にしない 1 = 逆順にする
reverseKey	整数	➡	順番を逆にするか? -1 = 変更しない 0 = 逆順にしない 1 = 逆順にする
location	整数	➡	位置コード
horizOffset	整数	➡	位置=0のときのプロットの左側からの 水平方向のオフセットをポイント単位
vertOffset	整数	➡	位置=0のときのプロットの上側からの 垂直方向のオフセットをポイント単位

説明

CT SET LEGEND ATTRIBUTES コマンドは、引数 *area*と *object* で指定された凡例の属性を設定します。

引数 *display* は、凡例が表示されるかどうかを指定します。

引数 *orientation* は、系列が凡例内で縦方向に表示されるか、横方向に表示されるのかを指定します。以下の図は、縦方向の凡例と横方向の凡例を示しています。

引数 *reverseOrder* は、凡例内の系列の順序が逆順になるかどうかを指定します。

引数 *reverseKey* は、系列の一意なパターンと色を説明する系列ラベルとキーが逆になるかどうかを指定します。デフォルトでは、キーがラベルの左側に表示されます。

以下の表は、引数 *location* のコードを示しています。

コード	位置
-1	変更なし
0	*horizOffset* と *vertOffset* で凡例を配置する
1	左上
2	左下
3	右上
4	右下
5	左
6	右
7	上
8	下

location が0に設定されていると、引数 *horizOffset* と *vertOffset* が使用されます。*horizOffset* はグラフの左側から凡例の左

側までをポイント単位で表します。*vertOffset* はグラフの上の端から凡例の上の端までをポイント単位で表します。

Note: 凡例テキストのテキスト属性を取得するには、*CT GET CHART TEXT ATTRIBUTES* コマンドを使用してください。

例題

以下の例は、グラフの上端中央に凡例を表示します。

```
CT SET LEGEND ATTRIBUTES(Area;$ChartID;1;0;0;0;7;0;0)
```

CT SET LEGEND TEXT

CT SET LEGEND TEXT (area ; object ; legendItem ; legendtext)

引数	型		説明
area	倍長整数	➡	4D Chart エリア
object	倍長整数	➡	オブジェクトID
legendItem	整数	➡	凡例項目番号
legendtext	テキスト	➡	凡例項目のテキスト

説明

CT SET LEGEND TEXT コマンドは、指定された凡例項目にテキストを設定します。

引数 *legendItem* は、凡例内での系列の番号 (あるいは、円グラフでの項目) です。ただし、凡例が逆順になった場合には、*legendItem* は逆順ではなく正順を反映します。

例題

以下の例は、$ChartIDに指定されたチャートの凡例テキストを変更します。

```
ARRAY STRING(20;aLegend;3)
aLegend{1}:="Sales"
aLegend{2}:="Marketing"
aLegend{3}:="Engineering"

For($i;1;3)
   CT SET LEGEND TEXT(vArea;$ChartID;$i;aLegend{$i})
End for
```

CT SET REAL SCALE

CT SET REAL SCALE (area ; object ; minAuto ; maxAuto ; majIncrAuto ; minIncrAuto ; minimum ; maximum ; majorIncr ; minorIncr)

引数	型		説明
area	倍長整数	➡	4D Chart エリア
object	倍長整数	➡	オブジェクトID
minAuto	整数	➡	デフォルトの最小値を使用しているか? -1 = 変更なし 0 = 使用しない 1 = 使用する
maxAuto	整数	➡	デフォルトの最大値を使用しているか? -1 = 変更なし 0 = 使用しない 1 = 使用する
majIncrAuto	整数	➡	デフォルトの主増分を使用しているか? -1 = 変更なし 0 = 使用しない 1 = 使用する
minIncrAuto	整数	➡	補助増分を使用しているか? -1 = 変更なし 0 = 使用しない 1 = 使用する
minimum	実数	➡	最小値
maximum	実数	➡	最大値
majorIncr	実数	➡	主増分
minorIncr	実数	➡	補助増分

説明

CT SET REAL SCALE コマンドは、デフォルト値が使用されるのかどうか、数値軸目盛用に設定されている補助値が何であるのかを設定します。*CT SET REAL SCALE* コマンドは、値が実数、整数、または倍長整数のときに使用されます。

引数 *minAuto* と *maxAuto* は、グラフがデフォルトの最小値と最大値を現在使用しているかどうかを示します。

引数 *majIncrAuto* と *minIncrAuto* は、グラフがデフォルトの主増分と補助増分を現在使用しているかどうかを示します。

引数 *minimum* と *maximum* は、代替最小値と代替最大値です。

引数 *majorIncr* と *minorIncr* は、代替主増分と代替補助増分です。

例題

この例題は配列からチャートを作成し、目盛りの値を設定します。

```
$ChartID:=CT Chart arrays(Area;2;1;aCategories;aSeries;aValues)
CT SET REAL SCALE(Area;$ChartID;0;0;0;0;-100;300;100;20)
```

CT SET TIPS ATTRIBUTES

CT SET TIPS ATTRIBUTES (area ; object ; axis ; toolBar ; status ; contents ; format ; formatX ; method)

