4D Chart®

Language Reference
Windows and Mac OS Versions

4D Chart Language Reference
Version 6.0 for Windows® and Mac™ OS

Contents

5. CT Chart 129

1 Introduction

4D Chart is a plug-in that adds the capabilities of a graphing package to 4th Dimension. 4D Chart is included within 4th Dimension and is available in both the User and Custom Menus environments.

4D Chart adds more than 100 commands to the 4th Dimension language. 4D Chart commands allow you to control tasks that users normally perform manually.

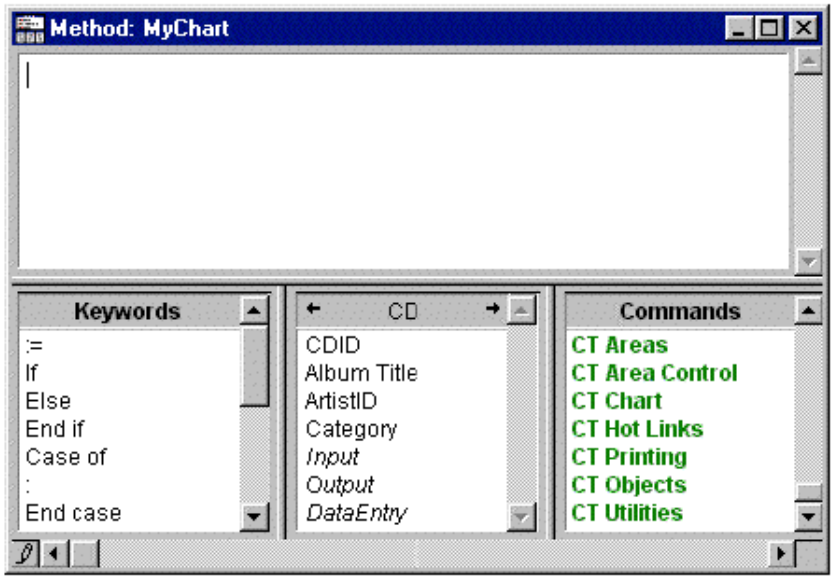
For example, you can open and save documents, create and modify graphs, execute any 4D Chart menu command, or work with hot links to other 4th Dimension plug-ins.

4D Chart Commands in the Method Editor

All the 4D Chart commands are preceded by the letters “CT” to distinguish them from the standard 4th Dimension commands and from the commands added by other plug-ins. They appear in methods like this:

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

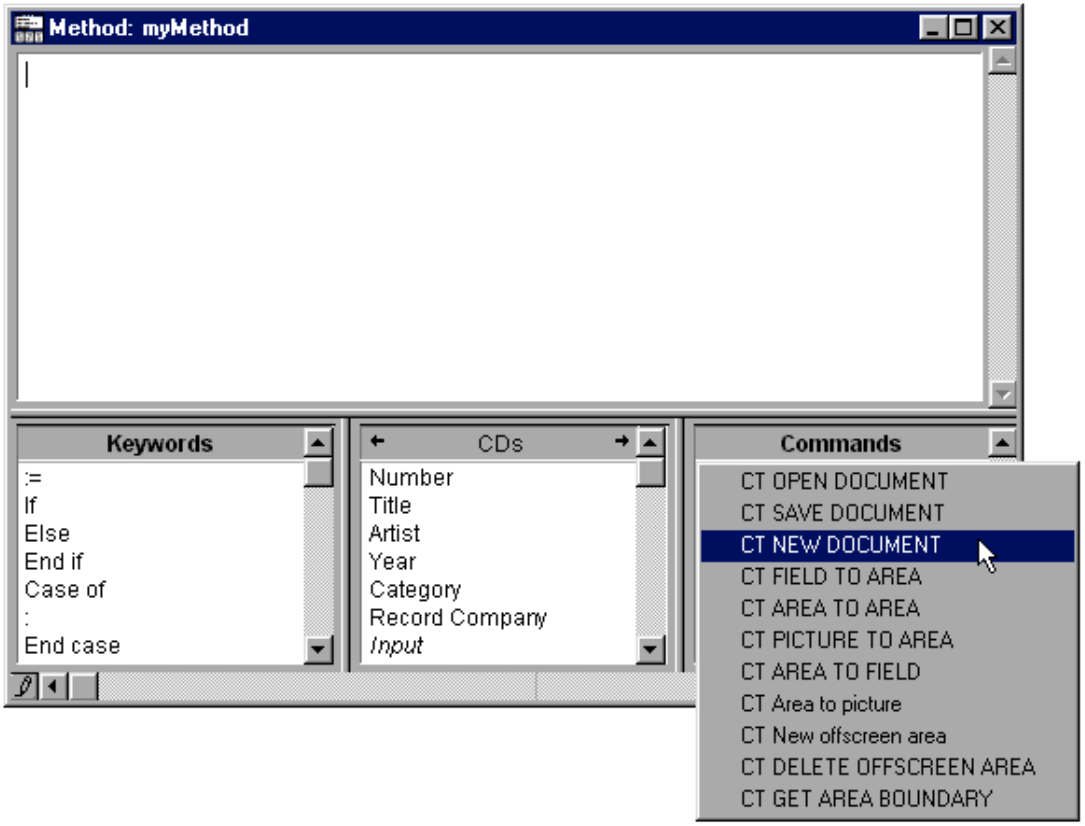
The 4D Chart commands are grouped into themes in the Method editor and appear at the end of the Routines list.



If you have more than one plug-in installed, the order in which the themes appear reflects the order in which the plug-ins were installed.

The 4D Chart commands remain grouped by theme, regardless of whether the built-in 4th Dimension commands are in alphabetical order or grouped by theme.

You place a 4D Chart command in a method by typing it into the method or by choosing it from a pop-up menu in the Routines list. Plug-in commands do not appear in the Commands page of the Explorer.



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

About this Manual

In this manual, the 4D Chart commands (which do not return values) are printed in all uppercase letters using a special font, for example: CT_OPEN DOCUMENT. 4D Chart functions (which do return values) are shown with an initial capital letter: CT Get chart type.

In addition, all table names are shown in brackets in the text to help distinguish them from the names of fields, forms and other items. For instance, the Companies table is written as the [Companies] table.

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

For more information about using the 4th Dimension language, please refer to the Introduction section of the *4th Dimension Language Reference* manual.

You can work with 4D Chart in the following locations:

- 4D Chart areas on forms,
- 4D Chart plug-in windows,
- 4D Chart offscreen areas.

This section describes how to create those locations in your databases.

4D Chart Areas on Forms

You can place 4D Chart in any form. Usually you place 4D Chart on an input form so that you can work with documents. You can place 4D Chart in an output form to display and print information as well. 4D Chart can use the entire form, or it can share space with fields and other form objects.

You use a Plug-in Area object to create a 4D Chart area on a form. A Plug-in Area object is one of several types of active objects in 4th Dimension, such as buttons, enterable areas, and scrollable areas. For complete information about Plug-in Area objects, see the *4th Dimension Design Reference* and the *4th Dimension Language Reference* manuals.

When you need to refer to a document on a form, use the object name that you used when you created the Plug-in Area object.

For more information about placing 4D Chart areas in a form, refer to the section Referring to 4D Chart Documents.

4D Chart Plug-in Windows

You use the Open external window function to open a plug-in window and display an empty 4D Chart document in it. For more information about this function, refer to the *4th Dimension Language Reference*.

Open external window opens a new window, displays the 4D plug-in area supported by the plugInArea parameter, and returns the ID number for the area.

Open external window creates modeless windows, allowing you to have several active windows open simultaneously. The command does not wait for user input, so you can have several active windows open at once. You can click between each window and edit the one in front. If the window type has a title bar, a Control-menu box (Windows) or a Close Box (Macintosh) will be added to enable the user to close the window.

To close a plug-in window programmatically, pass the variable returned by Open external window to the 4th Dimension command CLOSE WINDOW.

Example

The following is an example of the use of Open external window. This statement opens a plug-in window and displays an empty 4D Chart document.

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

Subsequently, you would use vChart whenever you need to specify the area for that document. For example:

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

4D Chart Offscreen Areas

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

4D Chart operations can be performed more quickly in an offscreen area because the screen does not have to be redrawn.

You can use the CT New offscreen area function to create an offscreen area. You can use the CT PICTURE TO AREA command to place a Picture field (which can contain a 4D Chart area) in a 4D Chart area (which can be an offscreen area). For a complete explanation of these commands, refer to the commands in the Area theme.

Remember to delete the offscreen area after you are done with it to free the memory it uses. 4th Dimension will display an error message when you close the database if you have not cleared all offscreen areas.

Example

When placed in a project method, the code in the following example creates an offscreen area for saving a document. Using a button on a form, you can allow the user to revert to the original saved document.

```
Area:=CT New offscreen area
QUERY ([Sales];[Sales]CustID=vCustID)
If (Records in selection ([Sales]=1)
   CT PICTURE TO AREA (Area;[Sales]Profits_)
      `Store the graph in the offscreen area
   MODIFY RECORD ([Sales])
      `Modify the Sales record
   CT DELETE OFFSCREEN AREA (Area)
      `Free the memory used by the offscreen area
End if
```

Create a button on the input form and assign it the following code:

```
Profits:=CT Area to picture (Area;-2)
   `Places the offscreen area that contains the original document into the plug-in
   `area contained in the Profits form.
```

See Also

No reference.

When you use commands to control a 4D Chart document, you need to identify the document by its area ID number. The area ID number is internal to 4D Chart and is normally stored in a variable.

A 4D Chart document can exist in any of three locations: in an area on a form, in a plug-in window, or in an offscreen area. Wherever the document is located, 4D Chart needs its area ID number to locate it and operate on it.

Area ID Numbers and the Area Variable

4D Chart uses variables to store the location of 4D Chart areas, plug-in windows, and offscreen areas. You reference the area on which you want to perform an operation by passing the variable containing the area's ID number as a parameter to the command or function.

In the command descriptions that follow this introduction, the area parameter refers to the variable identifying the document area.

There are two types of area variables:

- Plug-in area object names,
- Variables you create for a plug-in window or offscreen area.

Plug-in object names

When you create and name a 4D Chart area on a form, 4th Dimension automatically recognizes the name of the 4D Chart area as a variable referring to the area. For example, you would refer to the Profits area by specifying "Profits" as the area parameter.

Plug-in windows and offscreen areas

When you create a plug-in window or offscreen area using the Open external window or CT New offscreen area functions, the area ID number returned by the function should be stored in a variable. You can then use the variable to refer to the plug-in window or offscreen area in other commands and functions.

To store the value in a variable, you place the variable name and the assignment operator (:=) to the left of the function in the line of code.

The following example creates a 4D Chart plug-in window and stores the area ID number in the MyArea variable:

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

Using the 4D Chart Default Area

Introduction

version 6.0 (Modified)

The default area is a template in memory that can be used to set the default attributes of all new 4D Chart areas and plug-in windows. Any command that can be executed on a 4D Chart area can be executed on the default area by setting the area parameter to -1. You can use methods to perform operations on the default area as you would with any other area.

Using the default area, you can eliminate the unnecessary execution of code for 4D Chart areas. For example, if you want all new 4D Chart areas and plug-in windows to appear without scroll bars, you don't have to turn the scroll bars off for each area and plug-in window individually.

You can set the attributes of both 4D Chart areas and plug-in windows. The default area is used automatically as a template whenever a new 4D Chart area on a form or new plug-in window is opened. Since no code has to be executed, the default area provides a quick way to customize the chart area.

If you do not want the default area to apply to every new 4D Chart area, you can override it by creating a template on disk for the 4D Chart area or by executing appropriate code when the On Load event occurs. A template on disk or code that runs when the On Load event occurs takes precedence over the default area.

A 4D Chart document is composed of different objects such as graphs, axis labels, entered text, pictures, etc. 4D Chart enables you to handle those objects programmatically.

This section describes how to do the following programmatically:

- Refer to objects,
- Specify the coordinates of an object,
- Specify the scope of a command.

Referring to Objects

Every object in a 4D Chart document is given a unique number. This number is referred to as the object's ID and is assigned when the object is created.

This means that each time the user creates a graph, draws an object with the Tools palette, pastes an object from the Clipboard, groups several objects, duplicates an existing object, subscribes to a hot link, or pastes a field reference, a new ID is assigned. Since the object ID is unique, IDs are a convenient way to refer to objects. Object IDs are never reused within a document. Even if an object is deleted, its number is "retired" for the life of the document.

Object IDs are not transferable. An object whose ID is equal to 5 in one 4D Chart document will not necessarily have the same ID if it is pasted into another document.

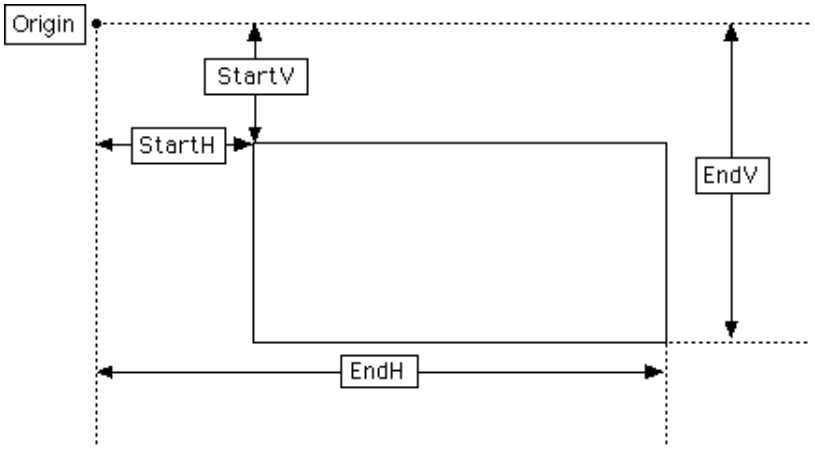
All of the standard object creation commands are functions and return the resulting object's ID. Unlike objects created by the user through the palette, objects created through the commands are not automatically selected.

You can obtain the ID of an object using the CT Get ID function.

Specifying the Coordinates of an Object

An object's position and size are referred to as its coordinates. All commands that describe or specify coordinates do so in points.

Commands that describe or specify a position do so with respect to the origin. The origin is the intersection of the zero points on the horizontal and vertical rulers. The following diagram illustrates the coordinate system.



Specifying the Scope of a Command

Many 4D Chart commands have a parameter called scope, which specifies which objects or text characters in a 4D Chart document are affected by the command.

The following table describes the general rules for scope. For a description of how scope affects a given command, see the description of that command.

Scope	Affected text or object
>0	Object ID
0	Selected objects
-1	All objects in the document
-2	Default values
-3	Selected characters in a text object

Chart Data Types

version 6.0 (Modified)

The following table lists the types of data that can be graphed with 4D Chart, the axes on which each data type can be graphed, and the data types that are compatible as multiple series on a single two-dimensional graph.

Data Type	Category or Series Axis?	Values Axis?	Compatible Types on Values Axis
Alphanumeric	Yes	No	-
Text	Yes	No	-
Numeric	Yes	Yes	Integer, Long integer
Integer	Yes	Yes	Numeric, Long integer
Long integer	Yes	Yes	Numeric, Integer
Date	Yes	Yes	-
Time	Yes	No	-
Boolean	Yes	No	-

Note: You cannot use Picture or BLOB fields in a graph.

4D Chart provides several ways to manage errors that occur during the execution of your code. You can use any combination of these techniques.

You can use the CT Error function to check for errors after performing an operation. CT Error returns an error code that represents the status of the last operation performed by 4D Chart. The section Error Codes lists all the 4D Chart error codes.

You can use CT ON ERROR to install a method for managing 4D Chart errors. After a method is installed with CT ON ERROR, 4D Chart calls that method when a 4D Chart error occurs.

You can check for errors after executing a function or a command that returns values in parameters. If an error occurs during the execution of a function, the function returns -32000. If a command that returns values in parameters encounters an error when retrieving a value for a particular parameter, it returns -32000 in that parameter.

See Also

No reference.

Creating Graphs from Records in a Database (examples) Introduction

version 6.0 (Modified)

This section provides examples for creating two- and three-dimensional graphs using the CT Chart selection and CT Chart data functions.

To learn about creating two- and three-dimensional graphs using arrays, refer to the section Creating Graphs from Arrays (examples).

In each example, you will find:

- A description of the situation used for the example,
- The structure of the example database,
- A graph that has been created entirely programmatically, using example data,
- The code used to create the example graph.

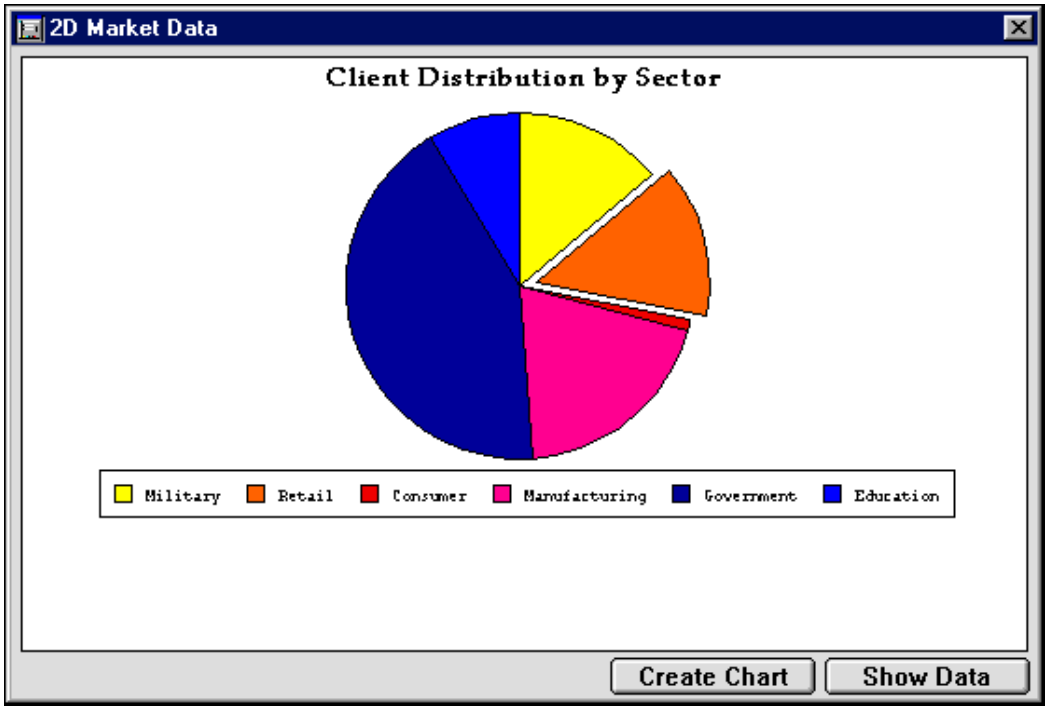
Creating a Two-Dimensional Graph With the Series Defined by Fields

Suppose you own a software company that has clients in a wide variety of market sectors, from government to education to retail. Your database tracks your customers and sales.

Your database uses the following table to store the information:

Market Data	
CustID	A
CustType	A
NumberOfUnits	I

Using 4D Chart, you produce the following pie chart. It illustrates the percentage of your total sales from each sector.



The following is the *GRAPH PROFILE* method, which was used to create the example graph.

```
`Method: GRAPH PROFILE
`Categories: Customer type
`Values: Number of units purchased

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

 `Generate the selection of records to chart
ALL RECORDS ([Market Data])

 `Array of fields for series data
ARRAY LONGINT ($aFields;1)
$aFields{1}:=Field(->[Market Data]NumberOfUnits)
```

```
CT SET DISPLAY (Area;3;0)  `Hide Object tools
CT SET DISPLAY (Area;6;0)  `Hide Scroll bars
CT SET DISPLAY (Area;9;0)  `Hide Rulers

 `Create a pie chart
$Chart:=CT Chart selection (Area;6;1;1;Table(->[Market Data]);2;$aFields)

 `Explode piece #3  by 10 degrees
CT EXPLODE PIE (Area;$Chart;2;10)

 `Set the legend position (bottom, horizontal orientation)
CT SET LEGEND ATTRIBUTES (Area;$Chart;1;0;0;0;8;0;0)

 `Add a title at the top left hand corner
$Title:=CT Draw text (Area;1;1;210;3;"Client Distribution by Sector")

 `Format title (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)

 `Get area dimensions for centering
CT GET AREA BOUNDARY (Area;1;$Left;$Top;$Right;$Bottom)

 `Resize the chart to window size minus 50 points
CT SIZE (Area;$Chart;$Right-50;$Bottom-50)

 `Center chart
CT GET BOUNDARY (Area;$Chart;$Left2;$Top2;$Right2;$Bottom2)
$Locate:=(($Right-$Left)-($Right2-$Left2))/2
CT MOVE (Area;$Chart;$Locate;$Top2)

 `Center title
CT GET BOUNDARY (Area;$Title;$Left2;$Top2;$Right2;$Bottom2)
$Locate:=(($Right-$Left)-($Right2-$Left2))/2
CT MOVE (Area;$Title;$Locate;$Top2)

 `Move the Chart 9 points down
CT GET BOUNDARY (Area;$Chart;$Left;$Top;$Right;$Bottom)
CT MOVE (Area;$Chart;$Left;$Top+9)
```

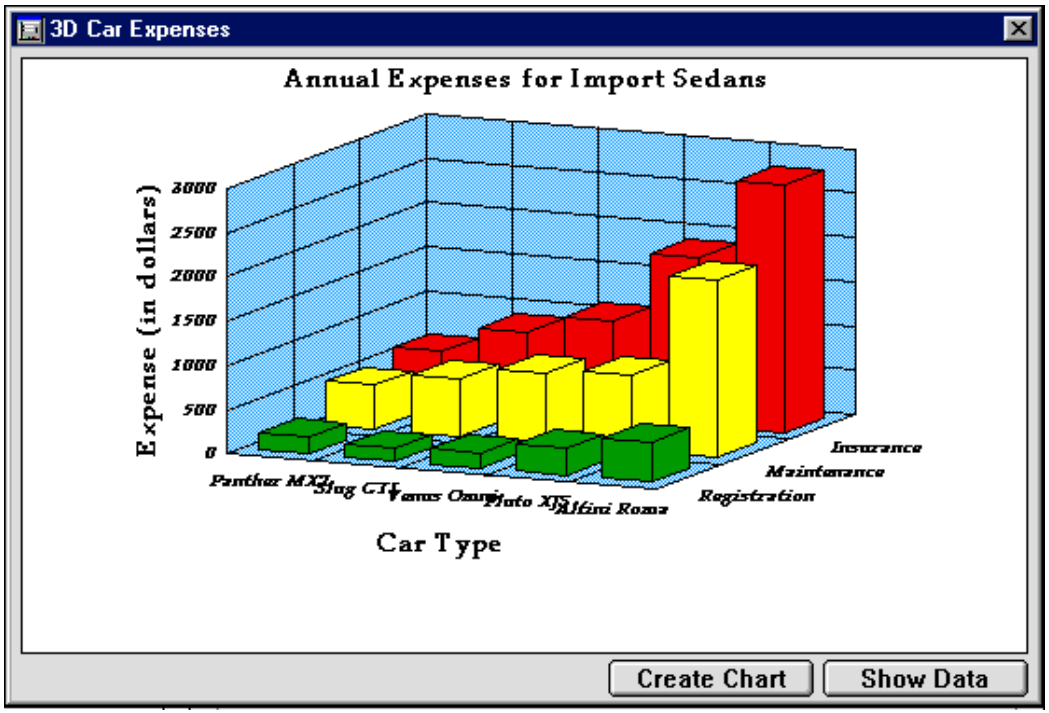
Creating a Three-Dimensional Graph With the Series Defined by Fields

Suppose that you are in the market for a new car and are using a database to determine the best car for your money. You have narrowed down your search to five cars and want to compare how much each car will cost for yearly registration, insurance, and maintenance.