引数	型		説明
area	倍長整数	⇒	4D Chart エリア
object	倍長整数	⇒	オブジェクトID
axis	整数	⇒	選択した軸 1 = 項目 2 = 系列 4 = 値
toolBar	整数	⇒	廃止された引数 (0を渡す)
status	整数	⇒	モードを表示 0 = ヒントなし 1 = 常にヒントを表示 2 = 要求時にのみヒントを表示
contents	整数	⇒	ヒントの内容 0 = 値のみ 1 = パーセンテージのみ 2 = 値とパーセンテージ
format	文字	⇒	Ｚ軸の値のフォーマットを表示
formatX	文字	⇒	Ｘ軸の値のフォーマットを表示
method	文字	⇒	実行するメソッドの名前

説明

CT SET TIPS ATTRIBUTES コマンドは、引数 *axis*、*toolBar*、*status*、*contents*、*format*、*formatX*、*method* に対して、*area* と *object* で指定したヒントの属性を修正します。

引数 *axis* はヒントが利用されている軸を示します。これは追加したいくつかの軸の数からの結果の数で構成しています。軸の数は：

数値	軸
1	項目軸
2	系列軸
4	値軸

*toolBar*引数は使用されません。この引数には0を渡してください。

引数 *status* はヒントの設定の表示を示します。ヒントは常に表示されているか、要求時に表示するか (Windows版では**Ctrl**、Macintosh版では**command**キーが押された時) 、または実行しないようにできます。

引数 *contents* でタイプ情報が表示されているか確認することができます。情報は値、パーセンテージまたは両方のいずれかです。

引数 *format* はＺ軸のヒントの値のフォーマットを表示します。フォーマットが "通常" の場合は、空の文字列 ("") が *format* に返されます。フォーマット表示に使用する特殊文字に関する詳細は、*4D Design Reference*マニュアルを参照してください。

引数 *formatX* は、Ｘ軸 (XYグラフのみ) に適用されることを除外すれば、*format* と同じです。

引数 *method* は、ヒントが表示されるのと同時に実行されているメソッドの名前です。この引数は、4つの引数を受け入れます。メソッドが空の文字列の場合は、ヒントが表示された時メソッドは実行されません。4D Chartがメソッドを呼び出した時、エラーを処理するために使用される4つの引数 ($1、$2、$3、$4) が渡されます。

引数	型	説明
$1	倍長整数	このメソッドが実行されている4D Chartエリアを表します
$2	倍長整数	グラフのIDを含んでいます
$3	倍長整数	X番目の項目の要素
$4	倍長整数	X番目の項目の要素

カーソルがグラフ要素の上に配置されていない場合は、引数$3と$4はゼロに設定されます。

データベースをコンパイルする場合は、これらの引数を以下のように定義してください。

```
C_LONGINT($1;$2;$3;$4)
```

CT SET TITLE ATTRIBUTES

CT SET TITLE ATTRIBUTES (area ; object ; axis ; position ; orientation ; title)

引数	型		説明
area	倍長整数	⇒	4D Chart エリア
object	倍長整数	⇒	オブジェクトID
axis	整数	⇒	属性を設定する軸 0 = 項目 1 = 系列 2 = 数値
position	整数	⇒	タイトルの位置 -1 = 変更なし 0 = なし 1 = 上 2 = 左 3 = 下 4 = 右
orientation	整数	⇒	タイトルの方向 -1 = 変更なし 0 = 通常 1 = 垂直 2 = 右回転 3 = 左回転
title	文字	⇒	タイトルのテキスト

説明

CT SET TITLE ATTRIBUTES コマンドは、引数 *area*、*object*、*axis* で指定された軸タイトルの位置、方向、テキストを設定します。

引数 *position* は、グラフに相対的なタイトルの位置です。

引数 *orientation* は、タイトルの方向です。

引数 *title* は、タイトルのテキストです。タイトルの長さは255文字以内です。

Note: タイトルのテキスト属性を取得するには、*CT SET CHART TEXT ATTRIBUTES* コマンドを使用してください。

例題

以下の例は、数値軸タイトルのテキスト、位置、方向を設定します。

```
CT SET TITLE ATTRIBUTES(Area;$ChartID;2;2;3;"Total Rainfall (in inches)")
```

CT SET VALUE ATTRIBUTES

CT SET VALUE ATTRIBUTES (area ; object ; position ; display ; orientation ; format)

引数	型		説明
area	倍長整数	⇒	4D Chart エリア
object	倍長整数	⇒	オブジェクトID
position	整数	⇒	値の位置 -1 = 変更なし 0 = なし 1 = 外部の上 2 = 外部の下 3 = 内部の上 4 = 内部の中央 5 = 内部の下 6 = 軸の上 8 = 左の上 9 = 右の上 10 = 下
display	整数	⇒	表示する情報のタイプ 1 = 値 2 = パーセンテージ 3 = カテゴリ 4 = 値とパーセンテージ 5 = カテゴリとパーセンテージ
orientation	整数	⇒	方向の値 -1 = 変更なし 0 = 標準 1 = 垂直 2 = 右の上 3 = 左の上
format	文字	⇒	値のフォーマット

説明

CT SET VALUE ATTRIBUTES コマンドは *area* や *object* で指定したグラフの値の属性を修正します。

position は位置の値を指定した整数です。

display は値として表示している情報のタイプを指定した整数です。

orientation は方向の値を指定した整数です。

CT SET X DATE SCALE

CT SET X DATE SCALE (area ; object ; minAuto ; maxAuto ; majIncrAuto ; minIncrAuto ; minimum ; maximum ; majIncrType ; majorIncr ; minIncrType ; minorIncr)

引数	型		説明
area	倍長整数	➡	4D Chart エリア
object	倍長整数	➡	オブジェクトID
minAuto	整数	➡	デフォルトの最小値を使用しているか? -1 = 変更なし 0 = 使用しない 1 = 使用する
maxAuto	整数	➡	デフォルトの最小値を使用しているか? -1 = 変更なし 0 = 使用しない 1 = 使用する
majIncrAuto	整数	➡	デフォルトの主増分を使用しているか? -1 = 変更なし 0 = 使用しない 1 = 使用する
minIncrAuto	整数	➡	デフォルトの補助増分を使用しているか? -1 =変更なし 0 = 使用しない 1 = 使用する
minimum	日付	➡	最小値
maximum	日付	➡	最大値
majIncrType	整数	➡	-1 = 変更なし 1 = 日 2 = 週 3 = 月 4 = 年
majorIncr	整数	➡	主増分
minIncrType	整数	➡	-1 = 変更なし 1 = 日 2 = 週 3 = 月 4 = 年
minorIncr	整数	➡	補助増分

説明

CT SET X DATE SCALE コマンドは、デフォルト値が使用されるのかどうか、XYチャートのX軸目盛り用の代替値の設定に使用します。同じチャートタイプの値軸 (Z軸) での *CT SET DATE SCALE* コマンドは、値が日付のときに使われます。

引数 *minAuto* と *maxAuto* は、デフォルトの最小値と最大値を使用するかどうかを示します。

引数 *majIncrAuto* と *minIncrAuto* は、デフォルトの最小値と最大値を使用するかどうかを示します。

引数 *minimum* と *maximum* は、代替最小値と代替最大値です。

引数 *majIncrType* と *minIncrType* は、*majorIncr* と *minorIncr* が指定される単位用のコードです。

引数 *majorIncr* と *minorIncr* は、代替主増分と代替補助増分です。

CT SET X REAL SCALE

CT SET X REAL SCALE (area ; object ; minAuto ; maxAuto ; majIncrAuto ; minIncrAuto ; minimum ; maximum ; majorIncr ; minorIncr)

引数	型		説明
area	倍長整数	⇒	4D Chart エリア
object	倍長整数	⇒	オブジェクトID
minAuto	整数	⇒	デフォルトの最小値を使用しているか? -1 = 変更なし 0 = 使用しない 1 = 使用する
maxAuto	整数	⇒	デフォルトの最大値を使用しているか? -1 = 変更なし 0 = 使用しない 1 = 使用する
majIncrAuto	整数	⇒	デフォルトの主増分を使用しているか? -1 = 変更なし 0 = 使用しない 1 = 使用する
minIncrAuto	整数	⇒	デフォルトの補助増分を使用しているか? -1 = 変更なし 0 = 使用しない 1 = 使用する
minimum	実数	⇒	最小値
maximum	実数	⇒	最大値
majorIncr	実数	⇒	主増分
minorIncr	実数	⇒	補助増分

説明

CT SET X REAL SCALE コマンドは、デフォルト値が使用されるのか、XYチャートでX軸目盛用に設定されている補助値が何であるのかを設定します。グラフタイプと同じ軸の値 (Z軸) では *CT SET REAL SCALE* コマンドは、値が実数、整数、または倍長整数のときに使用されます。

引数 *minAuto* と *maxAuto* は、グラフがデフォルトの最小値と最大値を現在使用しているかどうかを示します。

引数 *majIncrAuto* と *minIncrAuto* は、グラフがデフォルトの主増分と補助増分を現在使用しているかどうかを示します。

引数 *minimum* と *maximum* は、代替最小値と代替最大値です。

引数 *majorIncr* と *minorIncr* は、代替主増分と代替補助増分です。

例題

以下の例は、配列からチャートを生成し、目盛り値を設定します。

```
$ChartID:=CT Chart arrays(Area;2;1;aCategories;aSeries;aValues)
CT SET X REAL SCALE(Area;$ChartID;0;0;0;0;-100;300;100;20)
```

CT SHOW GRID LINES

CT SHOW GRID LINES (area ; object ; axis ; grid ; visible)

引数	型		説明
area	倍長整数	➡	4D Chart エリア
object	倍長整数	➡	オブジェクトID
axis	整数	➡	目盛線の表示/非表示に対する軸 0 = 項目 1 = 系列 2 = 数値
grid	整数	➡	表示/非表示対象のグリッド線 0 = 補助 1 = 主
visible	整数	➡	グリッド線の表示/非表示 0 = 非表示 1 = 表示

説明

CT SHOW GRID LINES コマンドは、引数 *area*、*object*、*axis* で指定される軸に対する主グリッド線または補助グリッド線あるいはその両方を表示または非表示にします。