You enter the data into the following table:

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

To analyze the information you have gathered, you decide to graph the information in a three-dimensional graph. Using 4D Chart, you produce the following graph:



The *GRAPH CARS* method was used to create this graph. This method uses the CT Chart selection command. When you use CT Chart selection, you pass it an array of fields. The field names become the series names, and the values in the fields are graphed on the Values axis. In this example, the Registration, Insurance, and Maintenance fields were used for the series and values.

The following is the *GRAPH CARS* method.

```
   `Method: GRAPH CARS
   `Categories: Car Type
   `Series: Type of expense
   `Values: Cost in dollars

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

 `Generate the selection of records to chart
ALL RECORDS ([Car Expenses])
ORDER BY ([Car Expenses];[Car Expenses]Cost;>)

 `Array of fields for series data
ARRAY LONGINT ($aFields;3)
$aFields{1}:=Field (->[Car Expenses]Registration)
$aFields{2}:=Field (->[Car Expenses]Maintenance)
$aFields{3}:=Field (->[Car Expenses]Insurance)

 `Hide interface elements
CT SET DISPLAY (Area;1;0) `Hide menus
CT SET DISPLAY (Area;2;0) `Hide Chart tools
CT SET DISPLAY (Area;3;0) `Hide Object tools
CT SET DISPLAY (Area;6;0) `Hide Scroll bars
CT SET DISPLAY (Area;9;0) `Hide Rulers

 `Create a 3D Column chart
$Chart:=CT Chart selection (Area;100;1;1;Table(->[Car Expenses]);1;$aFields)

 `Set up scale
CT SET REAL SCALE (Area;$Chart;0;0;0;0;0;0;3000;500;100)

 `Do not display legend
```

```
  `Add a title at the top left hand corner
$Title:=CT Draw text (Area;1;1;300;3;"Annual Expenses for Import Sedans")

  `Format title (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)

  `Set 1st series color/fill to green
$Color:=CT Index to color (10)
CT SET CHART FILL ATTRIBUTES (Area;$Chart;8;100;3;$Color)

  `Set 2nd series color/fill to yellow
$Color:=CT Index to color (2)
CT SET CHART FILL ATTRIBUTES (Area;$Chart;8;200;3;$Color)

  `Set 3rd series color/fill to red
$Color:=CT Index to color (4)
CT SET CHART FILL ATTRIBUTES (Area;$Chart;8;300;3;$Color)

  `Set fill attributes for all plot rectangles
$Color:=CT Index to color (8)
For ($i;1;3)
  CT SET CHART FILL ATTRIBUTES (Area;$Chart;1;$i;5;$Color)
End for

  `Set text attributes labels (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

  `Set text attributes for titles (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

  `Get area dimensions for centering
CT GET AREA BOUNDARY (Area;1;$Left;$Top;$Right;$Bottom)
```

```
 `Center title
CT GET BOUNDARY (Area;$Title;$Left2;$Top2;$Right2;$Bottom2)
$Locate:=(($Right-$Left)-($Right2-$Left2))/2
CT MOVE (Area;$Title;$Locate;$Top2)

 `Move the Chart 9 points down
CT GET BOUNDARY (Area;$Chart;$Left;$Top;$Right;$Bottom)
CT MOVE (Area;$Chart;$Left;$Top+9)

 `Deselect all objects
CT SELECT (Area;-1;0)
```

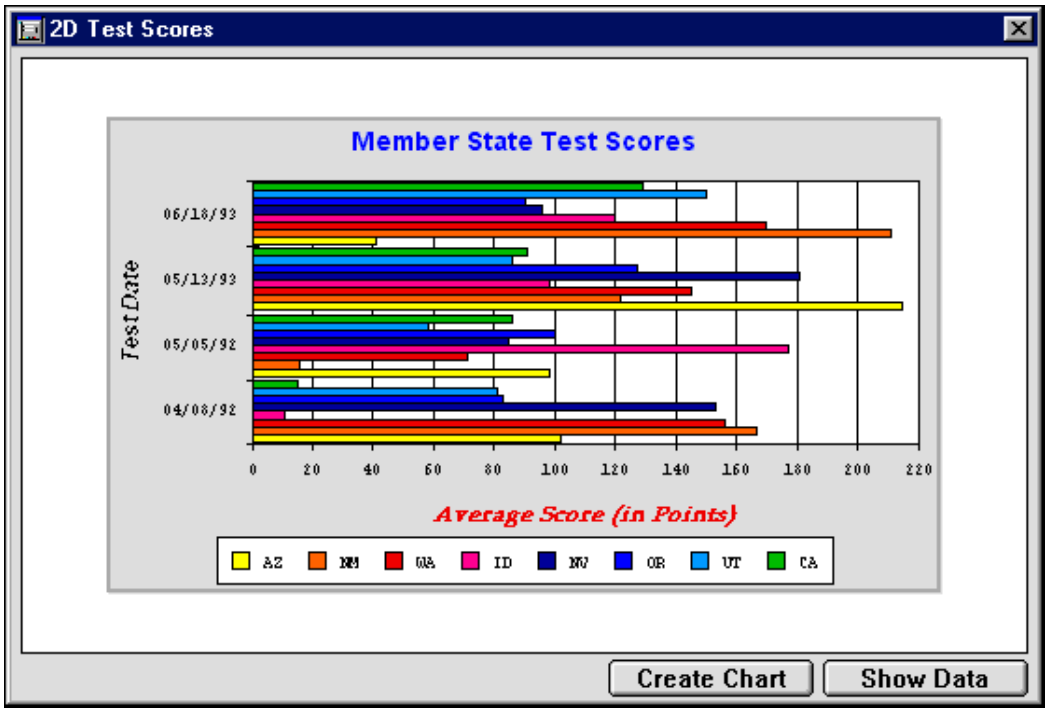
Creating a Two-Dimensional Graph With the Series Defined by Data in Records

Suppose you are Western Regional Director of an educational program, and you would like to compare standardized test scores for students in each state in your region on different test dates.

Your database contains the scores for students in your region, tagged by test date and home state.

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

Using 4D Chart, you create the following graph:



The preceding graph was created by the *GRAPH SCORES 2D* method. The following is the *GRAPH SCORES 2D* method:

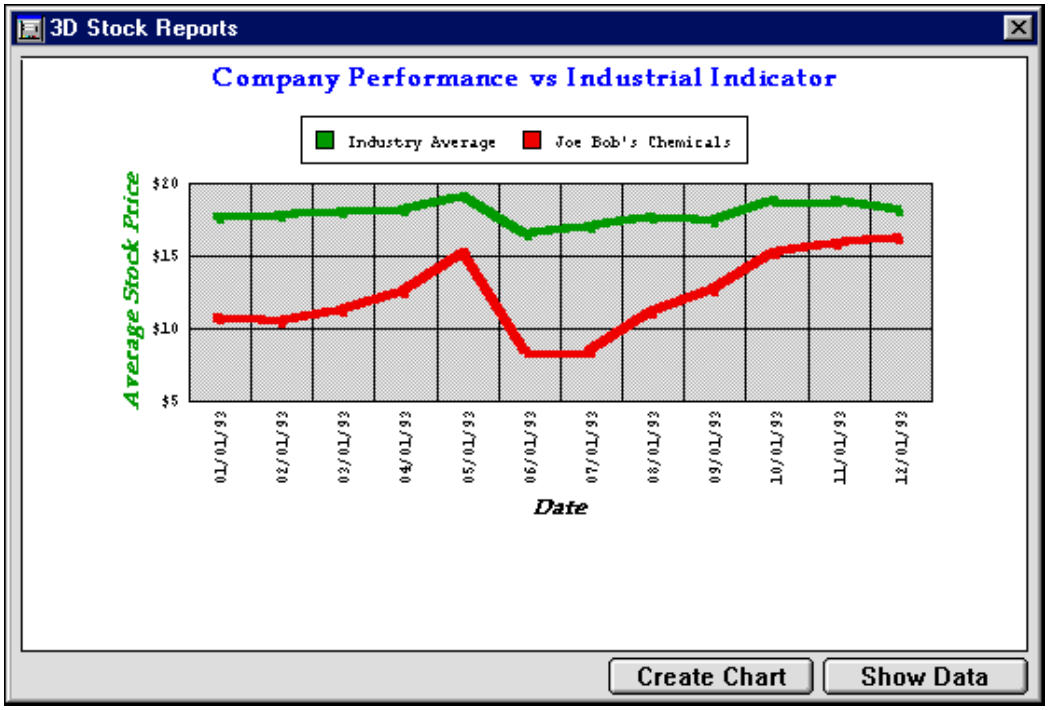
```
`Method: GRAPH SCORES 2D
`Categories: Test Dates
`Series: Member States
`Values: Test scores

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;>)
  `Hide interface elements
CT SET DISPLAY (Area;1;0)  `Hide menus
CT SET DISPLAY (Area;2;0)  `Hide Chart tools
CT SET DISPLAY (Area;3;0)  `Hide Object tools
```

```
 `Create a 2D bar chart
$Chart:=CT Chart data (Area;2;1;1;1;Table (->[Test Scores]);2;4;3)

 `Make it a horizontal chart
ARRAY LONGINT ($aOptions;4)
$aOptions{1}:=1 `orientation: horizontal
$aOptions{2}:=0 `stacked: normal
$aOptions{3}:=0 `Overlap:
$aOptions{4}:=50 `Gap:
CT SET CHART OPTIONS (Area;$Chart;$aOptions)

 `Show axis titles
CT SET TITLE ATTRIBUTES (Area;$Chart;0;2;3;"Test Date")
CT SET TITLE ATTRIBUTES (Area;$Chart;2;3;0;"Average Score (in Points)")

 `Format category axis title (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)

 `Format value axis title (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)

 `Set the legend position to bottom center, horizontal orientation
CT SET LEGEND ATTRIBUTES (Area;$Chart;1;0;0;0;8;0;0)

 `Add a chart title at the top left hand corner
$Title:=CT Draw text (Area;1;1;350;3;"Member State Test Scores")

 `Format chart title (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)

 `Use custom scale
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)
```

```
 `Center the title horizontaly
CT GET BOUNDARY (Area;$Title;$Left2;$Top2;$Right2;$Bottom2)
$Locate:=(($Right-$Left)-($Right2-$Left2))/2
CT MOVE (Area;$Title;$Locate;$Top2)

 `Move the Chart 10 points down from title
CT GET BOUNDARY (Area;$Chart;$Left;$Top;$Right;$Bottom)
CT MOVE (Area;$Chart;$Left;$Top+10)

 `Frame the chart and title with a grey rectangle
$Rect:=CT Draw rectangle (Area;$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)

 `Center all objects vertically
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)

 `Send the rectangle to the back
CT SELECT (Area;-1;0)  `Deselect all
CT SELECT (Area;$Rect;1)  `Select rectangle
CT DO COMMAND (Area;24002)  `send to back
CT SELECT (Area;-1;0)  `Deselect all

 `Deselect all objects
CT SELECT (Area;-1;0)
```

Creating a Three-Dimensional Graph With the Series Defined by Data in Records

Suppose you would like to know how your company's stock is performing compared to other companies in your industry. You record the daily closing value of each company's stock in a 4th Dimension database. In the example database, the stock values are recorded as monthly averages.



The *STOCKS CHART* method generates the graph using field values taken directly from the database. Since the values stored in the database are already calculated as averages, you do not need to perform any additional calculations. If you were recording daily values instead, you would need to calculate the monthly averages, store them in an array, and create the graph using the CT Chart arrays function.

For an example of averaging monthly values, see the section Creating a Two-Dimensional Graph from Arrays.

The following is the *STOCKS CHART* method:

```
`Method: GRAPH STOCKS
`Categories: Months  of the year
`Series: Company and Industrial Index
`Values: Average stock price

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

`Generate the selection of records to chart
ALL RECORDS ([Stock Reports])
ORDER BY ([Stock Reports];[Stock Reports]Month;>)

`Hide interface elements
CT SET DISPLAY (Area;1;0) `Hide menus
CT SET DISPLAY (Area;2;0) `Hide Chart tools
CT SET DISPLAY (Area;3;0) `Hide Object tools
CT SET DISPLAY (Area;6;0) `Hide Scroll bars
CT SET DISPLAY (Area;9;0) `Hide Rulers

`Create a 3D line chart
$Chart:=CT Chart data (Area;101;2;1;1;Table (->[Stock Reports]);2;1;3)

`Setup scale
CT SET REAL SCALE (Area;$Chart;0;0;0;0;5;20;5;1)

`Rotate the chart by 0 degrees in both directions
CT SET 3D VIEW (Area;$Chart;0;0)

`Set the plot recatangle fill/color (grey)
CT SET CHART FILL ATTRIBUTES (Area;$Chart;1;1;5;CT Index to color (15))

`Set the series fill/color (Green, Red)
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))

`Set the series line fill/color (Green, Red, 4 point)
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)
```

```
  `Show axis labels
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;"") `Hide these labels

  `Add category and value axis titles
CT SET TITLE ATTRIBUTES (Area;$Chart;0;3;0;"Date")
CT SET TITLE ATTRIBUTES (Area;$Chart;2;2;3;"Average Stock Price")

  `Set the legend position to top center, horizontal orientation
CT SET LEGEND ATTRIBUTES (Area;$Chart;1;0;0;0;7;0;0)

  `Add a chart title at the top left hand corner
$Title:=CT Draw text (Area;1;1;350;3;"Company Performance vs Industrial
    Indicator")

  `Format chart title (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)

  `Get area dimensions for centering
CT GET AREA BOUNDARY (Area;1;$Left;$Top;$Right;$Bottom)

  `Resize the chart to window size minus 50 points
CT SIZE (Area;$Chart;$Right-50;$Bottom-50)

  `Center chart
CT GET BOUNDARY (Area;$Chart;$Left2;$Top2;$Right2;$Bottom2)
$Locate:=(($Right-$Left)-($Right2-$Left2))/2
CT MOVE (Area;$Chart;$Locate;$Top2)

  `Center title
CT GET BOUNDARY (Area;$Title;$Left2;$Top2;$Right2;$Bottom2)
$Locate:=(($Right-$Left)-($Right2-$Left2))/2
CT MOVE (Area;$Title;$Locate;$Top2)

  `Move the Chart 10 points down
CT GET BOUNDARY (Area;$Chart;$Left;$Top;$Right;$Bottom)
CT MOVE (Area;$Chart;$Left;$Top+10)
```

This section provides examples that use the CT Chart arrays function to create two- and three-dimensional graphs.

If you want to create graphs programmatically from records in a database, refer to the section Creating Graphs from Records in a Database (examples).

In each example, you will find:

- A description of the situation used for the example,
- The structure of the example database,
- A graph that has been created entirely programmatically, using example data,
- The code used to create the example graph.

Creating a Two-Dimensional Graph from Arrays

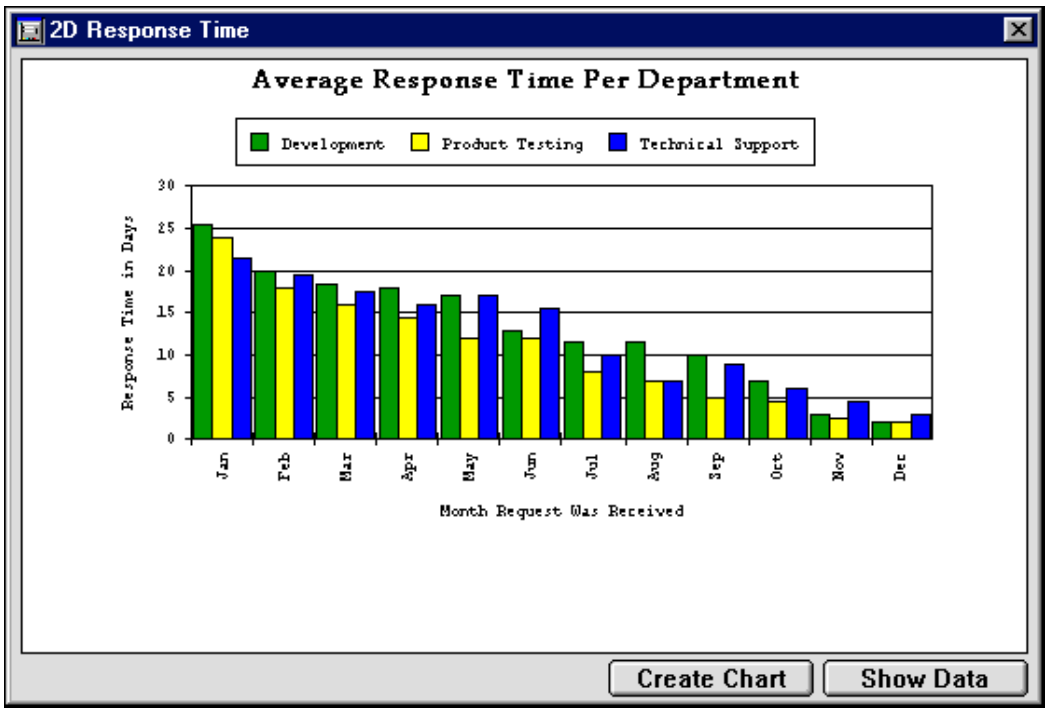
Suppose that you are a new manager in the Engineering division of a company that manufactures widgets. One of your new responsibilities is to decrease the amount of time your departments take to respond to customers' requests for help.

You decide to design a database to track the response time of each department. The structure of your database is as follows:

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

To analyze the information you have gathered, you decide to graph the average response time ((Completion Date - Request Date)/ # of requests) for each department for each month of the past year.

Using 4D Chart, you produce the following graph:



This graph was generated and formatted using the *GR RESPONSE* method. This method fills arrays with data, generates a graph from the arrays, and then customizes certain aspects of the graph, such as the axis titles and series colors.

The first part of the method uses 4th Dimension commands and functions to create and fill the arrays. The contents of the Categories array are hardcoded, the contents of the Series array are taken directly from the database, and the contents of the Values array are the results of data manipulation. The size of the Values array is equal to the size of the Categories array multiplied by the size of the Series array.

This method uses sets to manipulate the selection of records. After you have created a set, you can change the selection of records as needed by searching and then return to the original set of records again and again. In this method, a set is used to maintain the selection of records for the entire graph. The selection is altered during the filling of the Values array, because the values are determined by the selection that results from searching for the records for each department for each month.

The following is the *GR RESPONSE* method:

```
`Method: GRAPH RESPONSE
`Categories: Months  of the year
`Series: Department names
`Values: Average task duration (in days)

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)

`Define and fill the categories array
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"

`Generate the selection of records to chart
`Store the records in a set for later use
QUERY BY FORMULA ([Response Time];Year of ([Response Time]Request
   Date)=1993)
CREATE SET ([Response Time];"sChartData")

`Define and fill the Series array with department names
ARRAY STRING (20;$aSeries;0)
DISTINCT VALUES ([Response Time]Department;$aSeries)

`Determine the number of values to chart
`(number of values = categories * series)
`Dimension the values array
ARRAY REAL ($aValues;12*Size of array($aSeries))
```

```
  `Determine and fill the values array
  `For each department find average task durations per month
$Counter:=0  `counter to keep to track of values
For ($x;1;Size of array ($aSeries))  `...loop for the number of departments
  For ($y;1;12)  `...loop for the 12 months
    $Counter:=$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)
      $Duration:=0  `counter for incrementing durations
      For ($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")  `Restore the original selection of records
  End for
End for

  `Hide interface elements
CT SET DISPLAY (Area;1;0)  `Hide menus
CT SET DISPLAY (Area;2;0)  `Hide Chart tools
CT SET DISPLAY (Area;3;0)  `Hide Object tools
CT SET DISPLAY (Area;6;0)  `Hide Scroll bars
CT SET DISPLAY (Area;9;0)  `Hide Rulers

  `Create a column chart
$Chart:=CT Chart arrays (Area;2;1;$aCategories;$aSeries;$aValues)

  `Add category and value axis titles
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")

  `Set the series colors (Green, Yellow, Blue)
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))
```

```
 `Add a title at the top left hand corner
$Title:=CT Draw text (Area;1;1;300;3;"Average Response Time Per Department")

 `Format title (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)

 `Get area dimensions for centering
CT GET AREA BOUNDARY (Area;1;$Left;$Top;$Right;$Bottom)

 `Resize the chart to window size minus 50 points
CT SIZE (Area;$Chart;$Right-50;$Bottom-50)

 `Center chart
CT GET BOUNDARY (Area;$Chart;$Left2;$Top2;$Right2;$Bottom2)
$Locate:=(($Right-$Left)-($Right2-$Left2))/2
CT MOVE (Area;$Chart;$Locate;$Top2)

 `Center title
CT GET BOUNDARY (Area;$Title;$Left2;$Top2;$Right2;$Bottom2)
$Locate:=(($Right-$Left)-($Right2-$Left2))/2
CT MOVE (Area;$Title;$Locate;$Top2)

   `Move the Chart 10 points down
CT GET BOUNDARY (Area;$Chart;$Left;$Top;$Right;$Bottom)
CT MOVE (Area;$Chart;$Left;$Top+10)

 `Deselect all objects
CT SELECT (Area;-1;0)
```

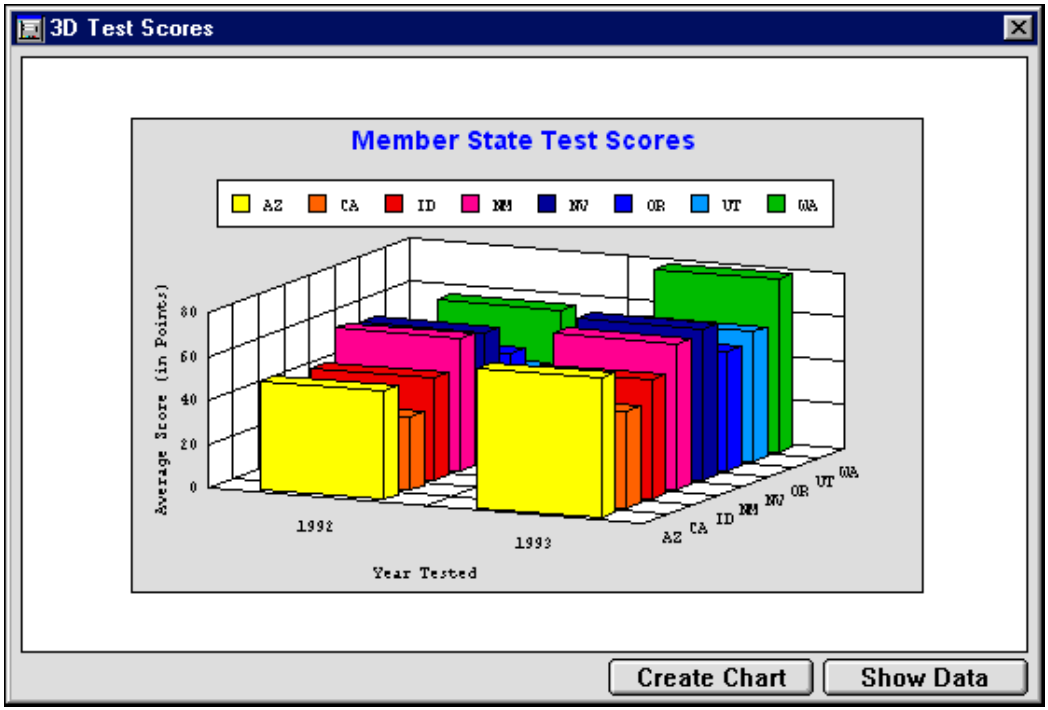
Creating a Three-Dimensional Graph From Arrays

Suppose you are Western Regional Director of an educational program, and you would like to determine whether the test scores of students in your region have been improving. Your database contains the scores for students in your region, tagged by test date and home state.