引数 *grid* によって、コマンドによってどちらのグリッド線が影響を受けるのかを指定できます。主グリッド線は主増分の間隔になり、補助グリッド線は補助増分の間隔になります。

引数 *visible* によって、指定したグリッド線を見えるようにするかどうかを指定できます。

例題

以下の例は、$ChartIDに指定されたチャート用の数値軸の補助グリッド線を表示します。

```
CT SHOW GRID LINES(Area;$ChartID;2;0;1)
```

CT UPDATE CHART

CT UPDATE CHART (area ; object ; displayAlert)

引数	型		説明
area	倍長整数	➡	4D Chart エリア
object	倍長整数	➡	オブジェクトID
displayAlert	整数	➡	ユーザに警告を表示するか? 0 = 表示しない 1 = 表示する

説明

CT UPDATE CHART コマンドは、データベース内のデータから作成されたグラフを更新します。このコマンドは、チャートメニューの**更新**メニューと同じです。

このコマンドはデータベースのフィールドからユーザが作成したグラフ、または開発者が *CT Chart selection* 関数または *CT Chart data* 関数を使用して作成したグラフだけを更新します。

グラフは、グラフ化されるテーブルのカレントセレクション内のレコードを使用して更新されます。

引数 *displayAlert* が1である場合には、ユーザに警告ボックスが表示されるため、アクションの受け付けかキャンセルを選択できます。*displayAlert* が0の場合は、警告ボックスは表示されません。

例題

以下の例は、グラフ化されるテーブルのセレクションを変更し、変更を表示するようにグラフを更新します。

```
REDUCE SELECTION([Statistics];350)
CT UPDATE CHART(Area;$ChartID;0)
```

CTプリント

- CT PRINT
- CT PRINT MERGE

CT PRINT

CT PRINT (area ; cancellable ; printDialog)

引数	型		説明
area	倍長整数	⇒	4D Chart エリア
cancellable	整数	⇒	印刷のキャンセルを許可するか? 0 = キャンセルを許可しない 1 = キャンセルを許可する
printDialog	整数	⇒	プリントダイアログボックスを表示するか? 0 = ダイアログボックスを表示しない 1 = ダイアログボックスを表示する

説明

CT PRINT コマンドは、引数 *area* 内のドキュメントを印刷します。*CT PRINT*をコールすることは、プリント設定ダイアログボックスがユーザに提示されない以外は、**ファイル**メニューから**プリント**を選択することに対応する機能を持ちます。印刷前にプリント設定ダイアログボックスを表示するには、*CT DO COMMAND* を使用します。

引数 *cancellable* が1の場合には、4D Chartはユーザが**Ctrl** (Macintosh版では**command**) + **. (ピリオド)** キーを押すことによって印刷をキャンセルできるダイアログボックスを表示します。ユーザが印刷をキャンセルした場合、*CT Error* 関数はエラー番号20を返します。*cancellable* が0の場合には、このダイアログボックスは表示されず、ユーザは印刷をキャンセルできません。

オプション引数の *printDialog* が0の場合標準のプリントダイアログボックスは表示されず、印刷ジョブが即座に開始されます。*printDialog* が1である場合には、標準のプリントダイアログボックスが表示されます。

CT PRINT MERGE

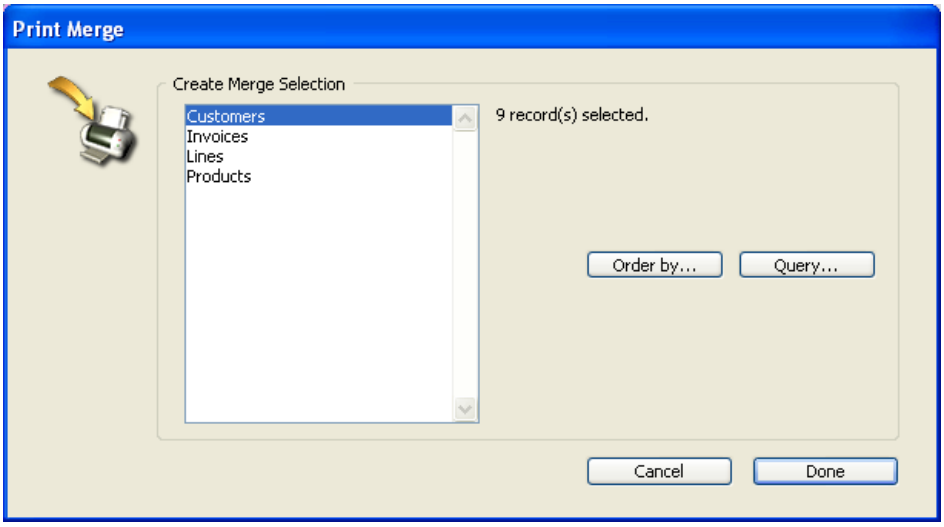
CT PRINT MERGE (area ; numTable ; cancellable ; printDialog)

引数	型		説明
area	倍長整数	➡	4D Chart エリア
numTable	整数	➡	テーブル番号
cancellable	整数	➡	印刷のキャンセルを許可するか? 0 = キャンセルを許可しない 1 =キャンセルを許可する
printDialog	整数	➡	プリントダイアログボックスを表示するか? 0 = ダイアログボックスを表示しない 1 = ダイアログボックスを表示する

説明

CT PRINT MERGE コマンドで、*fileNum* ファイルのカレントセレクション用のマージプリントを実行できます。マージプリント用のドキュメントは、*area*で指定されます。

table が0の場合には、マージセレクション作成ダイアログボックスが表示されます。



引数 *cancellable* が1の場合には、4D Chartはユーザが**Ctrl** (Macintosh版では**command**) + **. (ピリオド)** キーを押すことによって印刷をキャンセルできるダイアログボックスを表示します。ユーザが印刷をキャンセルした場合には、*CT Error* 関数はエラー番号20を返します。*cancellable* が0の場合には、このダイアログボックスは表示されず、ユーザは印刷をキャンセルできません。

printDialog は、プリントダイアログボックスを表示するかしないかを指定します。オプション引数の *printDialog* が0の場合は、標準のプリントダイアログボックスは表示されず、印刷ジョブが即座に開始されます。 *printDialog* が1である場合には、標準のプリントダイアログボックスが表示されます。

例題

以下は、4D Chart エリアが**値表示**モードに設定され、マージプリントを実行することを確認する例です:

```
  `プリントマージ用の選択を生成する
ALL RECORDS([MyFile])
  `表示モードが値表示に設定されいることを確認する
CT MENU STATUS(Area;6006;$Checked;$Available;$Name)

If($Name="Show Values")
```

```
    CT DO COMMAND(Area;6006) `表示を値表示モードに設定する
End if

`ユーザの操作なしにマージプリントを実行する
CT PRINT MERGE(Area;File(->[MyFile]);0;0)
```

CTユーティリティ

- 4D Chartコマンドでカラーを指定する
- CT Clipboard to picture
- CT Color to index
- CT COLOR TO RGB
- CT Font name
- CT Font number
- CT Index to color
- CT PICTURE TO CLIPBOARD
- CT RGB to color

4D Chartコマンドでカラーを指定する

4D Chartでカラーを指定するには、以下の3種類の方法があります。

- **RGB**：赤、緑、青のカラー (3つの倍長整数) 。
- **インデックス**： 4Dパレットの1から256までのカラーで、モニタが256色に設定されていれば**塗りつぶしカラー**、**線カラー**、**テキストカラー**の各サブメニューで表示されます (1つの倍長整数) 。
- **カラー**：4D Chart内部で使用される値 (1つの倍長整数) 。

この節で説明するユーティリティコマンドは、これらの3種類の表記の間でカラーの値を変換します。**CTチャート**と**CTオブジェクト**テーマに記載されているコマンドは、倍長整数タイプの**カラー**を引数として想定しています。

CT Clipboard to picture

CT Clipboard to picture -> 戻り値

引数	型		説明
戻り値	ピクチャー		クリップボードの内容のコピー

説明

*CT Clipboard to picture*は、クリップボードの内容のコピーである4Dピクチャを返します。

クリップボードにピクチャがない場合には、*CT Clipboard to picture*は空のピクチャを返します。

例題

以下の例は、クリップボードからvPict変数にピクチャをコピーします。

```
vPict:=CT Clipboard to picture
```

CT Color to index

CT Color to index (color) -> 戻り値

引数	型		説明
color	倍長整数	➡	カラーの値 (4D Chart内部の値)
戻り値	整数	➲	4Dパレットの最も近い色のインデックス

説明

CT Color to index 関数は、 引数 *color* に最も近い4Dカラーパレットの色のインデックスを返します。

例えば、特定の青の影が指定された場合に、*CT Color to index* 関数は4Dパレットに最も近い青の影を返します。4Dパレット上の色は1から256までの番号が付けられます。

例題

以下の例は、赤に最も近い色のパレットインデックスをvColor変数に配置します:

```
vColor:=CT Color to index(CT RGB to color(56683;2242;1698))
```

CT COLOR TO RGB

CT COLOR TO RGB (color ; red ; green ; blue)

引数	型		説明
color	倍長整数	⇒	カラーの値 (4D Chart内部の値)
red	倍長整数	⇐	赤の値を受け取る (0から65535まで)
green	倍長整数	⇐	緑の値を受け取る (0から65535まで)
blue	倍長整数	⇐	青の値を受け取る (0から65535まで)

説明

CT COLOR TO RGB コマンドは、引数 *color* にあるカラーの値を分解して、*red*、*green*、*blue* 変数にコンポーネントを表す値を返します。

例題

以下の例は、カラーの値用のコンポーネントRGB値を取得し、それらをいくつかのローカル変数に配置します:

```
$color:=100000
CT COLOR TO RGB($color;$red;$green;$blue)
```