Note: This is the same data used in the example Creating a Two-Dimensional Graph with

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

Because there were multiple test dates in each year, you cannot graph the data directly from the database. Instead, you create a method that places the data in arrays, enabling you to use each year as a category and the individual states as the series.



This graph was generated using the *GRAPH SCORES 3D* method. This method fills the arrays with data, generates a graph from the arrays, and then customizes certain aspects of the graph.

The contents of the Categories array are hardcoded. The contents of the Series array are taken from the database. The contents of the Values array are the averages of each

Using 4D Chart commands, the 4D Chart menu bar, scroll bars, rulers, and tool palettes are hidden. The 4D Chart area is set to Non-enterable, meaning that the user cannot select any object in the area or make any changes to it.

The following is the *GRAPH SCORES 3D* method:

```
  `Method: GRAPH SCORES 3D
  `Categories: Survey years
  `Series: Member States
  `Values: Test scores

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)

  `Define and fill the categories array
ARRAY STRING (4;$aCategories;2)
$aCategories{1}:="1992"
$aCategories{2}:="1993"

  `Define and fill the Series array with State names
ALL RECORDS ([Test Scores])
ARRAY STRING (2;$aSeries;0)
DISTINCT VALUES ([Test Scores]State;$aSeries)

  `Dimension the values array
  `(number of values to chart = categories * series)
ARRAY REAL ($aValues;2*Size of array ($aSeries))

    `Determine and fill the values array
    `For each State find average score
$Counter:=0  `counter to keep to track of values
For ($x;1;Size of array ($aSeries))  `...loop for the number of States
  For ($y;1;2)  `...loop for the 2 years
    $Counter:=$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)
```

```
      `Average the score
      $aValues{$Counter}:=$Score/Records in selection ([Test Scores])
    End if
  End for
End for

  `Restore selection
ALL RECORDS ([Test Scores])

  `Hide interface elements
CT SET DISPLAY (Area;1;0)  `Hide menus
CT SET DISPLAY (Area;2;0)  `Hide Chart tools
CT SET DISPLAY (Area;3;0)  `Hide Object tools
CT SET DISPLAY (Area;6;0)  `Hide Scroll bars
CT SET DISPLAY (Area;9;0)  `Hide Rulers

  `Create a 3D column chart
$Chart:=CT Chart arrays (Area;100;1;$aCategories;$aSeries;$aValues)

  `Show/hide axis titles
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)")

  `Set the legend position to top center, horizontal orientation
CT SET LEGEND ATTRIBUTES (Area;$Chart;1;0;0;0;7;0;0)

  `Add a chart title at the top left hand corner
$Title:=CT Draw text (Area;1;1;350;3;"Member State Test Scores")

  `Format chart title (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)

  `Get window dimensions to use for centering
CT GET AREA BOUNDARY (Area;1;$Left;$Top;$Right;$Bottom)

  `Resize the chart to window size minus 50 points
CT SIZE (Area;$Chart;$Right-50;$Bottom-50)
```

```
 `Center the title horizontally
CT GET BOUNDARY (Area;$Title;$Left2;$Top2;$Right2;$Bottom2)
$Locate:=(($Right-$Left)-($Right2-$Left2))/2
CT MOVE (Area;$Title;$Locate;$Top2)

 `Move the Chart 10 points down from title
CT GET BOUNDARY (Area;$Chart;$Left;$Top;$Right;$Bottom)
CT MOVE (Area;$Chart;$Left;$Top+10)

 `Frame the chart and title with a grey rectangle
$Rect:=CT Draw rectangle (Area;$Left-2;$Top2-2;$Right+2;$Bottom+2+10;0)
CT SET FILL ATTRIBUTES (Area;$Rect;3;CT Index to color (13))

 `Center all objects vertically
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)

 `Send the rectangle to the back
CT SELECT (Area;-1;0) `Deselect all
CT SELECT (Area;$Rect;1) `Select rectangle
CT DO COMMAND (Area;24002) `send to back
CT SELECT (Area;-1;0) `Deselect all

 `Deselect all objects
CT SELECT (Area;-1;0
```

See Also

No reference.

2 CT Area Control

CT DO COMMAND

CT Area Control

version 6.0.5

CT DO COMMAND (area; command)

Parameter	Type		Description
area	Longint	→	4D Chart area
command	Longint	→	Number of command

Description

CT DO COMMAND executes the menu command specified by command. The menu command is executed as if the user had chosen it from a 4D Chart menu. Use this command to perform any actions that do not have an equivalent command in the language. area can be an offscreen area.

The possible values for command are listed in the Command Codes section. These numbers will remain the same even if menu items are changed or moved in future versions of 4D Chart.

Example

This example selects all the objects in the specified area and duplicates them.

```
      `Same as choosing Select All from the Edit menu
⇒     CT DO COMMAND (Area;2009)
      `Same as choosing Duplicate from the Edit menu
⇒     CT DO COMMAND (Area;2007)
```

See Also

No reference.

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

Parameter	Type		Description
area	Longint	→	4D Chart area
command	Longint	→	Number of command
checked	Integer	←	Is the menu item checked? 0 = Not checked 1 = Checked
available	Integer	←	Is the menu item available or dimmed? 0 = Disabled 1 = Enabled
name	String	←	Receives name of menu item

Description

The CT MENU STATUS command returns in the checked, available, and name parameters information about the menu item in area represented by command.

The possible values for command are listed in the Command Codes section.

If available equals 0, the menu item is disabled. If available equals 1, the menu item is enabled.

If checked equals 0, the menu item is not checked. If checked equals 1, the menu item is checked.

name is the text of the menu item.

Example

This example checks a menu item to see if the area is in Show References or Show Values mode. If the area is in Show References mode, Show Values mode is turned on.

```
⇒   CT MENU STATUS (Area;6006;$Checked;$Available;$Name)
    If ($Name="Show References")
       CT DO COMMAND (Area;6006)
    End if
```

See Also

CT DO COMMAND.

CT ON ERROR

CT Area Control

version 6.0.5

CT ON ERROR (method)

Parameter	Type		Description
method	String	→	Method to execute

Description

The CT ON ERROR command installs method as the method for managing 4D Chart errors. After the installation of the error handling method, 4D Chart calls method when a 4D Chart error occurs.

If method is an empty string, no method is called and error handling returns to 4D Chart.

When 4D Chart calls method, it returns three parameters ($1, $2, and $3) that can be used to manage the error.

Parameter	Type	Description
$1	Longint	Represents the 4D Chart area in which the error took place. If the error is not specific to a 4D Chart area, $1 equals 0.
$2	Longint	Holds the error number. Equivalent of a call to CT Error.
$3	Text	Contains the text of the error. Equivalent of a call to CT Error.

If you plan to compile your database, you should declare these parameter types as follows:

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

Example

This example shows the installation of an error-handling method.

```
⇒ CT ON ERROR ("CHART ERROR")
```

The following method is *CHART ERROR*. This method tests $1 to determine whether an error occurred in area and then presents an alert box with the error number and message

```
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)
```

See Also

CT Error.

CT ON EVENT

CT Area Control

version 6.0.5

CT ON EVENT (method)

Parameter	Type		Description
method	String	→	Method to execute

Description

The CT ON EVENT command executes method whenever a previously specified event occurs. The events that cause method to execute are described by the CT EVENT FILTER command.

If method is an empty string, no method is executed. If the area in which the event occurs has both an object method and an event method, the object method executes last. CT ON EVENT is especially useful for 4D Chart areas in plug-in windows because they cannot have object methods.

When 4D Chart calls method, it returns four parameters ($1, $2, $3, and $4) that can be used to manage the event.

Parameter	Type	Description
$1	Longint	Represents the 4D Chart area in which the event took place.
$2	Longint	Holds the event code. Equivalent of a call to CT Last event.
$3	Longint	Table number of the form on which the area resides. If $3 equals -1, the area is in a plug-in window.
$4	Longint	Number of the field into which the area is being autosaved. If $4 equals 0, the area is not being autosaved.

If you plan to compile your database, you should declare these parameter types as follows:

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

Example

This example shows the installation of an event method. It opens a plug-in window, specifies Ctrl+click as the event (Command-click on Macintosh), and then installs the event method *EventProc*.

```
   `Open plug-in window
vArea := Open external window (20;50;400;350;0;"Chart";"_4D Chart")
   `Install the method EventProc
⇒  CT ON EVENT ("EventProc")
   `Ctrl+click will call the method
   CT EVENT FILTER (vArea;64)
```

See Also

CT EVENT FILTER, CT Last event.

CT ON MENU

CT Area Control

version 6.0.5

CT ON MENU (area; method)

Parameter	Type		Description
area	Longint	→	4D Chart area
method	String	→	Name of the method to call

Description

The CT ON MENU command executes method each time a menu command is activated in either the User or Custom Menus environments. The menu command can also be called by using the CT DO COMMAND command, as long as the menu command is called in method.

The called method returns three parameters:

Parameter	Description
$1	A Long integer containing the ID for the 4D Chart area.
$2	A Long integer containing the menu item number.
$3	A Long integer containing the number of the modifier key pressed.

The $3 parameter corresponds to one of the following modifier keys (or combination of modifier keys):

Value	Modifier Key
0	No modifier
1	Ctrl (Windows) or Command (Macintosh) key
2	Shift key
4	Alt (Windows) or Option (Macintosh) key
8	Control key (Macintosh)

If a combination of modifier keys is pressed, the values are added together and passed as a parameter. For example, a value of 10 indicates that the user held down the Shift and Control keys while choosing a menu item.

If you plan to compile your database, you should declare the types of these parameters, as follows:

Example

This example launches the *MenuProc* event method.

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

The *MenuProc* method controls the user's access to menu commands. If either the Save as Template or Properties menu command is selected, the user is presented with an alert and the menu selection is ignored. All other menu items are executed without interruption.

Following is the code for the *MenuProc* method.

```
C_LONGINT ($1;$2;$3)
Case of
  : ($2=1006)  `Save as Template
    ALERT("You cannot save templates.")
  : ($2=2011)  `Properties
    ALERT("You do not have access to Properties.")
 Else
    CT DO COMMAND (vArea;$2)
End case
```

See Also

No reference.

CT Error (message) → Integer

Parameter	Type		Description
message	String	←	Receives error message
Function result	Integer	←	Status of the last operation performed by 4D Chart 0 = the last operation did not cause an error >0 = an error occurred during the last operation

Description

CT Error returns a number that represents the status of the last operation performed by 4D Chart.

If CT Error returns 0, the last operation did not cause an error. If CT Error returns a number other than 0, an error occurred during the last operation.

If several areas are active on the same form, CT Error returns the last error, without discriminating between areas.

See the section Error Codes for a complete list of error codes.

If the optional message parameter is passed to CT Error, it must be a text variable that will contain the text of the error after the call.

Example

This example checks to see if an error occurred in the previous command.

```
⇒   If (CT Error # 0)
       `The last operation caused an error
    End if
```

See Also

CT ON ERROR.

CT EVENT FILTER

CT Area Control

version 6.0.5

CT EVENT FILTER (area; filter)

Parameter	Type		Description
area	Longint	→	4D Chart area
filter	Longint	→	Events to process

Description

The CT EVENT FILTER command specifies which events cause the object method of area or the event method to be executed.

By default, an object method attached to a 4D Chart area is executed when the user selects an object outside of the area. With CT EVENT FILTER you can specify other events that execute the method. In addition, a method installed with the CT ON EVENT command will also be executed.

filter specifies the events to use, expressed as the sum of event codes. The following are the event codes:

Value	Event
-1	All events
0	No events
1	Area creation
2	Area deletion
4	Area activated (clicked or brought to the front)
8	Area deactivated (area is no longer active)
16	Object creation (create, paste, duplicate)
32	Object deletion (delete, cut, clear)
64	Ctrl+Click (Windows) or Command-click (Macintosh), not necessarily on object
128	Object was moved (nudge, alignment, move, etc.)
256	Object was resized (arrow keys, drag, etc.)
1024	Change in selected object(s)
2048	Double-click
4096	Object was reshaped

Example

This example adds Ctrl+click and double-click to the list of default events trapped by the chart area's object method.

⇒ ***CT EVENT FILTER*** (Area;64+2048)

See Also

CT Last event, CT ON EVENT.

CT EXPERT COMMAND (area; command; status)

Parameter	Type		Description
area	Longint	→	4D Chart area
command	Longint	→	Number of command
status	Integer	→	Status of the menu item in expert mode -1 = Returns current value 0 = Enabled 1 = Disabled

Description

CT EXPERT COMMAND enables or disables a menu command for 4D Chart expert mode.

• If status equals 0, the menu command specified by command is enabled in expert mode.
• If status is greater than 0, the item is disabled.
• If status is a variable equal to -1, CT EXPERT COMMAND returns the current status of the menu command in status (0 = Enabled; 1 = Disabled).

The possible values for command are listed in the Command Codes section.

If a menu item is disabled with CT EXPERT COMMAND, it can still be executed by calling CT DO COMMAND.

Example

This example disables the Paste Field menu command in the Database menu.

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

See Also

CT EXPERT MODE.

CT EXPERT MODE

CT Area Control

version 6.0.5

CT EXPERT MODE (area; mode)

Parameter	Type		Description
area	Longint	→	4D Chart
mode	Integer	→	Turn expert mode on? -1 = Returns current value 0 = Off 1 = On

Description

The CT EXPERT MODE command turns expert mode on or off. When 4D Chart is in expert mode, certain items on 4D Chart menus may be disabled; expert mode is configured with the CT EXPERT COMMAND command.

• If mode equals 1, expert mode is invoked. When expert mode is invoked, the menu items previously specified with CT EXPERT COMMAND are disabled.
• If mode is equal to 0, expert mode is off.
• If mode is a variable equal to -1, CT EXPERT MODE returns the mode back into mode (0 = Off; 1 = On).

Example

This example disables the Go to Full Window menu command in the File menu.

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

See Also

CT EXPERT COMMAND.

CT Last event

CT Area Control

version 6.0.5

CT Last event (area) → Longint

Parameter	Type		Description
area	Longint	→	4D Chart area
Function result	Longint	←	Code of the code of the last event that occurred in area

Description

CT Last event returns the code of the last event that occurred in area.

CT Last event can be used in the object method of a 4D Chart area or in an event method installed with CT ON EVENT. CT Last event identifies the event that caused the object or project method to execute. When used with the CT EVENT FILTER command, CT Last event allows you to take action based on a user's actions.

The following table lists the event codes:

Value	Event
-1	All events
0	No events
1	Area creation
2	Area deletion
4	Area activated (clicked or brought to the front)
8	Area deactivated (area is no longer active)
16	Object creation (create, paste, duplicate)
32	Object deletion (delete, cut, clear)
64	Command-click (not necessarily on object)
128	Object was moved (nudge, alignment, move, etc.)
256	Object was resized (arrow keys, drag, etc.)
1024	Change in selected object(s)
2048	Double-click
4096	Object was reshaped

Example

In this example, CT Last event is used in a method installed by CT ON EVENT to identify double-clicks. When a user double-clicks a graph, a custom dialog box is displayed to allow the user to make changes to the graph.

```
⇒    If (CT Last event (Area)=2048)              `If it is a double-click
        If (CT Get object type (Area;0)=5)       `If it is a graph
              `Open the custom Change Chart dialog box
           CHANGE CHART (Area;CT Get ID (Area;0;1))
        End if
     End if
```

See Also

CT EVENT FILTER, CT Last event.

CT SET ENTERABLE (area; mode{; buttonMode})

Parameter	Type		Description
area	Longint	→	4D Chart area
mode	Integer	→	Enterable or non-enterable 0 = Non-enterable 1 = Enterable
buttonMode	Integer	→	0 = Button if area is less than 150 points high 1 = Always an area

Description

The CT SET ENTERABLE command controls access to the document in area.

- If mode equals 1, area is enabled and functions normally.
- If mode equals 0, area is disabled.

A disabled area cannot be modified by the user but can be modified by the language. When an area is disabled, the user can scroll through the area and copy the selected objects to the Clipboard. The user cannot change the selection or use the 4D Chart menu bar or tool palettes.

The optional parameter buttonMode allows you to prevent an area smaller than 150 points high from becoming a button.

Example

This example is a form method that makes area non-enterable.

```
    If (Form event=On Load)
⇒       CT SET ENTERABLE (Area;0)     `Make the area non-enterable
    End if
```

See Also

CT SET DISPLAY.

CT Get display

CT Area Control

version 6.0.5

CT Get display (area; item) → Integer

Parameter	Type		Description
area	Longint	→	4D Chart area
item	Integer	→	Item for which to get information
Function result	Integer	←	0 = the specified item is not being displayed 1 = the item is being displayed

Description

Use CT Get display to find out whether certain features of the 4D Chart window are being displayed.

CT Get display returns 0 if the specified item is not being displayed, and 1 if the item is being displayed.

The menu bar, Chart Tool palette, Object Tool palette, scroll bars, and rulers can be hidden or displayed in the User environment or programmatically, using the CT SET DISPLAY command.

The codes for the item parameter are:

Code	Item
1	Menu Bar
2	Chart Tools
3	Object Tools
6	Scroll Bars
9	Rulers

Example

This example checks to see if the menu bar is disabled and then disables it if it is not.

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

See Also

CT SET DISPLAY.

CT SET DISPLAY

CT Area Control

version 6.0.5

CT SET DISPLAY (area; item; displayCode)

Parameter	Type		Description
area	Longint	→	4D Chart area
item	Integer	→	Item to display or hide (see codes)
displayCode	Integer	→	Display the item? 0 = Hide 1 = Show 2 = Toggle

Description

The CT SET DISPLAY command sets whether the specified item is displayed or hidden in the 4D Chart window.

The menu bar, Chart Tool palette, Object Tool palette, scroll bars, and rulers can be hidden or displayed using the CT SET DISPLAY command.

The codes for the item parameter are:

Code	Item
1	Menu Bar
2	Chart Tools
3	Object Tools
6	Scroll Bars
9	Rulers

Example

This example hides the 4D Chart menu bar, Chart Tool palette, Object Tool palette, and rulers.

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

See Also

CT Get display.

CT GET DOCUMENT SIZE

CT Area Control

version 6.0.5

CT GET DOCUMENT SIZE (area; width; height)

Parameter	Type		Description
area	Longint	→	4D Chart area
width	Real	←	Receives width of the document (in points)
height	Real	←	Receives height of the document (in points)

Description

The CT GET DOCUMENT SIZE command returns the size of the document area. A 4D Chart document can measure up to 3500 x 3500 points.

width is the width of the document area in points.

height is the height of the document area in points.

Example

This example uses the CT GET DOCUMENT SIZE command to get the current document size before changing it.

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

See Also

CT SET DOCUMENT SIZE.

CT SET DOCUMENT SIZE

CT Area Control

version 6.0.5

CT SET DOCUMENT SIZE (area; width; height)

Parameter	Type		Description
area	Longint	→	4D Chart area
width	Real	→	Width of the document (in points) -1 = No change
height	Real	→	Height of the document (in points) -1 = No change

Description

The CT SET DOCUMENT SIZE command sets the size of the document area. The default size of a 4D Chart document is 588 by 768 pixels. This size may differe from the the size of a 4D Chart plug-in area or plug-in window. A 4D Chart document can measure up to 3500 x 3500 points.

width is the width of the document area in points.

height is the height of the document area in points.

Example

This example, which could be placed in the On Startup database method, sets the default document size for all new documents to 2208 x 1460 points.

⇒ ***CT SET DOCUMENT SIZE*** (-1;2208;1460)

See Also

CT GET DOCUMENT SIZE.