CT Font name

CT Font name (fontNumber) -> 戻り値

引数	型		説明
fontNumber	整数	⇒	フォントID番号
戻り値	文字	⇐	フォントの名前

説明

CT Font name 関数は、IDが *fontNumber* であるフォントの名前を返します。

fontNumber に対応するフォントが存在しない場合、*CT Font name* 関数は空の文字列を返します。

例題

以下の例は、IDが3のフォントの名前を返します:

```
vName:=CT Font name(3)
```

CT Font number

CT Font number (fontName) -> 戻り値

引数	型		説明
fontName	文字	➡	フォントの名前
戻り値	整数	➲	フォントのID番号

説明

CT Font number 関数は、名前が *fontName* であるフォントのIDを返します。

例題

以下の例は、フォント "Times" の数値を返します:

```
vNumber:=CT Font number("Times")
```

CT Index to color

CT Index to color (index) -> 戻り値

引数	型		説明
index	整数	➡	パレットインデックス
戻り値	倍長整数	⮌	インデックスで指定された色

説明

CT Index to color 関数は、引数 *index* で記述されたカラーを返します。

Index は4Dカラーパレットで特定の色を指定する整数です。4Dパレット上のカラーは1から256までの番号がついています。

CT Index to color 関数は、個別のコンポーネントを知らなくても色を指定できる便利な方法です。

例題

以下の例は、vColor変数にシアンの色 (パレットインデックス番号8) の色の値を代入します:

```
vColor:=CT Index to color(8)
```

CT PICTURE TO CLIPBOARD

CT PICTURE TO CLIPBOARD (picture)

引数	型		説明
picture	ピクチャー	➡	ピクチャ

説明

CT PICTURE TO CLIPBOARD コマンドは、引数 *picture* をクリップボードにコピーします。

クリップボード上にコピーされると、ピクチャがペーストできるところならどこにでも *picture* を貼り付けることができます。

例題

以下の例は[Chart]Objectフィールドの内容をクリップボードにコピーします:

```
CT PICTURE TO CLIPBOARD([Chart]Object)
```

CT RGB to color

CT RGB to color (red ; green ; blue) -> 戻り値

引数	型		説明
red	倍長整数	➡	赤の値 (0から65535まで)
green	倍長整数	➡	緑の値 (0から65535まで)
blue	倍長整数	➡	青の値 (0から65535まで)
戻り値	倍長整数	↩	カラーの値 (4D Chart内部)

説明

CT RGB to color 関数は、引数 *red*、*green*、*blue* のコンポーネントを表す値を返します。この数値は複数の4D Chartコマンドで使用されます。

red、*green*、*blue*は、Macintoshのカラーピッカーで使用されているものと同じです。

以下の表は、一般的に使用される3つのカラーでの*red*、*green*、*blue*の値を示しています。

カラー	red	green	blue
赤	56683	2242	1698
緑	0	32768	4528
青	0	0	54272

例題

以下の例は$color変数に赤色を設定します:

```
$color:=CT RGB to color(56683;2242;1698)
```

制御コード

コマンドコード

次の表はcommand引数に使用されるコードの一覧です。

メニュー	コマンド番号	メニュー項目
ツールパレット	1	ポインタ
	2	テキスト
	3	線
	4	矩形
	5	角の丸い
	6	楕円
	7	多角形
ファイル	1001	新規
	1002	開く
	1003	保存
	1004	新規保存
	1006	テンプレートとして保存
	1008	用紙設定
	1009	プリント
	1010	プリントマージ
	1012	フルウインドウ / フォームに戻る
	1013	データ読み込み
	1014	セレクションデータを書き出し
	1015	データを書き出し
編集	2001	取り消し
	2003	切り取り
	2004	コピー
	2005	貼り付け
	2006	消去
	2007	複製
	2009	すべてを選択
	2011	プロパティ
	4016	表示メニュー
テキスト	3001	フォントメニュー
	3002	サイズメニュー
	3003	書体メニュー
	3004	カラーメニュー
	3005	文字揃えメニュー
チャート	4002	新規チャート
	4003	軸メニュー
	4004	目盛線メニュー
	4005	タイトルメニュー
	4007	凡例
	4012	ビュー
	4009	数値
	4017	ヒント
	4010	オプション
	4014	更新
オブジェクト	5001	塗りつぶしパターンメニュー
	5002	塗りつぶしカラーンメニュー
	5004	線パターンメニュー
	5005	線カラーメニュー

	5007	線幅メニュー
	5008	矢印形態メニュー
	5010	角丸めメニュー
	5012	調整メニュー
データベース	6001	フィールド貼り付け
	6002	フォーマット
	6003	参照
	6004	参照解除
	6006	値表示 / 参照表示
フォント	7001 - 7999	個々のフォント名
サイズ	8001 - 8009	個々のフォントサイズ
	8010	その他のサイズ
書体	9001	標準
	9002	太字
	9003	斜体
	9004	下線
	9005	アウトライン
	9006	シャドウ
カラー	10001 - 10999	個々のカラー
文字揃え	11001	左
	11002	中央
	11003	右
新規2Dチャート	12001	2D 面
	12002	2D 棒
	12003	2D 線
	12005	2D 円
	12006	2D ピクチャ
	12007	2D ポーラー
	12008	2D XY
新規3Dチャート	13001	3D 棒
	13002	3D 線
	13003	3D 面
	13004	3D 等高線
	13005	3D 三角形
	13006	3D ピン
軸	14001	項目 (X軸)
	14002	数値 (Z軸) (2Dグラフ)
		系列 (Y軸) (3Dグラフ)
	14003	数値 (Z軸) (3Dグラフ)
目盛線	15001	項目 (X軸)
	15002	数値 (Z軸) (2Dグラフ)
		系列 (Y軸) (3Dグラフ)
	15003	数値 (Z軸) (3Dグラフ)
タイトル	16001	項目 (X軸)
	16002	数値 (Z軸) (2Dグラフ)
		系列 (Y軸) (3Dグラフ)
	16003	数値 (Z軸) (3Dグラフ)
表示	17001	メニューバー

	17002	オブジェクトツール
	17003	チャートツール
	17004	スクロールバー
	17005	ルーラ
塗りつぶしパターン	18001 - 18036	個々の塗りつぶしパターン
塗りつぶしカラー	19001 - 19256	個々の塗りつぶしカラー
線パターン	20001 - 20036	個々の線パターン
線カラー	21001 - 21256	個々の線カラー
線幅	22001	0.25ポイント
	22002	1ポイント
	22003	2ポイント
	22004	4ポイント
	22005	6ポイント
	22006	その他の線幅
矢印	23001	形態なし
	23002	始点
	23003	終点
	23004	両端
調整	24001	最前面へ
	24002	最背面へ
	24003	前面へ
	24004	背面へ
	24006	オブジェクトの整列
	24008	グループ化
	24009	グループ解除

4D Chart エラーコード

次の表は、4D Chart エラーメッセージのコードです。

エラー	メッセージ
1	4D Chartのエリア参照が正しくありません。
2	セグメントをロードすることができません。
3	コマンド番号が正しくありません。
4	このメニュー項目を使用できません。
5	ホットリンクが見つかりません。
6	4Dテーブル番号が正しくありません。
7	4Dフィールド番号が正しくありません。
8	ピクチャを作成できませんでした。
9	*CT AREA TO AREA* コマンドのスコープが正しくありません。
10	送信元と送信先エリアは異ならなければなりません。
11	スコープが正しくありません。
12	オブジェクトIDが正しくありません。
13	この操作はこのオブジェクトタイプで実行することはできません。
14	オブジェクトインデックスが正しくありません。
15	選択されたオブジェクトがありません。
16	このドキュメント内にオブジェクトがありません。
17	ファイルタイプが正しくありません。
18	この4D Chartドキュメントのバージョンでは、サポートされていません。
19	この4D Chartドキュメントは最新バージョンで作成されています。
20	ユーザがダイアログボックスをキャンセルしました。
21	ホットリンクのタイプが正しくありません。
22	このホットリンクへの追加はホットリンク連鎖で反復して生成されます。
23	この操作はオブジェクトをドキュメント外に移動しました。
24	4Dテーブルがありません。
25	ビットマップイメージが大きすぎます。
26	この操作はオブジェクトの最大番号を越えました。
27	チャートを作成するための項目数が十分ではありません。
28	チャートを作成するには項目数が多すぎます。
29	チャートを作成するための系列数が十分ではありません。
30	チャートを作成するには系列数が多すぎます。
31	チャートタイプが正しくありません。
32	チャートサイズが正しくありません。
33	項目軸に設定するフィールドタイプが正しくありません。
34	数値軸に設定するフィールドタイプが正しくありません。
35	配列タイプが正しくありません。
36	値が複数あります。
37	そのホットリンク名またはホットリンクタイプはすでに存在します。
38	少なくとも1文字必要です。
39	オブジェクト境界が正しくありません。
40	角の丸みが正しくありません。
41	RGB値が正しくありません。
42	カラー値が正しくありません。
43	カラーインデックスが正しくありません。
44	クリップボードにピクチャがありません。
45	すべての値が範囲外です。
46	テキスト編集モードではありません。
47	軸インデックスが正しくありません。

48	目盛線インデックスが正しくありません。
49	多角形の頂点数が正しくありません。
50	座標が正しくありません。
51	オブジェクトサイズが正しくありません。
52	指定されたオブジェクトにはこの属性はありません。
53	フルパス名が255バイトを越えました。
54	フィールドタイプが正しくありません。
55	この種のチャートには適用できません。
56	クリップボードからの値が正しくありません。
57	クリップボードからの次元数が正しくありません。
58	セレクション値が正しくありません。
59	プリンタリソースを初期化できませんでした。
60	オフスクリーンエリアへのメモリ割り当てが十分ではありません。
61	カラーリソースで使用するメモリが十分ではありません。
62	表示項目番号が正しくありません。
63	テキストオブジェクトをコピーするメモリが十分ではありません。
64	フォントが正しくありません。
65	選択されたレコードがありません。
66	チャートのタイプの値が正しくありません。

引数コード

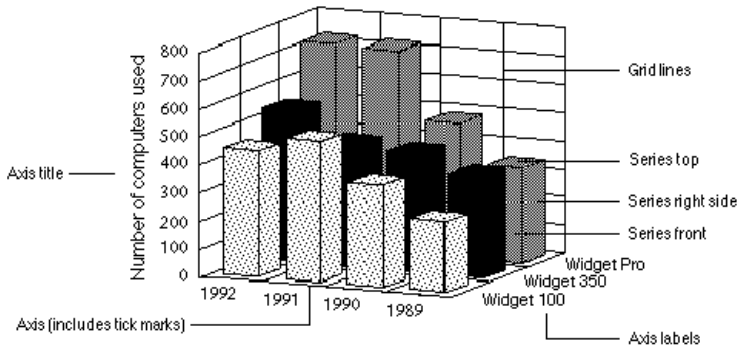
この付録では頻繁に使用される4D Chart の引数コードを掲載します。以下の引数が関連します:

- *partType*
- *partSpecifics*
- *pattern*
- *options*

引数タイプと特定部分のコード

グラフ軸のひとつまたは凡例などのグラフの一部をプログラムで修正する場合、引数 *partType* と *partSpecifics* を使って、修正するグラフの一部を指定します。

グラフの主要部分を次の図に示します。



- 下記は、引数 *partType* 用のコードです。

コード	グラフ要素
1	プロット矩形
2	凡例
3	軸
4	ラベル
5	タイトル
6	主目盛線
7	補助目盛線
8	系列
9	系列のラベル

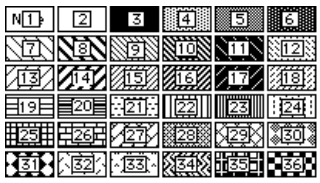
- 下記は引数 *partSpecifics* 用のコードです。

グラフ要素	特定部分コード
プロット矩形	2Dグラフ: 0 = 矩形全体
	3Dグラフ: 1 = 背面、 2 = 側面、 3 = 下
凡例	0 = 凡例矩形
軸	0 = 項目、 1 = 系列、 2 = 数値
ラベル	0 = 項目、 1 = 系列、 2 = 数値
タイトル	0 = 項目、 1 = 系列、 2 = 数値
主目盛線	0 = 項目、 1 = 系列、 2 = 数値
補助目盛線	0 = 項目、 1 = 系列、 2 = 数値
系列 / 系列のラベル	(系列番号*100) + (側面番号)
側面番号 (系列側面用)	0 = すべての側面 (すべての2Dグラフに使用)、1 = 前面、 2 = 左、 3 = 右、4 = 上、 5 = 下

Note: 2Dグラフの系列は、ひとつの側面 (前面) を持っています。3Dグラフでは、一度に3つの側面を表示することができます。

パターンコード

以下は引数 *pattern* のコードです。:



オプションコード

下記は*CT GET CHART OPTIONS*と*CT SET CHART OPTIONS*で使用される引数 *options* 用のコードです。

グラフタイプ	要素番号	コード
2D 面	1	方向: 0 = 垂直、 1 = 水平
	2	積み重ね: 0 = 通常、 1 = 積み重ね、 2 = 比率
2D 棒	1	方向: 0 = 垂直、 1 = 水平
	2	積み重ね: 0 = 通常、 1 = 積み重ね、 2 = 比率
	3	重ねる比率 (0から100%)
	4	間隔比率 (0から100%)
2D 線	1	方向: 0 = 垂直、 1 = 水平
	2	積み重ね: 0 = 通常、 1 = 積み重ね、 2 = 比率
	3	区画の表示コントロール: 0 = 非表示、 1 = 表示
2D 散布図	1	方向: 0 = 垂直、 1 = 水平
	2	積み重ね: 0 = 通常、 1 = 積み重ね、 2 = 比率
2D 円	1	開始角度 (0から360度)
ピクチャ	1	方向: 0 = 垂直、 1 = 水平
	2	積み重ね: 0 = 通常、 1 = 積み重ね、 2 = 比率
	3	重ねる比率 (0から100%)
	4	間隔比率 (0から100%)
	5	水平整列: 0 = 左、 1 = 中央、 2 = 右
	6	垂直整列: 0 = 上、 1 = 中央、 2 = 下
	7	水平方向表示: 0 = 切り捨て、 1 = 拡大縮小、2 = 積み重ね
	8	垂直方向表示: 0 = 切り捨て、 1 = 拡大縮小、2 = 積み重ね
ポーラー		なし
XY チャート	1	ポイントフォーム: 0 = ポイント表示なし、 1 = 円、 2 = 四角、 3 = 星型
	2	線のタイプ: 0 = 線なし、 1 = 直線、 2 = 矢印線
	3	戻り線の表示: 0 = なし、 1 = あり
3D 棒	1	項目軸を比率で指定: 0 = 幅、 1 = 間隔
	2	系列軸を比率で指定: 0 = 幅、 1 = 間隔
	3	項目軸比率 (0から100%)
	4	系列軸比率 (0から100%)
	5	表示範囲: 0 = すべて、 1 = 上部のみ
3D 線	1	系列軸を比率で指定: 0 = 幅、 1 = 間隔
	2	系列軸比率 (0から100%)
3D 面	1	系列軸を比率で指定: 0 = 幅、 1 = 間隔
	2	系列軸比率 (0から100%)
3D 等高線	1	表示範囲: 0 = すべて、 1 = 上部のみ
3D 三角形	1	上下反転: 0 = なし、 1 = あり
	2	数値0も描画する: 0 = なし、 1 = あり
	3	系列軸を比率で指定: 0 = 幅、 1 = 間隔
	4	系列軸比率 (0から100%)
3D ピン	1	ピン先頭の形: 0 = 楕円形、 1 = 四角形

4D Chart - コマンドリスト (文字順)

- CT AREA TO FIELD
- CT Area to picture
- CT Array to polygon