CT GET PROPERTIES

CT Area Control

version 6.0.5

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

Parameter	Type		Description
area	Longint	→	4D Chart area
printOrder	Integer	←	Receives print order 0 = By row 1 = By column
changeAlert	Integer	←	Receives graph type change alert 0 = No alert 1 = Alert
hotlinkType	Integer	←	Receives default graph type for hot links
saveAlert	Integer	←	Receives save alert 0 = No alert 1 = Alert

Description

The CT GET PROPERTIES command gets the properties that have been set for the specified 4D Chart area.

printOrder is the order in which pages of the document are printed. The print order affects only the order in which the document prints; it does not affect the page orientation.

changeAlert specifies whether the user will receive an alert box when attempting to change the type of a graph. The user has the option to cancel or continue with the change.

hotlinkType is the default type of a graph created from data stored in a hot link. The codes for this parameter are listed in the following table:

Code	Chart Type
1	Area
2	Column
3	Picture
4	Line
5	Scatter
6	Pie
7	Polar
8	XY Chart
100	3D Column
101	3D Line
102	3D Area
103	3D Surface
104	3D Triangle
105	3D Spike

saveAlert specifies whether the user will receive an alert box when closing a 4D Chart document with unsaved changes.

• If saveAlert is equal to 1, 4D Chart presents a standard alert box when the user closes a 4D Chart document with unsaved changes. The alert box gives the user the option of saving the changes, not saving the changes, or returning to the document without closing.

• If saveAlert is equal to 0, 4D Chart does not save the changes or present an alert to the user. You are responsible for saving the changes. The exception is a 4D Chart area on a form that is being saved in a Picture field; the contents of these areas are automatically saved in the Picture field.

Example

This example returns area properties — print order, graph type change alert, and hot link chart type — in the $POrder, $CAlert, $HType, and $SAlert variables.

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

See Also

CT SAVE DOCUMENT, CT SET PROPERTIES.

CT SET PROPERTIES

CT Area Control

version 6.0.5

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

Parameter	Type		Description
area	Longint	→	4D Chart area
printOrder	Integer	→	Print order 0 = Across 1 = Down -1 = No change
changeAlert	Integer	→	Graph type change alert 0 = No alert 1 = Alert -1 = No change
hotlinkType	Integer	→	Default graph type for hot links -1 = No change
saveAlert	Integer	→	Alert box when closing a document 0 = No alert 1 = Alert -1 = No change

Description

The CT SET PROPERTIES command gets the properties that have been set for the specified 4D Chart area.

printOrder is the order in which pages of the document are printed. The print order affects only the order in which the document prints; it does not affect the page orientation.

changeAlert specifies whether the 4D Chart presents an alert box when the user attempts to change the type of a graph. The user has the option to cancel or continue with the change.

hotlinkType is the default type of a graph created from data stored in a hot link. The codes for this parameter are listed in the description for the CT GET PROPERTIES command.

saveAlert specifies whether 4D Chart presents an alert box when the user closes a 4D Chart document with unsaved changes.

• If saveAlert is equal to 1, 4D Chart presents a standard alert box when the user closes a 4D Chart document with unsaved changes. The alert box gives the user the option of saving the changes, not saving the changes, or returning to the document without closing.

• If saveAlert is equal to 0, 4D Chart does not save the changes or present an alert to the user. You are responsible for saving the changes. The exception is a 4D Chart area on a form that is being saved into a Picture field; the contents of these areas are automatically saved to the Picture field.

Example

This example sets the default hot link chart type to 3D Surface without changing the other properties.

⇒ ***CT SET PROPERTIES*** (Area;-1;-1;103;-1)

See Also

CT GET PROPERTIES, CT SAVE DOCUMENT.

3 CT Objects

CT Draw text

CT Objects

version 6.0.5

CT Draw text (area; left; top; right; bottom; text) → Longint

Parameter	Type		Description
area	Longint	→	4D Chart area
left	Real	→	Horizontal position of start (in points)
top	Real	→	Vertical position of start (in points)
right	Real	→	Horizontal position of end (in points)
bottom	Real	→	Vertical position of end (in points)
text	Text	→	Text of new text object
Function result	Longint	←	The new object's Object ID

Description

CT Draw text creates a new text object in area and returns the new object's Object ID. The object is positioned according to the left, top, right and bottom coordinates.

Example

The following method draws the text "Hello World" in the top left corner of the chart area.

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

See Also

No reference.

CT Draw line

CT Objects

version 6.0.5

CT Draw line (area; left; top; right; bottom; arrowhead) → Longint

Parameter	Type		Description
area	Longint	→	4D Chart area
left	Real	→	Horizontal position of start (in points)
top	Real	→	Vertical position of start (in points)
right	Real	→	Horizontal position of end (in points)
bottom	Real	→	Vertical position of end (in points)
arrowhead	Integer	→	Arrow location code (see table)
Function result	Longint	←	The new object's Object ID

Description

CT Draw line creates a new line object in area and returns the new object's Object ID. The object is positioned according to the left, top, right, and bottom coordinates.

The following are the codes for the arrowhead parameter:

Code	Location
-1	Use default
0	None
1	At start
2	At end
3	Both ends

Example

The following method draws a line with arrowheads in the chart area:

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

See Also

No reference.

CT Draw rectangle

CT Objects

version 6.0.5

CT Draw rectangle (area; left; top; right; bottom; round) → Longint

Parameter	Type		Description
area	Longint	→	4D Chart area
left	Real	→	Horizontal position of start (in points)
top	Real	→	Vertical position of start (in points)
right	Real	→	Horizontal position of end (in points)
bottom	Real	→	Vertical position of end (in points)
round	Real	→	Corner rounding amount (in points)
Function result	Longint	←	The new object's Object ID

Description

CT Draw rectangle creates a new rectangle object in area and returns the new object's Object ID. The object is positioned according to the left, top, right, and bottom coordinates.

round controls the amount of corner rounding in the new rectangle. If round equals zero, there is no corner rounding.

Example

The following method draws a rounded rectangle in the chart area:

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

See Also

No reference.

CT Draw oval

CT Objects

version 6.0.5

CT Draw oval (area; left; top; right; bottom) → Longint

Parameter	Type		Description
area	Longint	→	4D Chart area
left	Real	→	Horizontal position of start (in points)
top	Real	→	Vertical position of start (in points)
right	Real	→	Horizontal position of end (in points)
bottom	Real	→	Vertical position of end (in points)
Function result	Longint	←	The new object's Object ID

Description

CT Draw oval creates a new oval object in area and returns the new object's Object ID. The object is positioned according to the left, top, right and bottom coordinates.

Example

The following method draws a circle in the chart area:

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

See Also

No reference.

CT Array to polygon

CT Objects

version 6.0.5

CT Array to polygon (area; arrayH; arrayV) → Longint

Parameter	Type		Description
area	Longint	→	4D Chart area
arrayH	Real array	→	Array of horizontal values for vertices
arrayV	Real array	→	Array of vertical values for vertices
Function result	Longint	←	The new object's Object ID

Description

CT Array to polygon creates a new polygon in area based on the arrays arrayH and arrayV and returns the new object's ID.

arrayH and arrayV describe the position of each vertex of the polygon. The two arrays can be of type Real, Long Integer, or Integer, and are specified in points. Each array must have at least three elements for the polygon to be successfully created. If the arrays are not the same size, the extra elements in the larger array are ignored. To create a closed polygon, the last value in each array must match the first value.

Example

This example fills two arrays and creates a polygon from them. It then moves and resizes the polygon:

```
$Vertices:=Num (Request ("Enter number of vertices:"))
If (OK=1)   `Declare the arrays
   ARRAY REAL (aVerticeH;$Vertices)
   ARRAY REAL (aVerticeV;$Vertices)
   For ($i;1;$Vertices)   `Fill the arrays
      aVerticeH{$i}:=Sin($i)
      aVerticeV{$i}:=Cos($i)
   End for
      `Draw the polygon
⇒  $Poly:= CT Array to polygon (Area;aVerticeH;aVerticeV)
      `Move the polygon to the area cordinates (10,10)
   CT MOVE (Area;$Poly;10;10)
   CT SIZE (Area;$Poly;200;200)   `Resize the polygon to 200x200
End if
```

See Also

No reference.

CT Place picture

CT Objects

version 6.0.5

CT Place picture (area; picture; left; top) → Longint

Parameter	Type		Description
area	Longint	→	4D Chart area
picture	Picture	→	4D picture to place
left	Real	→	Left coordinate, in points from left
top	Real	→	Top coordinate, in points from top
Function result	Longint	←	The picture's Object ID

Description

CT Place picture pastes picture into area at a point determined by left and top and returns the picture's object ID.

Picture must be a valid 4th Dimension picture expression.

Example

This example pastes the contents of the Picture field [Logos]Logo into Area for the specified company.

```
    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
```

See Also

No reference.

CT Get ID

CT Get ID (area; scope; index) → Longint

Parameter	Type		Description
area	Longint	→	4D Chart area
scope	Longint	→	Scope of the function -1 = All 0 = Selected objects >0 = Group ID
index	Longint	→	Number of object in scope
Function result	Longint	←	The object's unique Object ID

Description

CT Get ID returns the unique ID for the object in area described by scope and index. This number is used by many other 4D Chart commands and is referred to as an object's ID.

To get an object's ID, you first specify which set of objects to refer to and then the order of the object within the set. Objects are ordered from back to front. The backmost object has an index of 1.

• If scope equals -1, then index refers to the order of the object within the entire document.
• If scope equals 0, then index refers to the order of the object within the currently selected objects.
• If scope is greater than 0, it must be the ID for a group and index refers to the order of objects within the group. This last syntax lets you manipulate objects in a group without ungrouping.

Example

This example shows how to extract the ID for a selected object.

```
⇒    vID := CT Get ID (Area;0;1)      `Get the ID of the first selected object
```

See Also

No reference.

CT Get object type

CT Objects

version 6.0.5

CT Get object type (area; scope) → Integer

Parameter	Type		Description
area	Longint	→	4D Chart area
scope	Longint	→	Scope of the function -1 = All 0 = Selected objects >0 = Object ID
Function result	Integer	←	Object type of the objects in area

Description

CT Get object type returns the object type for the object(s) in area described by scope. An object's type is described by an integer code which cannot be changed once an object is created.

• If scope equals -1, CT Get object type returns the object type for all objects in the document. If the types of the objects are not equal, CT Get object type returns -32000.

• If scope equals 0, CT Get object type returns the object type for the selected objects. If the types of the objects are not equal, CT Get object type returns -32000.

• If scope is greater than 0, it must be equal to a specific object's ID and that object's type is returned. If the object does not exist, CT Get object type returns -32000.

The following table lists all object codes:

Code	Object Type
1	Text
2	Hot link
3	Picture
4	*not used*
5	Chart
6	Rectangle
7	Polygon
8	Oval
9	*not used*
10	Line

Example

This example returns the ID of the selected object in the $ID variable.

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

See Also

CT Get ID.

CT Get refnum

CT Objects

version 6.0.5

CT Get refnum (area; scope) → Longint

Parameter	Type		Description
area	Longint	→	4D Chart area
scope	Longint	→	Scope of the function -2 = Default -1 = All 0 = Selected objects >0 = Object ID
Function result	Longint	←	Reference number for the objects in area

Description

CT Get refnum returns the reference number for the object(s) in the area described by scope. A reference number is a long integer associated with an object and is not necessarily unique. Reference numbers can only be manipulated programmatically. You assign a reference number to an object. The object ID, on the other hand, is assigned by 4D Chart.

• If scope equals -2, CT Get refnum returns the default reference number.

• If scope equals -1, CT Get refnum returns the reference number for all objects in the document. If the reference numbers for the objects are not equal, CT Get refnum returns -32000.

• If scope equals 0, CT Get refnum returns the reference number for the selected objects. If the reference numbers for the objects are not equal, CT Get refnum returns -32000.

• If scope is greater than 0, it must be equal to a specific object's ID and that object's reference number is returned. If the object does not exist, CT Get refnum returns -32000.

Example

This example is the object method for a button on a form that contains Area. When the method executes, it checks to see if only one object is selected, searches in the [Parts] table for the corresponding record, and displays its description

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

See Also

CT Get ID, CT SET REFNUM.

CT SET REFNUM (area; scope; refNum)

Parameter	Type		Description
area	Longint	→	4D Chart area
scope	Longint	→	Scope of the command -2 = Default -1 = All 0 = Selected objects >0 = Object ID
refNum	Longint	→	Reference number

Description

CT SET REFNUM makes refNum the reference number for the objects in area described by scope. A reference number is like an object name: it is a way to identify an object but it is not unique. A reference number is not an object ID number, which is a unique number assigned by 4D Chart for each object in a document.

• If scope equals -2, CT SET REFNUM sets the default reference number. This is the reference number that will be used for any new objects.
• If scope equals -1, CT SET REFNUM sets the reference number for all objects in the document.
• If scope equals 0, CT SET REFNUM sets the reference number for the selected objects.
• If scope is greater than 0, it must be equal to a specific object's ID. If the object does not exist, CT SET REFNUM does nothing.

A reference number is a non-unique long integer associated with an object. Reference numbers can only be manipulated programmatically. The default value for reference numbers is 0.

Example

This example changes the reference number of the selected objects to the value contained in the vNumber process variable.

⇒ ***CT SET REFNUM*** (Area;0;vNumber)

See Also

CT Get ID, CT Get refnum.

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

Parameter	Type		Description
area	Longint	→	4D Chart area
scope	Longint	→	Scope of the command -1 = All 0 = Selected object >0 = Object ID
left	Real	←	Receives horizontal position of start (in points)
top	Real	←	Receives vertical position of start (in points)
right	Real	←	Receives horizontal position of end (in points)
left	Real	←	Receives vertical position of end (in points)

Description

The CT GET BOUNDARY command returns in left, top, right, and bottom the boundary of the object(s) in area described by scope.

The boundary is the coordinates of the smallest rectangular region that contains the object(s).

• If scope equals -1, CT GET BOUNDARY returns the boundary for all objects in the document.
• If scope equals 0, CT GET BOUNDARY returns the boundary for the selected objects.
• If scope is greater than 0, it must be equal to a specific object's ID, and that object's boundary is returned. If the object does not exist, CT GET BOUNDARY returns -32000 for each coordinate.

Example

The following line returns the selected object's boundary in the $left, $top; right, and $bottom variables.

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

See Also

CT GET AREA BOUNDARY.

CT GET TEXT ATTRIBUTES

CT Objects

version 6.0.5

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

Parameter	Type		Description
area	Longint	→	4D Chart area
scope	Longint	→	Scope of the command -3 = Selected text -2 = Default -1 = All 0 = Selected object >0 = Object ID
fontID	Integer	←	Receives font ID
fontSize	Integer	←	Receives font size (in points)
style	Integer	←	Receives font style
color	Longint	←	Receives text color
justification	Integer	←	Receives justification of the text 0 = Left 1 = Center 2 = Right

Description

The CT GET TEXT ATTRIBUTES command returns the text attributes of the text specified by area and scope in the fontID, size, style, color, and justification parameters.

fontID is the ID of the font in your system. You can get the ID number of a font using the CT Font number function.

fontSize is the size in points of the highlighted text or text object(s).

style is a composite number that results from the addition of several style numbers. The following table lists the style numbers:

Value	Style
1	Plain
2	Bold
3	Italic
4	Underline

color is a long integer that specifies the color of the object. You can specify a value for the color parameter by using the CT Index to color or the CT RGB to color functions.

justification is the justification of the text.

Note: This command should be used to get the attributes of text added to the document using the Text tool or the CT Draw text function. To get the attributes of chart text, such as axis labels, use the commands in the Chart theme.

Example

This example returns the attributes of the selected text object in the $Font, $Size, $Style, $Color, and $Justify variables.

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

See Also

CT GET CHART TEXT ATTRIBUTES, CT SET TEXT ATTRIBUTES.

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

Parameter	Type		Description
area	Longint	→	4D Chart area
scope	Longint	→	Scope of the command -3 = Selected text -2 = Default -1 = All 0 = Selected objects >0 = Object ID
fontID	Integer	→	Font ID
fontSize	Integer	→	Font size (in points)
style	Integer	→	Font style
color	Longint	→	Text color
justification	Integer	→	Justification of the text 0 = left 1 = Center 2 = Right

Description

The CT SET TEXT ATTRIBUTES command sets the font, font size, font style, color, and justification of the text specified by area and scope.

fontID is the ID of the font in your system. You can get the ID number of a font using the CT Font number function.

fontSize is the size in points of the highlighted text or text object(s).

style is a composite number that results from the addition of several style numbers. The following table lists the style numbers:

Value	Style
1	Plain
2	Bold
3	Italic
4	Underline

color is a long integer that specifies the color of the object. You can specify a value for the color parameter by using the CT Index to color or the CT RGB to color functions.

justification is the justification of the text.

Note: This command should be used to set the attributes of text added to the document using the Text tool or the CT Draw text function. To set the attributes of chart text, such as axis labels, use the commands in the Chart theme.

Example

This example sets the selected text to Times, 14 point, Bold Italic, Green, Centered.

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

See Also

CT GET TEXT ATTRIBUTES, CT SET CHART TEXT ATTRIBUTES.

CT GET LINE ATTRIBUTES

CT Objects

version 6.0.5

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

Parameter	Type		Description
area	Longint	→	4D Chart area
scope	Longint	→	Scope of the command -2 = Default -1 = All 0 = Selected objects >0 = Object ID
pattern	Integer	←	Receives pattern index
color	Longint	←	Receives color value
lineWidth	Real	←	Receives line width (in points)

Description

The CT GET LINE ATTRIBUTES command returns in the variables the line attributes for the object(s) in area described by scope. For objects other than lines, the line attributes apply to the object's border.

pattern is the number of the pattern in the palette. The following are the codes for the pattern parameter:

color is a long integer that specifies the color of the object. You can specify a value for the color parameter by using the CT Index to color or the CT RGB to color functions.

lineWidth is the width of the line, measured in points.

Example

This example returns the selected object's line attributes in the $Pattern, $Color, and $Width variables.

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

See Also

CT GET CHART LINE ATTRIBUTES, CT SET LINE ATTRIBUTES.

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

Parameter	Type		Description
area	Longint	→	4D Chart area
scope	Longint	→	Scope of the command -2 = Default -1 = All 0 = Selected objects >0 = Object ID
pattern	Integer	→	Pattern (0 to 36) -1 = No change
color	Longint	→	Color value -1 = No change
lineWidth	Real	→	Line width (in points) -1 = No change

Description

The CT SET LINE ATTRIBUTES command changes the line attributes for the object(s) in area described by scope. For objects other than lines, the line attributes apply to the object's border.

pattern is the number of the pattern in the palette. Codes for the pattern parameter are as follows:

color is a long integer that specifies the color of the object. You can specify a value for the color parameter by using the CT Index to color or the CT RGB to color functions.

Note: Use this command to set the attributes of lines added to the document using the Line tool or the CT Draw line function. To set the attributes of chart lines, such as grid lines, use the commands in the Chart theme.

Example

This example sets the selected object's line attributes to solid, blue, 3 points.

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

See Also

CT GET LINE ATTRIBUTES, CT SET CHART LINE ATTRIBUTES.

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

Parameter	Type		Description
area	Longint	→	4D Chart area
scope	Longint	→	Scope of the command -2 = Default -1 = All 0 = Selected objects >0 = Object ID
pattern	Integer	←	Receives pattern (0 to 36)
color	Longint	←	Receives color value

Description

The CT GET FILL ATTRIBUTES command returns in the variables the fill attributes for the object(s) in area described by scope. Fill attributes are determined by the interiors of objects.

pattern is the number of the pattern in the palette. The following are the codes for the pattern parameter:

color is a long integer that specifies the color of the object. You can specify a value for the color parameter by using the CT Index to color or the CT RGB to color functions.

Note: Use this command to get the attributes of objects added to the document using the drawing tools or the drawing functions in this theme. To get the attributes of chart objects, such as series columns, use the commands in the Chart theme.

Example

This example returns the selected object's fill attributes in the $Pattern and $Color variables.

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

See Also

CT GET CHART FILL ATTRIBUTES, CT SET FILL ATTRIBUTES.

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

Parameter	Type		Description
area	Longint	→	4D Chart area
scope	Longint	→	Scope of the command -2 = Default -1 = All 0 = Selected objects >0 = Object ID
pattern	Integer	→	Pattern (0 to 36) -1 = No change
color	Longint	→	Color value -1 = No change

Description

The CT SET FILL ATTRIBUTES command changes the fill attributes for the object(s) in area described by scope. Fill attributes are determined by the interiors of objects.

pattern is the number of the pattern in the palette. The following are the codes for the pattern parameter:

color is a long integer that specifies the color of the object. You can specify a value for the color parameter by using the CT Index to color or the CT RGB to color functions.

Note: Use this command to set the attributes of objects added to the document using

Example

This example sets the selected object's fill attributes to solid yellow.

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

See Also

CT GET FILL ATTRIBUTES, CT SET CHART FILL ATTRIBUTES.

CT SELECT (area; scope; action)

Parameter	Type		Description
area	Longint	→	4D Chart area
scope	Longint	→	Scope of the command -1 = All 0 = Selected objects >0 = Object ID
action	Integer	→	Select or deselect object? 0 = Deselect 1 = Select 2 = Toggle

Description

The CT SELECT command selects or deselects the objects in area described by scope.

- If scope equals -1, CT SELECT affects all objects in the document.
- If scope equals 0, CT SELECT affects the selected objects.
- If scope is greater than 0, it must be equal to a specific object's ID and that object is affected. If the object does not exist, CT SELECT does nothing.

The object(s) described by scope are selected or deselected according to the action parameter. If action equals 0, the objects described by scope are deselected. If action equals 1, the objects described by scope are selected. If action equals 2, the current state of the objects are toggled, that is, selected becomes deselected and vice-versa.

Objects outside of scope are not affected by CT SELECT. That is, objects that are already selected in Area and are not specified by scope remain selected.

Example

This example deselects all objects in the document and then selects the object with an ID number equal to 1.

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

See Also

CT Get ID.

CT MOVE (area; scope; newLeft; newRight)

Parameter	Type		Description
area	Longint	→	4D Chart area
scope	Longint	→	Scope of the command -1 = All 0 = Selected objects >0 = Object ID
newLeft	Real	→	New coordinate of the left edge
newRight	Real	→	New coordinate of the right edge

Description

The CT MOVE command repositions the objects in area described by scope.

- If scope equals -1, CT MOVE repositions all objects in the document.
- If scope equals 0, CT MOVE repositions the selected objects.
- If scope is greater than 0, it must be equal to a specific object's ID and that object is repositioned.

The object(s) described by scope are moved according to the newLeft and newTop parameters, specified as offsets from the current origin.

Example

This example moves the selected objects to the top left corner of the chart area.

⇒ ***CT MOVE*** (Area;0;0;0)

See Also

CT Get ID, CT SIZE.

CT SIZE (area; scope; width; height)

Parameter	Type		Description
area	Longint	→	4D Chart area
scope	Longint	→	Scope of the command -1 = All 0 = Selected objects >0 = Object ID
width	Real	→	New width (in points) -1 = No change
height	Real	→	New height (in points) -1 = No change

Description

The CT SIZE command resizes the objects described in area by scope. When you resize an object, the top left corner of the object is anchored.

- If scope equals -1, CT SIZE resizes all objects in the document.
- If scope equals 0, CT SIZE resizes the selected objects.
- If scope is greater than 0, it must be equal to a specific object's ID and that object is resized.

The object(s) described by scope are resized according to the width and height parameters, specified in points.

Example

This example resizes the object with ID equal to 5.

⇒ ***CT AREA*** (Area;5;10;10)

See Also

CT Get ID, CT MOVE.

CT Count

CT Objects

version 6.0.5

CT Count (area; scope) → Integer

Parameter	Type		Description
area	Longint	→	4D Chart area
scope	Longint	→	-1 = All 0 = Selected >0 = Group ID
Function result	Integer	←	Number of objects in the area

Description

CT Count returns the number of objects in the area described by scope.

• If scope equals -1, CT Count returns the number of objects in the document that are not in a group. Groups appear as a single object.
• If scope equals 0, CT Count returns the number of currently selected that are not in a group. Groups appear as a single object.
• If scope is greater than 0, it must be the ID for a group and CT Count returns the number of objects inside the group. With this syntax you can get information about objects in a group without ungrouping. You can examine nested groups by using subsequent calls to CT Count.

Example

This example opens an alert box that displays the number of currently selected objects.

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

See Also

CT Get ID.

CT ALIGN (area; scope; horizontal; vertical)

Parameter	Type		Description
area	Longint	→	4D Chart area
scope	Longint	→	-1 = All 0 = Selected
horizontal	Integer	→	0 = None 1 = Left 2 = Middle 3 = Right
vertical	Integer	→	0 = None 1 = Top 2 = Middle 3 = Bottom

Description

The CT ALIGN command aligns the objects in area described by scope.

- If scope equals -1, CT ALIGN aligns all objects in the document.
- If scope equals 0, CT ALIGN aligns the selected objects.

The object(s) described by scope are aligned according to the horizontal and vertical parameters.
Following are the values for the Horizontal parameter, a description of their effects, and the corresponding icons in the Align Objects dialog box

- The following table shows the horizontal parameter values and their effects:

Value	Aligns
0	No horizontal alignment
1	Left sides
2	Centers
3	Right sides

- The following table shows the vertical parameter values and their effects:

Example

This example centers the selected objects both horizontally and vertically.

⇒ ***CT ALIGN*** (Area;0;2;2)

See Also

CT Get ID.

CT GET HIGHLIGHT

CT Objects

version 6.0.5

CT GET HIGHLIGHT (area; first; last)

Parameter	Type		Description
area	Longint	→	4D Chart area
first	Integer	←	Receives position of first character minus 1
last	Integer	←	Receives position of last character

Description

The CT GET HIGHLIGHT command returns into the first and last variables the character positions of the highlighted text in area.

first is one less than the first character position highlighted and last is the last character highlighted. If first equals last, no characters are highlighted and the insertion point is between first and first +1.

Since only one object can have highlighted text at a time, the scope parameter is not needed. If there is no highlighted text in area, CT GET HIGHLIGHT returns -32000 for first and last.

Examples

This example returns the position of the highlighted text, and if no text is selected, it alerts the user.

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

See Also

CT SET HIGHLIGHT.

CT SET HIGHLIGHT

CT Objects

version 6.0.5

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

Parameter	Type		Description
area	Longint	→	4D Chart area
scope	Longint	→	-1 = First object in a document 0 = First object in selection >0 = Object ID
first	Integer	→	Position of first character minus 1
last	Integer	→	Position of last character

Description

The CT SET HIGHLIGHT command highlights characters within the text object in area described by scope.

• If scope equals -1, CT SET HIGHLIGHT highlights characters in the first object of the document.

• If scope equals 0, CT SET HIGHLIGHT highlights characters in the first selected object.

• If scope is greater than 0, it must be equal to a specific text object's ID and characters within that text object are highlighted. If the object does not exist, CT SET HIGHLIGHT does nothing. CT SET HIGHLIGHT causes the object described by scope to become the only object selected in area.

If the object described by scope is not a text object, CT SET HIGHLIGHT does nothing.

first and last determine which characters are highlighted. first is one less than the first character position to be highlighted. last is the last character position to be highlighted. If first equals last, no characters are selected and the insertion point is between first and first +1. If last is greater than the number of characters in the text object, CT SET HIGHLIGHT highlights characters to the end of the text object.

CT SET HIGHLIGHT does not highlight only part of a reference. If any portion of a reference is highlighted, CT SET HIGHLIGHT adjusts the highlight to include the entire reference.

Example

This example gets the text of the selected text object and looks for the name “4th Dimension”. If “4th Dimension” is found, it is highlighted and then made bold.

```
      $Find := Position ("4th Dimension";$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
```

See Also

CT GET HIGHLIGHT.

CT INSERT FIELD (area; scope; first; last; table; field{; format})

Parameter	Type		Description
area	Longint	→	4D Chart area
scope	Longint	→	-1 = First object in document 0 = First object in selection >0 = Object ID
first	Integer	→	Position of first character minus 1
last	Integer	→	Position of last character
table	Integer	→	Table number of reference
field	Integer	→	Field number of reference
format	String	→	Format of reference

Description

The CT INSERT FIELD command inserts a field reference into the text object in area described by scope.

• If scope equals -1, CT INSERT FIELD inserts the reference into the first object of the document.

• If scope equals 0, CT INSERT FIELD inserts the reference into the first selected object.

• If scope is greater than 0, it must be equal to a specific text object's ID and the reference is inserted within that text object. If the object does not exist, CT INSERT FIELD does nothing.

If the object described by scope is not a text object, CT INSERT FIELD does nothing.

first and last determine where the reference is inserted. first is one less than the first character position to be replaced and last is the last character position to be replaced. If first equals last, no characters are replaced and the reference is inserted between first and first +1. If last is greater than the number of characters in the text object, CT INSERT FIELD replaces characters from first to the last character in the text object.

table and field determine which field is referenced. table is the number of the table and field is the number of the field. Tables and fields are numbered in the order in which they were created.

If format is a one- or two-digit string, then the format applied to field is from the list. If format is not a one- or two-digit string, then it is compared to the text values of each format in the list. If it matches one of the values in the list, that format is applied. This means that you can refer to the first date format as either “19” or “Short”.

If format is not in the list of formats, it is interpreted as a custom numeric format. If format is inappropriate for the resulting value of the reference, it is ignored. For instance, if you use a date format on a number, the number appears unformatted.

Examples

1. This example inserts a reference to the first field of the first table into the text object that has an ID of 1, replacing any text in the object, and then formats it according to the eleventh format in the list.

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

2. You can use 4th Dimension’s Field and Table functions to determine a field or table’s number. This can make your code easier to read. For example, if the field used in the previous example is [Customers]Name, the code would look like this:

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

See Also

CT INSERT EXPRESSION.

CT INSERT EXPRESSION

CT Objects

version 6.0.5

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

Parameter	Type		Description
area	Longint	→	4D Chart area
scope	Longint	→	-1 = First object 0 = First object in selection >0 = Object ID
first	Integer	→	Position of first character minus 1
last	Integer	→	Position of last character
expression	String	→	Expression
format	String	→	Format of expression

Description

The CT INSERT EXPRESSION command inserts a reference to expression into the text object in the area described by scope.

• If scope equals -1, CT INSERT EXPRESSION inserts the reference into the first object of the document.

• If scope equals 0, CT INSERT EXPRESSION inserts the reference into the first selected object.

• If scope is greater than 0, it must be equal to a specific text object's ID and the reference is inserted within that text object. If the object does not exist, CT INSERT EXPRESSION does nothing.

If the object described by scope is not a text object, CT INSERT EXPRESSION does nothing.

first and last determine where the reference is inserted. first is one less than the first character position to be replaced and last is the last character position to be replaced. If first equals last, no characters are replaced and the reference is inserted between first and first +1. If last is greater than the number of characters in the text object, CT INSERT EXPRESSION replaces characters from first to the end.

expression is the text equivalent of any valid 4th Dimension expression that returns a value. expression can be a reference to any of the following: a field, a variable, a 4th Dimension function, a user-defined function (project method), a plug-in function,

The following table gives examples for each expression type:

Example	Type
[Drawings]Object	Field
vCriteria	Variable
Current Date	4th Dimension function
GetNum	User defined function (project method)
CT Count	4D Chart function
3 * "Hello"	Statement

The optional format parameter is the display format for the reference. This parameter is equivalent to choosing a format from the Format dialog box. Formats are referred to either by their number or their name. Formats are numbered in the order in which they appear in the list of the Format dialog box.

If format is a one or two digit string, then the format applied to field is from the list. If format is not a one or two digit string, then it is compared to the text values of each format in the list. If it matches one of the values in the list, that format is applied. This means that you can refer to the first date format as either “19” or “Short”.

If format is not in the list of formats, it is interpreted as a custom numeric format. If format is inappropriate for the resulting value of the reference, it is ignored. For instance, if you use a date format on a number, the number appears unformatted.

Example

This example creates a new text object, fills it with a reference to the 4th Dimension function Current date and formats it using the Long date format.

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

See Also

CT INSERT FIELD.

4 CT Area

CT AREA TO AREA

CT Area

version 6.0.5

CT AREA TO AREA (source; destination; copyCode)

Parameter	Type		Description
source	Longint	→	Source 4D Chart area
destination	Longint	←	Receives destination 4D Chart area
copyCode	Integer	→	Items to copy 1 = Settings 2 = Objects 3 = Both

Description

The CT AREA TO AREA command copies the contents of the 4D Chart area source into the 4D Chart area destination.

The contents to be transferred are based on the copyCode parameter.

- If copyCode equals 1, the document settings such as display options are transferred.
- If copyCode equals 2, all objects in source are transferred to destination.
- If copyCode equals 3, both objects and document settings are transferred to destination.

When document settings are transferred, they replace the document settings in destination. When objects are transferred they are appended to the objects in destination. CT AREA TO AREA is especially useful for manipulating offscreen areas.

Example

This example copies the contents of the 4D Chart area SalesChart into a new offscreen area.

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

See Also

CT DELETE OFFSCREEN AREA, CT New offscreen area.

CT AREA TO FIELD

CT Area

version 6.0.5

CT AREA TO FIELD (area; scope; table; field{; saveOption})

Parameter	Type		Description
area	Longint	→	4D Chart area
scope	Longint	→	Scope of the command -2 = Document -1 = All 0 = Selected objects >0 = Object ID
table	Integer	→	Table number
field	Integer	→	Field number
saveOption	Integer	→	Method of saving the contents of area 1 = Picture only 2 = Data only 3 = Picture and data -1 = No change

Description

The CT AREA TO FIELD command copies the contents of area into the Picture field specified by table and field. field must be a Picture field.

CT AREA TO FIELD is useful when you want to store objects in a field of a related table or store only specific objects. CT AREA TO FIELD simply assigns the objects to Field. The record in table must still be saved.

scope controls what is copied.

The optional saveOption parameter determines how the document in the 4D Chart area is saved.

• If saveOption equals 1, only the picture (PICT) is saved. Objects can no longer be manipulated individually.

• If saveOption equals 2, only the data concerning the objects in the 4D Chart area is saved. The image is later rebuilt using the information in the saved data. This save option is the quickest and uses the least amount of memory. If there is not enough

Example

This example creates a related record for an object in area.

```
    `Create a record to store the object
CREATE RECORD ([Objects])
    `Assign the relating value
[Objects]Key:=[Charts]Name
    `Get the object's ID
$Temp:=CT Get ID (Area;-1;3)
    `Copy the object into the record
⇒   CT AREA TO FIELD (Area;$Temp;3;2;1)
    `Save the record
SAVE RECORD ([Objects])
```

See Also

CT FIELD TO AREA.

CT FIELD TO AREA

CT Area

version 6.0.5

CT FIELD TO AREA (area; table; field)

Parameter	Type		Description
area	Longint	→	4D Chart area
table	Integer	→	Table number
field	Integer	→	Field number

Description

The CT FIELD TO AREA command places into area the document contained in the Picture field specified by table and field.

The picture value is taken from the current record of table.

field must be of type Picture. field may contain either a previously saved 4D Chart document or a picture. The contents of field replaces the contents of area. If field is empty, the command is ignored.

Example

This form method opens the 4D Chart document contained in the fifth field of the second table.

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

See Also

CT AREA TO FIELD.

CT Area to picture

CT Area

version 6.0.5

CT Area to picture (area; scope) → PIcture

Parameter	Type		Description
area	Longint	→	4D Chart area
scope	Longint	→	Scope of the function -2 = Document -1 = All 0 = Selected objects >0 = Object ID
Function result	PIcture	←	4th Dimension picture of the objects in area

Description

CT Area to picture returns a 4th Dimension picture of the objects in area.

The objects included in the picture are controlled by the scope parameter.

• If scope equals -2, the entire document is copied. This includes document settings such as display options.

• If scope equals -1, all objects in area are copied but without document settings.

• If scope equals 0, only the selected objects are copied.

• If scope is greater than 0, it must equal a specific object's ID and only that object is copied.

Example

This example opens a new offscreen area, creates a bar chart from existing arrays, stores the chart in a picture variable, and finally, deletes the offscreen area.

```
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)
```

See Also

CT Get ID, CT PICTURE TO CLIPBOARD, CT PLACE PICTURE.

CT PICTURE TO AREA

CT Area

version 6.0.5

CT PICTURE TO AREA (area; picture)

Parameter	Type		Description
area	Longint	→	4D Chart area
picture	Picture	→	Picture

Description

The CT PICTURE TO AREA command places into area the document contained in picture.

picture must be a valid 4th Dimension picture expression. The contents of area are replaced with picture. If picture is empty, the command is ignored.

Example

This object method allows the user to copy a graph from a Picture field to a 4D Chart area.

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

See Also

CT Area to picture.

CT NEW DOCUMENT

CT Area

version 6.0.5

CT NEW DOCUMENT (area)

Parameter	Type		Description
area	Longint	→	4D Chart area

Description

The CT NEW DOCUMENT command clears the contents of the document in area. CT NEW DOCUMENT is equivalent to choosing New from the File menu, except that no confirmation dialog box is presented. CT NEW DOCUMENT clears all objects and all document settings such as document size and ruler scaling.

Warning: When you use this command, the current document in area is not saved. If you want to save the current document, you must call CT SAVE DOCUMENT before calling CT NEW DOCUMENT.

Example

This example clears the document in area.

⇒ ***CT NEW DOCUMENT*** (Area)

See Also

CT SAVE DOCUMENT.

CT OPEN DOCUMENT

CT Area

version 6.0.5

CT OPEN DOCUMENT (area; document{; mode})

Parameter	Type		Description
area	Longint	→	4D Chart area
document	String	→	Name of the document, with path 255 characters maximum
mode	Integer	→	Replace document or append to document 0 = Replace 1 = Append to document

Description

The CT OPEN DOCUMENT command opens document and places its contents into area.

If document is an empty string, CT OPEN DOCUMENT displays a standard open-file dialog box, enabling the user to choose the document.

Otherwise, it attempts to open the specified document. If document does not exist, the contents of area remain unchanged and CT Error returns a system error code.

4D Chart expects to find document in the directory that contains the database structure. If you want to open a document outside this directory, specify a complete pathname. See the *4th Dimension Language Reference* for a brief description of pathnames. If document is already open, CT Error returns a system error.

The optional mode parameter controls how the document is opened. mode is used only when document is not an empty string and is not a 4D Chart document. If mode equals 0 or is not specified, document replaces the contents of area. If mode equals 1, document is combined with the current contents of area.

Example

This example opens a different document based on the value of the Client Type field.

```
Case of
        `If the type is Distributor
   : ([Client]Client Type="Distributor")
        `Open the document
```

```
            `If the type is FinalClient
         :([Client]Client  Type="FinalClient")
            `Open this document
⇒        CT OPEN DOCUMENT (Area;"FinalClient")
      End case
```

See Also

CT SAVE DOCUMENT.

CT SAVE DOCUMENT

CT Area

version 6.0.5

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

Parameter	Type		Description
area	Longint	→	4D Chart area
document	Text	→	Name of document, with path
type	String	→	Type of document
scope	Longint	→	Scope of the command 0 = All objects 1 = Selected objects

Description

The CT SAVE DOCUMENT command saves the contents of area in document.

If document is an empty string, CT SAVE DOCUMENT displays the standard save-file dialog box, enabling the user to specify the document name, type, and scope. If document is not an empty string, CT SAVE DOCUMENT saves document with the type type.

If document does not exist, CT SAVE DOCUMENT creates it. If document exists, CT SAVE DOCUMENT overwrites it.

If type is an empty string, a standard 4D Chart document is created. To save the document as a PICT, type should be "PICT".

The optional scope parameter controls what is saved in document. Use scope only when document is not an empty string and when saving a document as a PICT.

By default, document is saved in the directory that contains the database structure. If you want to save a document outside of this directory, specify a complete pathname. See the *4th Dimension Language Reference* for a brief description of pathnames.

Example

This example saves a 4D Chart document in a document with the same name as the company, followed by the year.

```
                `Request the year number
      $Year := Request ("For what year?")
                `If the request is validated
      If (OK=1)
                `Concatenate the document name
            $SaveName := [Company]Name+" " + $Year
                `Save the document
⇒           CT SAVE DOCUMENT (Area;$SaveName;"")
      End if
```

See Also

CT OPEN DOCUMENT.

CT New offscreen area

CT Area

version 6.0.5

CT New offscreen area → Longint

Parameter	Type		Description
This command does not require any parameters			
Function result	Longint	←	4D Chart offscreen area's ID

Description

CT New offscreen area creates a 4D Chart offscreen area and returns the area's ID. The value returned by CT New offscreen area can be used in any 4D Chart command that requires a 4D Chart area.

Example

This example searches for a record, creates an offscreen area, copies a document from the record into the area, and then prints the area.

```
    `Search for the record
QUERY ([Table3];[Table3]Field1 = "Level1")
    `Create a new offscreen area
⇒   $Offscreen := CT New offscreen area
    `Copy the document stored in a field
CT FIELD TO AREA ($Offscreen;3;2)
    `Print the area
CT PRINT ($Offscreen;0)
    `Get rid of the offscreen area
CT DELETE OFFSCREEN AREA ($Offscreen)
```

See Also

CT AREA TO AREA, CT DELETE OFFSCREEN AREA.

CT DELETE OFFSCREEN AREA

CT Area

version 6.0.5

CT DELETE OFFSCREEN AREA (area)

Parameter	Type		Description
area	Longint	→	4D Chart area

Description

The CT DELETE OFFSCREEN AREA command disposes of a 4D Chart offscreen area that was created with CT New offscreen area and frees the memory used.

area must be an offscreen area rather than an area on a form or in a window. You should always call CT DELETE OFFSCREEN AREA when you are finished with an offscreen area.

Example

This example illustrates how to pair every call to CT New offscreen area with a corresponding call to CT DELETE OFFSCREEN AREA.

```
    `Create a new offscreen area
$NewArea := CT New offscreen area
    `Do some processing here
    `Remove the offscreen area
⇒   CT DELETE OFFSCREEN AREA ($NewArea)
```

See Also

CT AREA TO AREA, CT New offscreen area.

CT GET AREA BOUNDARY

CT Area

version 6.0.5

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

Parameter	Type		Description
area	Longint	→	4D Chart area
boundaryCode	Integer	→	Boundary code 0 = Document boundary 1 = Clipped boundary
left	Real	←	Receives left boundary of area
top	Real	←	Receives top boundary of area
right	Real	←	Receives right boundary of area
bottom	Real	←	Receives bottom boundary of area

Description

The CT GET AREA BOUNDARY command returns into the left, top, right, and bottom variables the coordinates of the area rectangle.

• If boundaryCode is 0, CT GET AREA BOUNDARY returns the boundary for the whole document.
• If boundaryCode is 1, CT GET AREA BOUNDARY returns the boundary for the 4D Chart area on a form or for the current size of the 4D Chart plug-in window

Example

This example creates a geometric object composed of several lines in an existing chart area, gets the area's boundary coordinates, and centers the object in the area.

```
      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)
```

See Also

No reference.

5 CT Chart

CT Chart arrays

version 6.0.5

CT Chart arrays (area; type; size; categoryArray; seriesArray; valuesArray) → Number

Parameter	Type		Description
area	Longint	→	4D Chart area
type	Integer	→	Type of graph (see codes below)
size	Integer	→	Option for initial size of graph 1 = Variable 2 = Relative to window (Auto-Variable) 3 = Relative to graph (Auto-Document)
categoryArray	Array	→	Array of categories
seriesArray	Array	→	Array of series
valuesArray	Array	→	Array of values
Function result	Number	←	Object ID number

Description

CT Chart arrays creates a graph based on the specified arrays and returns the graph's Object ID. This command can be used to create either a two-dimensional or three-dimensional graph.

The following table lists the codes for the type parameter.

Code	Chart Type
1	Area
2	Column
3	Picture
4	Line
5	Scatter
6	Pie
7	Polar
8	2D XY
100	3D Column
101	3D Line
102	3D Area
103	3D Surface
104	3D Triangle

The size parameter determines how much space the graph fills when it is generated and how the graph size changes when you resize the window:

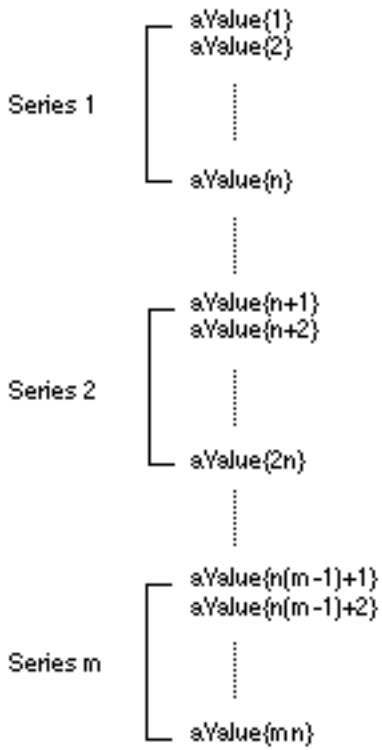
• If you pass 1 (size **Variable**), the graph will fill the 4D chart area or the window. It will keep this size until you modify it using the handles.
• If you pass 2 (size **Relative to the window**), the graph will fill the 4D chart area or the external window. If you later change the size of the window, the graph will adapt automatically. However, once you modify the size of the graph using the handles, this change will not take place anymore.
• If you pass 3 (size **Relative to the document**), the graph will adapt to the dimensions of the page that you selected in the Page Setup dialog. It will keep this size until you modify the graph size using the handles.

The categoryArray parameter contains the X-axis categories.

seriesArray contains the series. In a two-dimensional graph, the series is displayed on the Category axis. In a three-dimensional graph, the series is displayed on the Series axis.

valuesArray is a on-edimensional array that contains all the values to be graphed on the Values axis. valuesArray must be filled in so that there is a value for each element of the categoryArray and seriesArray. That is, if there are n categories and m series, there will be n*m elements in valuesArray.

The following illustration shows the order in which valuesArray should be filled. n represents the total number of categories. m represents the total number of series.



For example, consider the following data and resulting Values array:

School (Categories)	Year (Series)	Students (Values)	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

See Also

CT Chart data, CT Chart selection.

CT Chart selection

CT Chart

version 6.0.5

CT Chart selection (area; type; size; groupCategory; table; categoryField; series/valuesFields) → Longint

Parameter	Type		Description
area	Longint	→	4D Chart area
type	Integer	→	Type of graph (see codes below)
size	Integer	→	Option for initial size of graph 1 = Variable 2 = Relative to window (Auto-Variable) 3 = Relative to document (Auto-Document)
groupCategory	Integer	→	Group the Category data? 0 = No 1 = Yes
table	Integer	→	Number of the table to graph
categoryField	Integer	→	Number of the field to plot on the Category axis
series/valuesFields	Integer array	→	Array of field numbers
Function result	Longint	←	Object ID of the chart

Description

The command CT Chart selection creates a graph of the current selection of records of table. The function returns the chart's Object ID.

The following table lists the codes for the type parameter.

Code	Chart Type
1	Area
2	Column
3	Picture
4	Line
5	Scatter
6	Pie
7	Polar
8	2D XY
100	3D Column

The size parameter determines how much space the graph fills when it is generated and how the graph size changes when you resize the window:

• If you pass 1 (size **Variable**), the graph will fill the 4D chart area or the window. It will keep this size until you modify it using the handles.
• If you pass 2 (size **Relative to the window**), the graph will fill the 4D chart area or the external window. If you later change the size of the window, the graph will adapt automatically. However, once you modify the size of the graph using the handles, this change will not take place anymore.
• If you pass 3 (size **Relative to the document**), the graph will adapt to the dimensions of the page that you selected in the Page Setup dialog. It will keep this size until you modify the graph size using the handles.

GroupCategory specifies whether the data on the Category axis should be grouped.

• If groupCategory = 1, then each category will be unique, and values for any duplicate categories will be summed.
• If groupCategory = 0, then the values for every category will be graphed separately.

Note: There is no need to group series, because the series are field names and are therefore unique.

table is the number of the table from which to graph data. You can retrieve the number of a table by passing a pointer to the table as a parameter to the Table function.

categoryField is the number of the field to graph on the Category axis. You can retrieve the number of a field by passing a pointer to the field as a parameter to the Field function.

series/valuesFields is an array of field numbers of the fields to graph as the series and values. The field names become the series; the values stored in the fields are graphed on the Values axis. In a two-dimensional graph, the series are displayed on the Category axis. In a three-dimensional graph, the series are displayed on the Series axis.

See Also

CT Chart arrays, CT Chart data.

CT Chart data

CT Chart

version 6.0.5

CT Chart data (area; type; size; groupCategory; groupSeries; table; categoryField; seriesField; valuesField) → Longint

Parameter	Type		Description
area	Longint	→	4D Chart area
type	Integer	→	Type of graph (see codes below)
size	Integer	→	Option for initial size of graph 1 = Variable 2 = Relative to window (Auto-Variable) 3 = Relative to graph (Auto-Document)
groupCategory	Integer	→	Group the Category data? 0 = No 1 = Yes
groupSeries	Integer	→	Group the Series data? 0 = No 1 = Yes
table	Integer	→	Number of the table from which to graph data
categoryField	Integer	→	Number of the field to plot on the Category axis
seriesField	Integer	→	Number of the field to plot as the series
valuesField	Integer	→	Number of the field to plot on the Values axis
Function result	Longint	←	Object ID of the chart

Description

CT Chart data creates a graph of the current selection of records of table. The function returns the chart's Object ID.

The following table lists the codes for the type parameter.

Code	Chart Type
1	Area
2	Column
3	Picture
4	Line
5	Scatter
6	Pie
7	Polar
8	2D XY
100	3D Column
101	3D Line
102	3D Area
103	3D Surface
104	3D Triangle
105	3D Spike

The size parameter determines how much space the graph fills when it is generated and how the graph size changes when you resize the window:

• If you pass 1 (size **Variable**), the graph will fill the 4D chart area or the window. It will keep this size until you modify it using the handles.
• If you pass 2 (size **Relative to the window**), the graph will fill the 4D chart area or the external window. If you later change the size of the window, the graph will adapt automatically. However, once you modify the size of the graph using the handles, this change will not take place anymore.
• If you pass 3 (size **Relative to the document**), the graph will adapt to the dimensions of the page that you selected in the Page Setup dialog. It will keep this size until you modify the graph size using the handles.

groupCategory specifies whether or not the data on the Category axis should be grouped.

• If groupCategory is equal to 1, then each category will be unique, and values for any duplicate categories will be summed.
• If groupCategory is equal to 0, then the values for every category will be graphed separately.

groupSeries specifies whether or not the data on the Category axis should be grouped.

• If groupSeries is equal to 1, then each category will be unique, and values for any duplicate categories will be summed.
• If groupSeries is equal to 0, then the values for every category will be graphed separately.

table is the number of the table to graph. You can retrieve the number of a table by passing a pointer to the table as a parameter to the Table function.

categoryField is the number of the field to graph on the Category axis.

seriesField is the number of the field to graph on the Series axis. In a two-dimensional graph, the series are displayed on the Category axis. In a three-dimensional graph, the series are displayed on the Series axis.

valuesField is the number of the field to graph on the Values axis.

You can retrieve the number of a field by passing a pointer to the field as a parameter to the Field function.

See Also

CT Chart arrays, CT Chart selection.

CT GET CHART FILL ATTRIBUTES

CT Chart

version 6.0.5

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

Parameter	Type		Description
area	Longint	→	4D Chart area
object	Longint	→	Object ID
partType	Integer	→	Type of object for which to get attributes
partSpecifics	Longint	→	Specific part of object for which to get attributes
pattern	Integer	←	Receives pattern number (1 to 36)
color	Longint	←	Receives color value

Description

The CT GET CHART FILL ATTRIBUTES command gets the fill attributes of the chart object specified by area, object, partType, and partSpecifics.

partType and partSpecifics specify the part of the graph for which to get the attributes. The codes for these parameters are listed in the section Parameter Codes.

pattern is an integer from 1 to 36 that specifies one of the patterns available on the Pattern palette. Codes for the pattern parameter are listed in the section Parameter Codes.

color is a long integer that specifies the color of the object. You can specify a value for the color parameter by using the CT Index to color or the CT RGB to color functions.

Note: To get the attributes of objects added to the document using the drawing tools or the drawing functions, use the commands in the Object theme.

Example

This example returns the fill attributes of the first series of the chart specified by $ChartID in the $Pattern and $Color parameters.

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

See Also

CT SET CHART FILL ATTRIBUTES.

CT SET CHART FILL ATTRIBUTES

CT Chart

version 6.0.5

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

Parameter	Type		Description
area	Longint	→	4D Chart area
object	Longint	→	Object ID
partType	Integer	→	Type of object for which to set attributes
partSpecifics	Longint	→	Specific part of object for which to set attributes
pattern	Integer	→	Pattern number (1 to 36) -1 = No change
color	Longint	→	Color value -1 = No change

Description

The CT SET CHART FILL ATTRIBUTES command sets the fill attributes of the chart object specified by area, object, partType, and partSpecifics.

partType and partSpecifics specify the exact part of the graph for which to get the attributes. The codes for these parameters are listed in the section Parameter Codes.

pattern is an integer from 1 to 36 that specifies one of the patterns available on the Pattern palette. Codes for this parameter are listed in the section Parameter Codes.

color is a long integer that specifies the color of the object. You can specify a value for the color parameter by using the CT Index to color or the CT RGB to color functions. For more information about these functions, refer to the commands in the CT Utilities section.

Note: To set the attributes of objects added to the document using the drawing tools or the drawing functions, use the commands in the Object theme.

Example

This example changes the first series fill attributes for the chart specified by $ChartID. The color is set to Red, and the pattern is set to Solid.

⇒ ***CT SET CHART FILL ATTRIBUTES*** (Area;$ChartID;8;100;3;***CT Index to color*** (4))

See Also

CT GET CHART FILL ATTRIBUTES.

CT GET CHART TEXT ATTRIBUTES

CT Chart

version 6.0.5

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

Parameter	Type		Description
area	Longint	→	4D Chart area
object	Longint	→	Object ID
partType	Integer	→	Type of object for which to get attributes
partSpecifics	Longint	→	Specific part of object for which to get attributes
fontID	Integer	←	Receives the font ID
fontSize	Integer	←	Receives the font size
style	Integer	←	Receives font style
color	Longint	←	Receives the color value

Description

The CT GET CHART TEXT ATTRIBUTES command gets the attributes of the chart text specified by area, object, partType, and partSpecifics.

partType and partSpecifics specify the exact part of the graph for which to get the attributes. The codes for these parameters are listed in the section Parameter Codes.

fontID is the ID of the font in your system. You can get the ID number of a font using the CT Font number function.

fontSize is the size in points of the highlighted text or text object(s).

style is a composite number that results from the addition of several style numbers. The following table lists the style numbers:

Value	Style
0	Plain
1	Bold
2	Italic
4	Underline
8	Outlined (Macintosh only)
16	Shadowed (Macintosh only)

Note: To get the attributes of text added to the document using the Text tool or the CT Draw text function, use the commands in the Object theme.

Example

This example checks to see whether the text attributes of the Category axis title have been customized, and if so, resets them.

```
⇒    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
```

See Also

CT SET CHART TEXT ATTRIBUTES.

CT SET CHART TEXT ATTRIBUTES

CT Chart

version 6.0.5

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

Parameter	Type		Description
area	Longint	→	4D Chart area
object	Longint	→	Object ID
partType	Integer	→	Type of object for which to get attributes
partSpecifics	Longint	→	Specific part of object for which to get attributes
fontID	Integer	→	Font ID -1 = No change
fontSize	Integer	→	Font size -1 = No change
style	Integer	→	Code for font style -1 = No change
color	Longint	→	Color value -1 = No change

Description

The CT SET CHART TEXT ATTRIBUTES command sets the attributes of the chart text specified by area, object, partType, and partSpecifics.

partType and partSpecifics specify the exact part of the graph for which to get the attributes. The codes for these parameters are listed in section Parameter Codes.

fontID is the ID of the font in your system. You can get the ID number of a font using the CT Font number function.

fontSize is the size in points of the highlighted text or text object(s).

style is a composite number that results from the addition of several style numbers. The following table lists the style numbers:

Value	Style
0	Plain
1	Bold

color is a long integer that specifies the color of the object. You can specify a value for the color parameter by using the CT Index to color or the CT RGB to color functions.

See the example for the CT GET CHART TEXT ATTRIBUTES command.

Note: To set the attributes of text added to the document using the Text tool or the CT Draw text function, use the commands in the Object theme.

See Also

CT GET CHART TEXT ATTRIBUTES.

CT GET CHART LINE ATTRIBUTES

CT Chart

version 6.0.5

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

Parameter	Type		Description
area	Longint	→	4D Chart area
object	Longint	→	Object ID
partType	Integer	→	Type of object for which to get attributes
partSpecifics	Longint	→	Specific part of object for which to get attributes
pattern	Integer	←	Receives pattern number (1 to 36)
color	Longint	←	Receives color value (>= 0)
lineWidth	Real	←	Receives line thickness in points (>= 0)

Description

The CT GET CHART LINE ATTRIBUTES command returns the attributes of the specified line in the pattern, color, and lineWidth parameters.

The line for which the attributes should be retrieved are specified by area, object, partType, and partSpecifics.

partType and partSpecifics specify the part of the graph for which to get the attributes. The codes for these parameters are listed in the section Parameter Codes.

pattern is an integer from 1 to 36 that specifies one of the patterns available on the Pattern palette. The codes for the pattern parameter are listed in the section Parameter Codes.

color is a long integer that specifies the color of the object. You can specify a value for the color parameter by using the CT Index to color or the CT RGB to color functions.

lineWidth is the thickness of the line, measured in points.

Note: To get the attributes of lines added to the document using the Line tool or the CT Draw line function, use the commands in the Object theme.

Example

This example returns the plot rectangle's line attributes for the chart specified by $ChartID in the $Pattern, $Color, and $Line parameters.

⇒ ***CT GET CHART LINE ATTRIBUTE***S (Area;$ChartID;1;0;$Pattern;$Color;$Line)

See Also

CT SET CHART LINE ATTRIBUTES.

CT SET CHART LINE ATTRIBUTES

CT Chart

version 6.0.5

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

Parameter	Type		Description
area	Longint	→	4D Chart area
object	Longint	→	Object ID
partType	Integer	→	Type of object for which to get attributes
partSpecifics	Longint	→	Specific part of object for which to get attributes
pattern	Integer	→	Pattern number (1 to 36) -1 = No change
color	Longint	→	Color value -1 = No change
lineWidth	Real	→	Line thickness in points (>= 0) -1 = No change

Description

The CT SET CHART LINE ATTRIBUTES command sets the attributes of the line specified by area, object, partType, and partSpecifics.

partType and partSpecifics specify the exact part of the graph for which to get the attributes. The codes for these parameters are listed in the section Parameter Codes.

pattern is an integer from 1 to 36 that specifies one of the patterns available on the Pattern palette. Codes for this parameter are listed in the section Parameter Codes.

color is a long integer that specifies the color of the object. You can specify a value for the color parameter by using the CT Index to color or the CT RGB to color functions. For more information about these functions, refer to the commands in the Utilities theme.

lineWidth is the thickness of the line, measured in points.

Note: To set the attributes of lines added to the document using the Line tool or the CT Draw line function, use the commands in the Object theme.

Example

This example changes the plot rectangle's line attributes for the chart specified by $ChartID. The color is changed to green, the line width is set to 3 points, and the pattern is set to solid.

```
⇒ CT SET CHART LINE ATTRIBUTES (Area;$ChartID;1;0;3;
     CT Index to color (10);3)
```

See Also

CT GET CHART LINE ATTRIBUTES.

CT SET FILL ATTRIBUTES

CT Chart

version 6.0.5

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

Parameter	Type		Description
area	Longint	→	4D Chart area
objects	Longint array	→	List of object ID numbers
patterns	Integer array	→	List of pattern numbers
colors	Longint array	→	List of color values

Description

The CT SET FILL ATTRIBUTES command acts the same as CT SET CHART FILL ATTRIBUTES, except that it applies to a list of objects. In the objects parameter, you pass a long integer array containing a list of the ID numbers of objects for which you want to set fill attributes.

The patterns and colors parameters are arrays containing the corresponding attributes.

For more information, refer to the CT SET CHART FILL ATTRIBUTES command.

Example

In this example, you have a form that contains a 4D Chart area named vct. You want to simultaneously create 100 rectangles using specific lines and patterns. Instead of calling the CT SET CHART LINE ATTRIBUTES and CT SET CHART FILL ATTRIBUTES commands 100 times, you fill arrays and define rectangle attributes in one call.

Here is the method for the form:

```
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) `or 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
```

See Also

CT SET CHART FILL ATTRIBUTES, CT SET CHART LINE ATTRIBUTES, CT SET LINE ATTRIBUTES.

CT SET LINE ATTRIBUTES

CT Chart

version 6.0.5

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

Parameter	Type		Description
area	Longint	→	4D Chart area
objects	Longint array	→	List of object ID numbers
patterns	Integer array	→	List of pattern numbers
colors	Longint array	→	List of color values
lineWidths	Real array	→	List of line widths (in points)

Description

The CT SET LINE ATTRIBUTES command acts like the CT SET CHART LINE ATTRIBUTES command, except that it applies to a list of objects. In the objects parameter, you pass a long integer array that contains a list of the ID numbers of objects for which you want to set attributes.

The CT SET LINE ATTRIBUTES command sets the attributes of the line specified by area and objects. The patterns, colors and lineWidths parameters are arrays containing the corresponding attributes.

For more information, refer to the CT SET CHART LINE ATTRIBUTES command.

Example

See the example for the CT SET FILL ATTRIBUTES command.

See Also

CT SET CHART LINE ATTRIBUTES, CT SET FILL ATTRIBUTES.

CT GET VALUE ATTRIBUTES

CT Chart

version 6.0.5

CT GET VALUE ATTRIBUTES (area; object; position; display; orientation; format)

Parameter	Type		Description
area	Longint	→	4D Chart area
object	Longint	→	Object ID
position	Integer	←	Receives position of values -1 = No change 0 = None 1 = Exterior top 2 = Exterior bottom 3 = Interior top 4 = Interior middle 5 = Interior bottom 6 = On the axis 8 = On the left 9 = On the right 10 = Under
display	Integer	←	Receives type of information 1 = Values 2 = Percentage 3 = Category 4 = Value and percentage 5 = Category and percentage
orientation	Integer	←	Receives orientation of values -1 = No change 0 = Standard 1 = Vertical 2 = On the right 3 = On the left
format	Alpha	←	Format for values

Description

The CT GET VALUE ATTRIBUTES command returns the attributes of the graph's values specified by area and object.

position is an integer that specifies the position of the values.

orientation is an integer that specifies the orientation of values.

See Also

No reference.

CT SET VALUE ATTRIBUTES

CT Chart

version 6.0.5

CT SET VALUE ATTRIBUTES (area; object; position; display; orientation; format)

Parameter	Type		Description
area	Longint	→	4D Chart area
object	Longint	→	Object ID
position	Integer	→	Position of values -1 = No change 0 = None 1 = Exterior top 2 = Exterior bottom 3 = Interior top 4 = Interior middle 5 = Interior bottom 6 = On the axis 8 = On the left 9 = On the right 10 = Under
display	Integer	→	Type of information to display 1 = Values 2 = Percentage 3 = Category 4 = Value and percentage 5 = Category and percentage
orientation	Integer	→	Orientation of values -1 = No change 0 = Standard 1 = Vertical 2 = On the right 3 = On the left
format	Alpha	→	Format for values

Description

The CT SET VALUE ATTRIBUTES command modifies the attributes of the graph's values specified by area and object.

position is an integer that specifies the position of the values.

orientation is an integer that specifies the orientation of values.

See Also

No reference.

CT SHOW GRID LINES

CT Chart

version 6.0.5

CT SHOW GRID LINES (area; object; axis; grid; visible)

Parameter	Type		Description
area	Longint	→	4D Chart area
object	Longint	→	Object ID
axis	Integer	→	Axis for which to show/hide grid lines 0 = Category 1 = Series 2 = Values
grid	Integer	→	Grid lines to show/hide 0 = Minor 1 = Major
visible	Integer	→	Hide or show grid lines 0 = Hide 1 = Show

Description

The CT SHOW GRID LINES command displays or hides the major and/or minor grid lines for the axis specified by area, object, and axis.

The grid parameter allows you to specify which grid lines should be affected by the command. Major grid lines are spaced at the major increments; minor grid lines are spaced at the minor increments.

The visible parameter allows you to specify whether or not the specified grid lines are visible.

Example

This example displays the minor grid lines of the Values axis for the chart specified by $ChartID.

```
⇒ CT SHOW GRID LINES (Area;$ChartID;2;0;1)
```

See Also

CT GET CHART LINE ATTRIBUTES, CT SET CHART LINE ATTRIBUTES.

CT GET AXIS ATTRIBUTES

CT Chart

version 6.0.5

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

Parameter	Type		Description
area	Longint	→	4D Chart area
object	Longint	→	Object ID
axis	Integer	→	Chart axis 0 = Category 1 = Series 2 = Values
minorTick	Integer	←	Receives minor tick kind 0 = None 1 = Inside 2 = Outside 3 = Cross
majorTick	Integer	←	Receives major tick kind 0 = None 1 = Inside 2 = Outside 3 = Cross
location	Real	←	Receives axis location (position at which axis is placed)
reverse	Integer	←	Receives reverse order 0 = Not reversed 1 = Reversed

Description

The CT GET AXIS ATTRIBUTES command returns the attributes of the axis specified by area, object, and axis in the variables for the minorTick, majorTick, location, and reverse parameters. This command applies only to two-dimensional graphs.

minorTick and majorTick refer to the tick marks on axis. The tick mark options can be set in the Axis dialog box for each axis or using the CT SET AXIS ATTRIBUTES command.

location refers to the number of the value at which the axis crosses another axis. If axis is a horizontal axis, location is the number of increments from the bottom of the

Example

This example returns the category axis attributes of the chart specified by $ChartID in the $MajorTick, $MinorTick, $Location, and $Reverse parameters.

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

See Also

CT SET AXIS ATTRIBUTES.

CT SET AXIS ATTRIBUTES

CT Chart

version 6.0.5

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

Parameter	Type		Description
area	Longint	→	4D Chart area
object	Longint	→	Object ID
axis	Integer	→	Chart axis 0 = Category 1 = Series 2 = Values
minorTick	Integer	→	Minor tick kind -1 = No change 0 = None 1 = Inside 2 = Outside 3 = Cross
majorTick	Integer	→	Major tick kind -1 = No change 0 = None 1 = Inside 2 = Outside 3 = Cross
location	Real	→	Axis location (position at which the axis is placed)
reverse	Integer	→	Reverse order 0 = Do not reverse 1 = Reverse -1 = No change

Description

The CT SET AXIS ATTRIBUTES command sets the attributes of the axis specified by area, object, and axis. This command applies only to two-dimensional graphs.

minorTick and majorTick refer to the tick marks on axis.

location refers to the number of the value at which the axis crosses another axis. If axis is a horizontal axis, location is the number of increments from the bottom of the

Example

This example changes the Category axis attributes for the chart specified by $ChartID.

⇒ ***CT SET AXIS ATTRIBUTES*** (vArea;$ChartID;0;0;3;160;1)

See Also

CT GET AXIS ATTRIBUTES.

CT GET LABEL ATTRIBUTES

CT Chart

version 6.0.5

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

Parameter	Type		Description
area	Longint	→	4D Chart area
object	Longint	→	Object ID
axis	Integer	→	Chart axis 0 = Category 1 = Series 2 = Values
position	Integer	←	Receives position of label 0 = None 1 = Top 2 = Left 3 = Bottom 4 = Right
orientation	Integer	←	Receives label orientation 0 = Normal 1 = Vertical 2 = Rotated right 3 = Rotated left 4 = Staggered 5 = Wrap around
format	String	←	Receives label format
frequency	Integer	←	Receives label display

Description

The CT GET LABEL ATTRIBUTES command returns in the position, orientation, and format parameters the attributes for the label of the axis specified by area, object, and axis.

position is the position of the axis labels relative to the graph.

orientation is the orientation of each label. The following label orientations are available for each axis:

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

format is the display format for the label text. When the display format is “General”, an empty string "" is returned in the format parameter. For more information on the special characters used in display formats, see the 4th Dimension Design Reference.

The optional parameter frequency receives the label display frequency for the series or categories axis. This parameter returns the number of categories/series of the graph for which only one label is displayed. By default, this parameter returns 1(all labels are displayed). If the axis is the Values axis, frequency returns the value -32000. For information on setting the frequency parameter, see the description of the CT SET LABEL ATTRIBUTES command.

Note: To get the text attributes of axis labels, use the CT GET CHART TEXT ATTRIBUTES command

Example

This example returns the category axis label attributes for the chart specified by $ChartID in the $Position, $Orient, and $Format parameters.

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

See Also

CT GET CHART TEXT ATTRIBUTES, CT SET LABEL ATTRIBUTES.

CT SET LABEL ATTRIBUTES

CT Chart

version 6.0.5

CT SET LABEL ATTRIBUTES (area; object; axis; position; orientation; format{; frequency})

Parameter	Type		Description
area	Longint	→	4D Chart area
object	Longint	→	Object ID
axis	Integer	→	Chart axis 0 = Category 1 = Series 2 = Values
position	Integer	→	Position of label -1 = No change 0 = None 1 = Top 2 = Left 3 = Bottom 4 = Right
orientation	Integer	→	Label orientation -1 = No change 0 = Normal 1 = Vertical 2 = Rotated right 3 = Rotated left 4 = Staggered 5 = Wrap around
format	String	→	Label format
frequency	Integer	→	Label display frequency (1-255)

Description

The CT SET LABEL ATTRIBUTES command changes the attributes of position, orientation, and format for the label of the axis specified by area, object, and axis.

position is the position of the axis labels relative to the graph.

orientation is the orientation of each label. For a table showing each orientation option, see the description of CT GET LABEL ATTRIBUTES.

The optional parameter frequency allows you to set the label display frequency for category or series axes. This parameter is useful if you have a large number of distinct values on a category or series axis and there is not enough room on the graph to print all the labels. By setting frequency to a value greater than 1, you can choose to suppress the printing of a specified percentage of the category or series labels. For example, setting frequency to 3 prints every third label.

frequency must be in the range of 1 to 255. If it is zero, the default value of 1 (print all labels) is used. frequency is ignored if the specified axis is the Values axis. Also, frequency is not used for polar or pie charts. The last label is always printed.

If frequency is set to -1, 4D Chart will automatically set the label display frequency depending on the graph dimensions.

Note: To set the text attributes of axis labels, use the CT SET CHART TEXT ATTRIBUTES command.

Example

This example changes the Category axis label attributes for the chart specified by $ChartID. The code sets the label position to Bottom, the format to General, and the orientation to Rotated Right.

```
⇒ CT SET LABEL ATTRIBUTES (Area;$ChartID;0;3;2;"")
```

See Also

CT GET LABEL ATTRIBUTES, CT SET CHART TEXT ATTRIBUTES.

CT GET TITLE ATTRIBUTES

CT Chart

version 6.0.5

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

Parameter	Type		Description
area	Longint	→	4D Chart area
object	Longint	→	Object ID
axis	Integer	→	Axis for which to get attributes 0 = Category 1 = Series 2 = Values
position	Integer	←	Receives position of title 0 = None 1 = Top 2 = Left 3 = Bottom 4 = Right
orientation	Integer	←	Receives orientation of title 0 = Normal 1 = Vertical 2 = Rotated right 3 = Rotated left
title	String	←	Receives text of title

Description

The CT GET TITLE ATTRIBUTES command returns the position, orientation, and text of the axis title specified by area, object, and axis.

position is the position of the title relative to the graph.

orientation is the orientation of the title.

title is the text of the title. The maximum length of the title is 255 characters.

Note: To get the text attributes of a title, use the CT GET CHART TEXT ATTRIBUTES command.

Example

This example returns Category axis title attributes in the $Position, $Orient, and $Title variables.

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

See Also

CT GET CHART TEXT ATTRIBUTES, CT SET TITLE ATRIBUTES.

CT SET TITLE ATTRIBUTES

CT Chart

version 6.0.5

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

Parameter	Type		Description
area	Longint	→	4D Chart area
object	Longint	→	Object ID
axis	Integer	→	Axis for which to set attributes 0 = Category 1 = Series 2 = Values
position	Integer	→	Position of title -1 = No change 0 = None 1 = Top 2 = Left 3 = Bottom 4 = Right
orientation	Integer	→	Orientation of title -1 = No change 0 = Normal 1 = Vertical 2 = Rotated right 3 = Rotated left
title	String	→	Text of title

Description

The CT SET TITLE ATTRIBUTES command sets the position, orientation, and text of the axis title specified by area, object, and axis.

position is the position of the title relative to the graph.

orientation is the orientation of the title.

title is the text of the title. The maximum length of the title is 255 characters.

Note: To set the text attributes of a title, use the CT SET CHART TEXT ATTRIBUTES command.

Example

This example sets the Values axis title's text, position, and orientation.

⇒ ***CT SET TITLE ATTRIBUTES*** (Area;$ChartID;2;2;3;"Total Rainfall (in inches)")

See Also

CT GET TITLE ATTRIBUTES, CT SET CHART TEXT ATTRIBUTES.

CT GET LEGEND ATTRIBUTES

CT Chart

version 6.0.5

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

Parameter	Type		Description
area	Longint	→	4D Chart area
object	Longint	→	Object ID
display	Integer	←	Is legend displayed? 0 = No 1 = Yes
orientation	Integer	←	Receives orientation of the series in the legend 0 = Horizontal 1 = Vertical
reverseOrder	Integer	←	Using reverse order? 0 = Not reversed 1 = Reversed
reverseKey	Integer	←	Key and text reversed? 0 = Not reversed 1 = Reversed
location	Integer	←	Receives location code
horizOffset	Integer	←	Receives horizontal offset from plot's left side in points
vertOffset	Integer	←	Receives vertical offset from plot's top side in points

Description

The CT GET LEGEND ATTRIBUTES command returns the attributes of the legend specified by area and object in the variables for the display, orientation, reverseOrder, reverseKey, location, horizOffset, and vertOffset parameters.

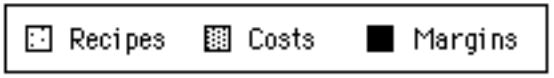
display specifies whether or not the legend is displayed.

orientation specifies whether the series in the legend are displayed vertically or horizontally with regard to each other.

The following are examples of vertical and horizontal legends:

Vertical Legend

Horizontal Legend



reverseOrder specifies whether the order of the series in the legend is reversed.

reverseKey specifies whether the series label and the key that explains the unique pattern and color of the series are reversed. The default is to show the key to the left of the label.

The following table lists the codes for the location parameter:

Code	Location
0	Legend's location is not one of the built-in locations
1	Top left
2	Bottom left
3	Top right
4	Bottom right
5	Left
6	Right
7	Top
8	Bottom

horizOffset and vertOffset are used when location is not one of the built-in legend locations (location = 0). horizOffset is expressed in points from the left side of the graph to the left side of the legend. vertOffset is expressed in points from the top of the graph to the top of the legend.

Note: To get the text attributes of legend text, use the CT GET CHART TEXT ATTRIBUTES command.

Example

This example gets the legend text attributes for the chart specified by $ChartID.

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

CT SET LEGEND ATTRIBUTES

CT Chart

version 6.0.5

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

Parameter	Type		Description
area	Longint	→	4D Chart area
object	Longint	→	Object ID
display	Integer	→	Display legend? 0 = No 1 = Yes
orientation	Integer	→	Orientation of the series in the legend 0 = Horizontal 1 = Vertical
reverseOrder	Integer	→	Reverse order? -1 = No change 0 = Do not reverse 1 = Reverse
reverseKey	Integer	→	Using reverse order? -1 = No change 0 = Do not reverse 1 = Reverse
location	Integer	→	Location code
horizOffset	Integer	→	If location = 0: Horizontal offset from plot's left side in points
vertOffset	Integer	→	If location = 0: Vertical offset from plot's top side in points

Description

The CT SET LEGEND ATTRIBUTES command sets the attributes of the legend specified by area and object.

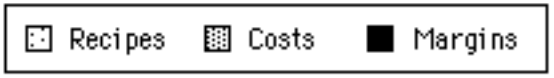
display specifies whether or not the legend is displayed.

orientation specifies whether the series in the legend are displayed vertically or horizontally with regard to each other.

Here are examples of vertical and horizontal legends:

Vertical Legend

Horizontal Legend



reverseOrder specifies whether the order of the series in the legend is reversed.

reverseKey specifies whether the series label and the key that explains the unique pattern and color of the series are reversed. The default is to show the key to the left of the label.

The following table contains the codes for the location parameter:

Code	Location
-1	No change
0	Position legend using horizOffset and vertOffset
1	Top left
2	Bottom left
3	Top right
4	Bottom right
5	Left
6	Right
7	Top
8	Bottom

horizOffset and vertOffset are used when location is set to 0. horizOffset is expressed in points from the left side of the graph to the left side of the legend. vertOffset is expressed in points from the top of the graph to the top of the legend.

Note: To get the text attributes of legend text, use the CT GET CHART TEXT ATTRIBUTES command.

Example

This example displays the legend, centered at the top of the graph.

```
⇒  CT GET LEGEND ATTRIBUTES (Area;$ChartID;1;0;0;0;7;0;0)
```

CT Get legend text

CT Chart

version 6.0.5

CT Get legend text (area; object; legendItem) → Text

Parameter	Type		Description
area	Longint	→	4D Chart area
object	Longint	→	Object ID
legendItem	Integer	→	Legend item number
Function result	Text	←	Text of specified legend item

Description

The CT Get legend text command returns the text of the specified legend item.

legendItem is the number of the series (or category for a pie chart) in the legend. However, if the legend order has been reversed, legendItem reflects the original order, not the reversed order.

Example

This example returns the legend text of the first series in the chart specified by $ChartID.

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

See Also

CT GET LEGEND ATTRIBUTES, CT SET LEGEND ATTRIBUTES, CT SET LEGEND TEXT.

CT SET LEGEND TEXT

CT Chart

version 6.0.5

CT SET LEGEND TEXT (area; object; legendItem; legendtext)

Parameter	Type		Description
area	Longint	→	4D Chart area
object	Longint	→	Object ID
legendItem	Integer	→	Legend item number
legendtext	Text	→	Text of legend item

Description

The CT SET LEGEND TEXT command sets the text of the specified legend item.

legendItem is the number of the series (or category for a pie chart) in the legend. However, if the legend order has been reversed, legendItem reflects the original order, not the reversed order.

Example

This example changes the legend text of the chart specified by $ChartID.

```
      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
```

See Also

CT GET LEGEND ATTRIBUTES, CT Get legend text, CT SET LEGEND ATTRIBUTES.

CT GET REAL SCALE

CT Chart

version 6.0.5

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

Parameter	Type		Description
area	Longint	→	4D Chart area
object	Longint	→	Object ID
minAuto	Integer	←	Using default minimum? 0 = No 1 = Yes
maxAuto	Integer	←	Using default maximum? 0 = No 1 = Yes
majIncrAuto	Integer	←	Using default major increment? 0 = No 1 = Yes
minIncrAuto	Integer	←	Using default minor increment? 0 = No 1 = Yes
minimum	Real	←	Receives minimum value
maximum	Real	←	Receives maximum value
majorIncr	Real	←	Receives major increment
minorIncr	Real	←	Receives minor increment

Description

The CT GET REAL SCALE command returns whether or not the default values are being used and what alternative values have been set for the Values axis scale. CT GET REAL SCALE is used when the values are real numbers, integers, and/or long integers.

minAuto and maxAuto specify whether the graph is currently using the default minimum and maximum values.

majIncrAuto and minIncrAuto specify whether the graph is currently using the default major and minor increments.

minimum and maximum are the minimum and maximum values set by the user in the Axis dialog box or by the database designer using the CT SET REAL SCALE command.

Example

This example returns scale data for the chart specified by $ChartID.

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

See Also

CT GET DATE SCALE, CT SET DATE SCALE, CT SET REAL SCALE.

CT SET REAL SCALE

CT Chart

version 6.0.5

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

Parameter	Type		Description
area	Longint	→	4D Chart area
object	Longint	→	Object ID
minAuto	Integer	→	Use default minimum? -1 = No change 0 = No 1 = Yes
maxAuto	Integer	→	Use default maximum? -1 = No change 0 = No 1 = Yes
majIncrAuto	Integer	→	Use default major increment? -1 = No change 0 = No 1 = Yes
minIncrAuto	Integer	→	Use default minor increment? -1 = No change 0 = No 1 = Yes
minimum	Real	→	Minimum value
maximum	Real	→	Maximum value
majorIncr	Real	→	Major increment
minorIncr	Real	→	Minor increment

Description

The CT SET REAL SCALE command to use the default values or to specify the alternative scale values for a graph. CT SET REAL SCALE is used when the values are real numbers, integers, and/or long integers.

minAuto and maxAuto specify whether to use the default minimum and maximum values.

Example

This example creates a chart from arrays and sets the scale values.

```
      $ChartID:=CT Chart arrays (Area;2;1;aCategories;aSeries;aValues)
⇒     CT SET REAL SCALE (Area;$ChartID;0;0;0;0;-100;300;100;20)
```

See Also

CT GET DATE SCALE, CT GET REAL SCALE, CT SET DATE SCALE.

CT GET X REAL SCALE

CT Chart

version 6.0.2

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

Parameter	Type		Description
area	Longint	→	4D Chart area
object	Longint	→	Object ID
minAuto	Integer	←	Using default minimum? 0 = No 1 = Yes
maxAuto	Integer	←	Using default maximum? 0 = No 1 = Yes
majIncrAuto	Integer	←	Using default major increment? 0 = No 1 = Yes
minIncrAuto	Integer	←	Using default minor increment? 0 = No 1 = Yes
minimum	Real	←	Receives minimum value
maximum	Real	←	Receives maximum value
majorIncr	Real	←	Receives major increment
minorIncr	Real	←	Receives minor increment

Description

The CT GET X REAL SCALE command returns whether the default values are being used and what alternative values have been set for the X-axis scale in an XY chart. Use CT GET REAL SCALE for the Values axis (Z-axis) in the same type of graph when the values are real numbers, integers, and/or long integers.

minAuto and maxAuto specify whether the graph is currently using the default minimum and maximum values.

majIncrAuto and minIncrAuto specify whether the graph is currently using the default major and minor increments.

Example

This example returns scale data for the chart specified by $ChartID.

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

See Also

CT GET DATE SCALE, CT SET DATE SCALE, CT SET REAL SCALE.

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

Parameter	Type		Description
area	Longint	→	4D Chart area
object	Longint	→	Object ID
minAuto	Integer	→	Use default minimum? -1 = No change 0 = No 1 = Yes
maxAuto	Integer	→	Use default maximum? -1 = No change 0 = No 1 = Yes
majIncrAuto	Integer	→	Use default major increment? -1 = No change 0 = No 1 = Yes
minIncrAuto	Integer	→	Use default minor increment? -1 = No change 0 = No 1 = Yes
minimum	Real	→	Minimum value
maximum	Real	→	Maximum value
majorIncr	Real	→	Major increment
minorIncr	Real	→	Minor increment

Description

Use CT SET X REAL SCALE to use the default values or to specify the alternative scale values for the X-axis scale in an XY chart. Use CT SET REAL SCALE for the Values axis (Z-axis) in the same type of graph when the values are real numbers, integers, and/or long integers.

minAuto and maxAuto specify whether to use the default minimum and maximum values.

Example

This example creates a chart from arrays and sets the scale values.

```
      $ChartID:=CT Chart arrays (Area;2;1;aCategories;aSeries;aValues)
⇒     CT SET X REAL SCALE (Area;$ChartID;0;0;0;0;-100;300;100;20)
```

See Also

CT GET DATE SCALE, CT GET REAL SCALE, CT SET DATE SCALE.

CT GET DATE SCALE

CT Chart

version 6.0.5

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

Parameter	Type		Description
area	Longint	→	4D Chart area
object	Longint	→	Object ID
minAuto	Integer	←	Using default minimum? 0 = No 1 = Yes
maxAuto	Integer	←	Using default maximum? 0 = No 1 = Yes
majIncrAuto	Integer	←	Using default major increment? 0 = No 1 = Yes
minIncrAuto	Integer	←	Using default minor increment? 0 = No 1 = Yes
minimum	Date	←	Receives minimum value
maximum	Date	←	Receives maximum value
majIncrType	Integer	←	What is the major increment type? 1 = Days 2 = Weeks 3 = Months 4 = Years
majIncr	Integer	←	Receives major increment
minIncrType	Integer	←	What is the minor increment type? 1 = Days 2 = Weeks 3 = Months 4 = Years
minIncr	Integer	←	Receives minor increment

Description

The CT GET DATE SCALE command returns whether the default values are being used or

majIncrAuto and minIncrAuto specify whether the graph is currently using the default major and minor increments.

minimum and maximum are the minimum and maximum values set by the user in the Axis dialog box or by the designer using the CT SET DATE SCALE command.

majIncrType and minIncrType are codes for the units in which the majorIncr and minorIncr parameters are specified.

majorIncr and minorIncr are the major and minor increments set by the user in the Axis dialog box or by the designer using the CT SET DATE SCALE command.

Example

This example returns scale data for the chart specified by $ChartID.

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

See Also

CT GET REAL SCALE, CT SET DATE SCALE, CT SET REAL SCALE.

CT SET DATE SCALE

CT Chart

version 6.0.5

CT SET DATE SCALE (area; object; minAuto; maxAuto; majIncrAuto; minIncrAuto; minimum; maximum; majIncrType; majorIncr; minIncrType; minorIncr)

Parameter	Type		Description
area	Longint	→	4D Chart object
object	Longint	→	Object ID
minAuto	Integer	→	Use default minimum? -1 = No change 0 = No 1 = Yes
maxAuto	Integer	→	Use default maximum? -1 = No change 0 = No 1 = Yes
majIncrAuto	Integer	→	Use default major increment? -1 = No change 0 = No 1 = Yes
minIncrAuto	Integer	→	Use default minor increment? -1 = No change 0 = No 1 = Yes
minimum	Date	→	Minimum value
maximum	Date	→	Maximum value
majIncrType	Integer	→	Use default major increment type? -1 = No change 1 = Days 2 = Weeks 3 = Months 4 = Years
majorIncr	Integer	→	Major increment
minIncrType	Integer	→	Use default minor increment type? -1 = No change 1 = Days 2 = Weeks 3 = Months

Description

Use CT SET DATE SCALE to use the default values or to set alternative values for the Values axis scale.

CT SET DATE SCALE is used when the values are dates.

minAuto and maxAuto specify whether to use the default minimum and maximum values.

majIncrAuto and minIncrAuto specify whether to use the default major and minor increments.

minimum and maximum are the alternative minimum and maximum values.

majIncrType and minIncrType are codes for the units in which the majorIncr and minorIncr parameters are specified.

majorIncr and minorIncr are the alternative major and minor increments.

Example

This example creates a chart from the database and sets the scale values.

```
      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)
```

See Also

CT GET DATE SCALE, CT GET REAL SCALE, CT SET REAL SCALE.

CT GET X DATE SCALE

CT Chart

version 6.0.5

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

Parameter	Type		Description
area	Longint	→	4D Chart area
object	Longint	→	Object ID
minAuto	Integer	←	Use default minimum? 0 = No 1 = Yes
maxAuto	Integer	←	Use default maximum? 0 = No 1 = Yes
majIncrAuto	Integer	←	Use default major increment? 0 = No 1 = Yes
minIncrAuto	Integer	←	Use default minor increment? 0 = No 1 = Yes
minimum	Date	←	Receives minimum value
maximum	Date	←	Receives maximum value
majIncrType	Integer	←	What is the major increment type? 1 = Days 2 = Weeks 3 = Months 4 = Years
majIncr	Integer	←	Receives major increment
minIncrType	Integer	←	What is the minor increment type? 1 = Days 2 = Weeks 3 = Months 4 = Years
minorIncr	Integer	←	Receives minor increment type

Description

The CT GET X DATE SCALE command returns whether the default values are being used or what alternative values have been set for the X-axis scale in an XY chart. Use the

minAuto and maxAuto specify whether the graph is currently using the default minimum and maximum values.

majIncrAuto and minIncrAuto specify whether the graph is currently using the default major and minor increments.

minimum and maximum are the minimum and maximum values set by the user in the Axis dialog box or by the designer using the CT SET X DATE SCALE command.

majIncrType and minIncrType are codes for the units in which the majorIncr and minorIncr parameters are specified.

majorIncr and minorIncr are the major and minor increments set by the user in the Axis dialog box or by the designer using the CT SET X DATE SCALE command.

See Also

CT GET DATE SCALE, CT SET DATE SCALE, CT SET X DATE SCALE.

CT SET X DATE SCALE

CT Chart

version 6.0.5

CT SET X DATE SCALE (area; object; minAuto; maxAuto; majIncrAuto; minIncrAuto; minimum; maximum; majIncrType; majorIncr; minIncrType; minorIncr)

Parameter	Type		Description
area	Longint	→	4D Chart area
object	Longint	→	Object ID
minAuto	Integer	→	Use default minimum? -1 = No change 0 = No 1 = Yes
maxAuto	Integer	→	Use default minimum? -1 = No change 0 = No 1 = Yes
majIncrAuto	Integer	→	Use default major increment? -1 = No change 0 = No 1 = Yes
minIncrAuto	Integer	→	Use default minor increment? -1 = No change 0 = No 1 = Yes
minimum	Date	→	Minimum value
maximum	Date	→	Maximum value
majIncrType	Integer	→	-1 = No change 1 = Days 2 = Weeks 3 = Months 4 = Years
majorIncr	Integer	→	Major increment
minIncrType	Integer	→	-1 = No change 1 = Days 2 = Weeks 3 = Months 4 = Years
minorIncr	Integer	→	Minor increment

Description

Use CT SET X DATE SCALE to use the default values or to set alternative values for the X-axis scale in an XY chart. Use the CT SET DATE SCALE command for the Values axis (Z-axis) in the same type of chart when the values are dates.

minAuto and maxAuto specify whether to use the default minimum and maximum values.

majIncrAuto and minIncrAuto specify whether to use the default major and minor increments.

minimum and maximum are the alternative minimum and maximum values.

majIncrType and minIncrType are codes for the units in which the majorIncr and minorIncr parameters are specified.

majorIncr and minorIncr are the alternative major and minor increments.

See Also

CT GET DATE SCALE, CT GET REAL SCALE, CT GET X DATE SCALE, CT GET X REAL SCALE, CT SET DATE SCALE, CT SET REAL SCALE, CT SET X REAL SCALE.

CT GET DEPTH

version 6.0.5

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

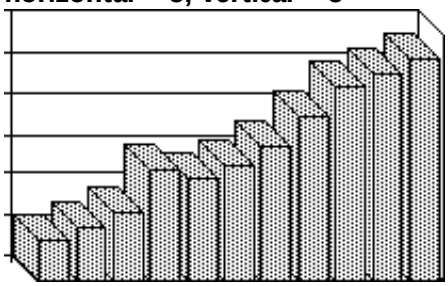
Parameter	Type		Description
area	Longint	→	4D Chart area
object	Longint	→	Object ID
horizontal	Integer	←	Receives horizontal offset in points
vertical	Integer	←	Receives vertical offset in points

Description

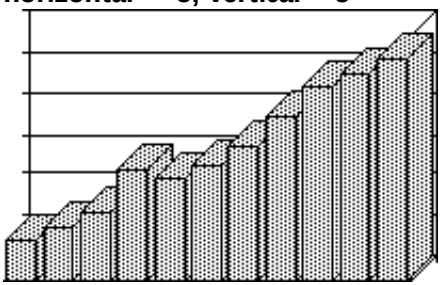
The CT GET DEPTH command returns the horizontal and vertical offsets (depth) of the graph specified by area and object. This command works only with two-dimensional graphs.

horizontal is the horizontal offset, measured in points. A positive value denotes an offset to the right; a negative value denotes an offset to the left.

horizontal = 5, vertical = 5

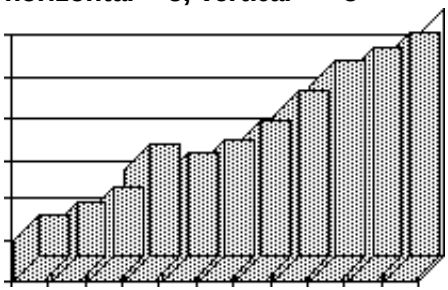


horizontal = -5, vertical = 5



vertical is the vertical offset, measured in points. A positive value denotes distance into the page from the X-axis; a negative value denotes distance out of the page from the X-axis.

horizontal = 5, vertical = - 5



Setting horizontal or vertical to zero removes the 3D effect for that dimension

Example

This example checks the depth of the chart specified by $ChartID and, if it the 3D effect is not present, sets it.

```
⇒    CT GET DEPTH (vArea;$ChartID;$Horiz;$Vert)
     If ($Horiz=0) & ($Vert=0)
        CT SET DEPTH (vArea;$ChartID;10;10)
     End if
```

See Also

CT SET DEPTH.

CT SET DEPTH

CT Chart

version 6.0.5

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

Parameter	Type		Description
area	Longint	→	4D Chart area
object	Longint	→	Object ID
horizontal	Integer	→	Horizontal offset in points (Must be greater than -32000)
vertical	Integer	→	Vertical offset in points (Must be greater than -32000)

Description

The CT SET DEPTH command sets the horizontal and vertical offsets (depth) of the graph specified by area and object. This command works only with two-dimensional graphs.

horizontal is the horizontal offset, measured in points. A positive value denotes an offset to the right; a negative value denotes an offset to the left.

vertical is the vertical offset, measured in points. A positive value denotes distance into the page from the X-axis; a negative value denotes distance out of the page from the X-axis.

For an illustration of horizontal and vertical depth, see the description for the CT GET DEPTH command.

Example

See the example for CT GET DEPTH.

See Also

CT GET DEPTH.

CT GET 3D VIEW

CT Chart

version 6.0.5

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

Parameter	Type		Description
area	Longint	→	4D Chart area
object	Longint	→	Object ID
rotation	Real	←	Receives rotation in degrees (0 to 90)
elevation	Real	←	Receives elevation in degrees (0 to 90)

Description

The CT GET 3D VIEW command returns the rotation and elevation of the graph specified by area and object. This command works only with three-dimensional graphs.

rotation is the rotation of the graph around the Z-axis.

elevation is the rotation of the graph around the X-axis.

Example

This example returns the rotation and elevation of the chart specified by $ChartID in the $Rotation and $Elevation parameters.

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

See Also

CT SET 3D VIEW.

CT SET 3D VIEW

CT Chart

version 6.0.5

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

Parameter	Type		Description
area	Longint	→	4D Chart area
object	Longint	→	Object ID
rotation	Real	→	Rotation in degrees (between 0 and 90, otherwise no effect)
elevation	Real	→	Elevation in degrees (between 0 and 90, otherwise no effect)

Description

The CT SET 3D VIEW command sets the rotation and elevation of the graph specified by area and object. This command works only with three-dimensional graphs.

rotation is the rotation of the graph around the Z-axis. rotation must be a value between 0 and 90.

elevation is the rotation of the graph around the X-axis. elevation must be a value between 0 and 90.

Example

This example sets the rotation and elevation of the chart specified by $ChartID to 30 degrees.

⇒ ***CT SET 3D VIEW*** (Area;$ChartID;30;30)

See Also

CT GET 3D VIEW.

CT GET CHART PART

CT Chart

version 6.0.5

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

Parameter	Type		Description
area	Longint	→	4D Chart area
object	Longint	→	Object ID
partType	Integer	←	Receives type of object selected in area
partSpecifics	Longint	←	Receives specific part of selected object

Description

The CT GET CHART PART command returns in the partType and partSpecifics parameters the codes for the chart object currently selected in the graph specified by area and object.

partType and partSpecifics specify the part of the graph that the user has selected. The codes for these parameters are listed in the section Parameter Codes.

Example

This example gets the part codes for the selected series and sets the series fill pattern to solid and fill color to green.

```
⇒    CT GET CHART PART (Area;$ChartID;$Type;$Specifics)
     If ($Type=8)  `Is it a series?
        CT SET CHART FILL ATTRIBUTES (Area;$ChartID;$Type;$Specifics;
                                      3;CT Index to color (10))
     End if
```

See Also

No reference.

CT GET CHART OPTIONS

CT Chart

version 6.0.5

CT GET CHART OPTIONS (area; object; options)

Parameter	Type		Description
area	Longint	→	4D Chart area
object	Longint	→	Object ID
options	Integer array	←	Array containing variables to receive option codes

Description

The CT GET CHART OPTIONS command returns the options for the selected graph. These options are equivalent to the options that the user can set in the Options dialog box for each chart type.

The codes for each graph type are listed in the section Parameter Codes.

Example

This example returns the options of the chart specified by $ChartID in aOptions.

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

See Also

CT SET CHART OPTIONS.

CT SET CHART OPTIONS

CT Chart

version 6.0.5

CT SET CHART OPTIONS (area; object; options)

Parameter	Type		Description
area	Longint	→	4D Chart area
object	Longint	→	Object ID
options	Integer array	→	Array containing option codes

Description

The CT SET CHART OPTIONS command sets the options for the selected graph. These options are equivalent to the options that the user can set in the Options dialog box for each chart type.

The options codes for each graph type are listed in the section Parameter Codes.

Example

This example sets the chart options of the column chart specified by $ChartID to horizontal, stacked proportional, 100 percent overlap, and 25 percent gap.

```
   ARRAY INTEGER (aOptions;4)
   aOptions{1}:=1
   aOptions{2}:=2
   aOptions{3}:=100
   aOptions{4}:=25
⇒  CT SET CHART OPTIONS (Area;$ChartID;aOptions)
```

See Also

CT GET CHART OPTIONS.

CT Get chart type

CT Chart

version 6.0.5

CT Get chart type (area; object) → Integer

Parameter	Type		Description
area	Longint	→	4D Chart area
object	Longint	→	Object ID
Function result	Integer	←	Chart type

Description

The command CT Get chart type returns the type of the chart specified by area and object.
The following table lists the codes for the chart types:

Code	Chart Type
1	Area
2	Column
3	Picture
4	Line
5	Scatter
6	Pie
7	Polar
8	2D XY
100	3D Column
101	3D Line
102	3D Area
103	3D Surface
104	3D Triangle
105	3D Spike

Example

In this example, CT Get chart type is used to get the chart type of the chart specified by $ChartID. If it is not a column chart, CT SET CHART TYPE changes it to a column chart.

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

See Also

CT SET CHART TYPE.

CT SET CHART TYPE

CT Chart

version 6.0.5

CT SET CHART TYPE (area; object; type)

Parameter	Type		Description
area	Longint	→	4D Chart
object	Longint	→	Object ID
type	Integer	→	Chart type

Description

The CT SET CHART TYPE command changes the type of the specified chart to type.

type must be a 2D chart type for a two-dimensional chart or a 3D chart type for a three-dimensional chart.

The following table lists the codes for the chart types:

Code	Chart Type
1	Area
2	Column
3	Picture
4	Line
5	Scatter
6	Pie
7	Polar
8	2D XY
100	3D Column
101	3D Line
102	3D Area
103	3D Surface
104	3D Triangle
105	3D Spike

Example

See the example for CT Get chart type.

See Also

CT Get chart type.

CT UPDATE CHART

CT Chart

version 6.0.5

CT UPDATE CHART (area; object; displayAlert)

Parameter	Type		Description
area	Longint	→	4D Chart area
object	Longint	→	Object ID
displayAlert	Integer	→	Display an alert to the user? 0 = No 1 = Yes

Description

The CT UPDATE CHART command updates a chart created from data in the database. This command is the equivalent of the Update menu command in the Chart menu.

This command updates only charts created by the user from fields in the database or programmatically using the CT Chart selection or CT Chart data functions.

The chart is updated using the records in the current selection of the table being graphed.
If displayAlert equals 1, an alert box is presented to the user, who can accept or cancel the action. If displayAlert equals 0, no alert box is presented.

Example

This example changes the selection of the table being graphed and updates the graph to display the changes.

```
     REDUCE SELECTION ([Statistics];350)
⇒    CT UPDATE CHART (Area;$ChartID;0)
```

See Also

No reference.

CT Get chart picture

CT Chart

version 6.0.5

CT Get chart picture (area; object; partType; partSpecifics) → Picture

Parameter	Type		Description
area	Longint	→	4D Chart area
object	Longint	→	Object ID
partType	Integer	→	Type of object for which to get picture
partSpecifics	Longint	→	Specific part of object for which to get picture
Function result	Picture	←	Picture displayed in the specified series of a picture chart

Description

CT Get chart picture returns the picture being displayed in the specified series of a picture chart. The picture is returned in a picture variable.

partType must be equal to 8, which is a series in a chart.

partSpecifics must be equal to the number of the series multiplied by 100.

Example

This example copies a picture from the first series of a picture graph and places it on the Clipboard.

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

See Also

CT SET CHART PICTURE.

CT SET CHART PICTURE

CT Chart

version 6.0.5

CT SET CHART PICTURE (area; object; partType; partSpecifics; picture)

Parameter	Type		Description
area	Longint	→	4D Chart area
object	Longint	→	Object ID
partType	Integer	→	Type of object for which to set attributes
partSpecifics	Longint	→	Specific part of object for which to set attributes
picture	Picture	→	Picture to paste in series in Picture chart

Description

CT SET CHART PICTURE pastes picture into the specified series.

object must be a picture chart.

partType must be equal to 8, which is a series in a chart.

partSpecifics must be equal to the number of the series multiplied by 100.

Example

This example copies a picture from the Clipboard to the first series in the selected graph, if the graph is a picture graph.

```
      $ChartID:=CT Get ID (Area;0;1)
      If (CT Get chart type (Area;$ChartID)=3)
         $Pict:=CT Clipboard to picture
⇒        CT SET CHART PICTURE (Area;$ChartID;8;100;$Pict)
      End if
```

See Also

CT Get chart picture.

CT EXPLODE PIE

CT Chart

version 6.0.5

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

Parameter	Type		Description
area	Longint	→	4D Chart area
object	Longint	→	Object ID
category	Integer	→	Category number of pie wedge to explode (0 = All wedges)
percentage	Integer	→	Percentage of the radius length (0 through 1000)

Description

The CT EXPLODE PIE command moves the specified wedges of a pie chart away from the center of the pie.

category is the number of the category whose wedge should be moved. If category is 0, all wedges are moved.

percentage is the distance to move the wedge, specified as a percentage of the radius length. Therefore, if the pie chart is resized, the distance changes to reflect the new radius.

Example

This example explodes the first three pieces of the pie chart specified by $ChartID. Each wedge is moved out a distance of 5% of the pie's radius.

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

See Also

No reference.

CT GET TIPS ATTRIBUTES

CT Chart

version 6.0.5

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

Parameter	Type		Description
area	Longint	→	4D Chart area
object	Longint	→	Object ID
axis	Integer	←	Selected axes 1 = Category 2 = Series 4 = Values
toolbar	Integer	←	Display 4D Chart toolbar 0 = No display 1 = Display
status	Integer	←	Display mode 0 = No tips 1 = Always display tips 2 = Only display tips on request
contents	Integer	←	Contents of tip 0 = Value only 1 = Percentage only 2 = Value and percentage
format	Alpha	←	Display format for Z-axis values
formatX	Alpha	←	Display format for X-axis values
method	Alpha	←	Name of the method to execute

Description

The command CT GET TIPS ATTRIBUTES returns the attributes of the tips specified by area and object in the variables for the parameters axis, toolbar, status, contents, format, formatX and method.

axis indicates the axes for which tips will be available. This is a composite number resulting from adding several axis numbers. The axis numbers are:

Number	Axis
1	Category axis
2	Series axis

status indicates the display settings for tips. Tips can be constantly active, active on request (if **Ctrl** on Windows or **Command** on Mac OS is pressed) or inactive.

contents allows you to know the type of information displayed. Information can be a value, a percentage or both.

format is the display format of the Z-axis tip values. If the format is “General”, an empty string "" is returned in format. For more information on special characters used in display formats, refer to the *4th Dimension Design Reference* manual.

formatX is similar to format except that it applies to the X-axis (XY graphs only).

method is the name of the method that will be executed each time a tip is displayed.

See Also

CT SET TIPS ATTRIBUTES.

CT SET TIPS ATTRIBUTES

CT Chart

version 6.0.5

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

Parameter	Type		Description
area	Longint	→	4D Chart area
object	Longint	→	Object ID
axis	Integer	→	Selected axes 1 = Category 2 = Series 4 = Values
toolBar	Integer	→	Display 4D Chart toolbar 0 = No display 1 = Display
status	Integer	→	Display mode 0 = No tips 1 = Always display tips 2 = Only display tips on request
contents	Integer	→	Contents of tip 0 = Value only 1 = Percentage only 2 = Value and percentage
format	Alpha	→	Display format for Z-axis values
formatX	Alpha	→	Display format for X-axis values
method	Alpha	→	Name of the method to execute

Description

The command CT SET TIPS ATTRIBUTES allows you to modify the attributes of the tips specified by area and object in the variables for the parameters axis, toolBar, status, contents, format, formatX and method.

axis indicates the axes for which tips will be available. This is a composite number resulting from adding several axis numbers. The axis numbers are:

Number	Axis
1	Category axis
2	Series axis
4	Values axis

toolBar indicates whether or not information is displayed in the 4D Chart toolbar.

status indicates the display settings for tips. Tips can be constantly displayed, displayed on request (if **Ctrl** on Windows or **Command** on Mac OS is pressed) or inactive.

contents allows you to know the type of information displayed. Information can be a value, a percentage or both.

format is the display format of the Z-axis tip values. If the format is “General”, an empty string "" is returned in format. For more information on special characters used in display formats, refer to the *4th Dimension Design Reference* manual.

formatX is similar to format except that it applies to the X-axis (XY graphs only).

method is the name of the method that will be executed each time a tip is displayed. It accepts four parameters. If method is an empty string, no method will be executes when the tip is displayed. When 4D Chart calls method, it returns four parameters ($1, $2, $3 and $4) that can be used for error processing.

Parameter	Description
$1 Long integer	Represents the 4D Chart area for which this method is executed
$2 Long integer	Contains the ID of the graph
$3 Long integer	Nth element of the category
$4 Long integer	Nth element of the series

The parameters $3 and $4 can be set to zero if the cursor is not located on a graph element.

If you want to compile your database later, declare these parameters as follows:

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

See Also

No reference.

CT GET CHART COORDINATES

CT Chart

version 6.0.5

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

Parameter	Type		Description
area	Longint	→	4D Chart area
object	Longint	→	Object ID
left	Integer	←	Horizontal position of chart's start (in points)
top	Integer	←	Vertical position of chart's start (in points)
right	Integer	←	Horizontal position of chart's end (in points)
bottom	Integer	←	Vertical position of chart's end (in points)

Description

The CT GET CHART COORDINATES command returns into the left, top, right, and bottom variables the coordinates of the rectangle of the chart designed by object inside area.

left indicates the distance between the left edge of the graph and the left edge of the screen.

top indicates the distance between the top of the graph and the top of the screen.

right indicates the distance between the right edge of the graph and the right edge of the screen.

bottom indicates the distance between the bottom of the graph and the bottom of the screen.

See Also

CT SET CHART COORDINATES.

CT SET CHART COORDINATES

CT Chart

version 6.0.5

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

Parameter	Type		Description
area	Longint	→	4D Chart area
object	Longint	→	Object ID
left	Integer	→	Horizontal position of chart's start (in points)
top	Integer	→	Vertical position of chart's start (in points)
right	Integer	→	Horizontal position of chart's end (in points)
bottom	Integer	→	Vertical position of chart's end (in points)

Description

The CT SET CHART COORDINATES command repositions the graph inside the 4D Chart area using the values passed as the left, top, right, and bottom parameters.

left indicates the distance between the left edge of the graph and the left edge of the screen.

top indicates the distance between the top of the graph and the top of the screen.

right indicates the distance between the right edge of the graph and the right edge of the screen.

bottom indicates the distance between the bottom of the graph and the bottom of the screen.

See Also

CT GET CHART COORDINATES.

6 CT Hot Links

CT PUBLISH (area; scope; name)

Parameter	Type		Description
area	Longint	→	4D Chart area
scope	Longint	→	Scope of the command -1 = All 0 = Selected object >0 = Object ID
name	String	→	Name of the hot link

Description

The CT PUBLISH command publishes the objects described by scope in area as a new hot link called name.

- If scope equals -1, CT PUBLISH publishes all of the objects in area.
- If scope equals 0, CT PUBLISH publishes the selected objects in area.
- If scope is greater than 0, it must be equal to a specific object's ID and only that object is published.

If name is an empty string (""), the Publish a Hot Link dialog box is displayed so that the user can choose a name.

Example

This example asks the user for a name and then publishes the selected objects as a hot link. If the name already exists, it alerts the user.

```
      $Name := Request ("Publish the selected object as...")
      If (OK = 1)
⇒        CT PUBLISH (Area;0;$Name)
         If (CT Error = 37)
            ALERT ("A hot link with that name already exists!")
         End if
      End if
```

See Also

CT Get ID, CT Subscribe, CT UNPUBLISH.

CT ADD TO HOT LINK

CT Hot Links

version 6.0.5

CT ADD TO HOT LINK (area; scope; name)

Parameter	Type		Description
area	Longint	→	4D Chart area
scope	Longint	→	Scope of the command -1 = All 0 = Selected objects >0 = Object ID
name	String	→	Name of hot link

Description

The CT ADD TO HOT LINK command adds to the hot link name, the objects in area described by scope.

• If scope equals -1, CT ADD TO HOT LINK adds all objects in the document to the hot link.

• If scope equals 0, CT ADD TO HOT LINK adds the currently selected objects to the hot link.

• If scope is greater than 0, it must be equal to a specific object's ID and that object is added to the hot link.

If name is an empty string (""), the Choose a Hot Link dialog box is displayed so that the user can choose a hot link.

If name is not a hot link of area, CT ADD TO HOT LINK does nothing.

Example

This example asks the user for a name and then adds the selected objects to the specified hot link. If the hot link does not exist, it alerts the user

```
    $Name := Request ("Add the selected object(s) to which hot link?")
    If (OK = 1)
⇒       CT ADD TO HOT LINK (Area;0;$Name)
        If (CT Error = 5)
            ALERT ("There is no hot link with that name.")
        End if
    End if
```

See Also

CT PUBLISH, CT Subscribe.

CT UNSUBSCRIBE

CT UNSUBSCRIBE (area; name)

Parameter	Type		Description
area	Longint	→	4D Chart area
name	String	→	Name of hot link

Description

The CT UNSUBSCRIBE command causes area to quit subscribing to the hot link described by name.

All copies of name are removed from area. This command is the procedural equivalent of choosing name from the Unsubscribe to Hot Link dialog box. To remove only one instance of a hot link, simply delete the object.

If name is an empty string (""), the Unsubscribe to Hot Link dialog box is displayed so that the user can choose a hot link.

Example

This example asks the user for a name and then removes all instances of the specified hot link area.

```
      $Name := Request ("Remove which hot link?")
      If (OK = 1)
⇒        CT UNSUBSCRIBE (Area;$Name)
      End if
```

See Also

CT PUBLISH, CT Subscribe.

CT Subscribe

CT Hot Links

version 6.0.5

CT Subscribe (area; name) → Longint

Parameter	Type		Description
area	Longint	→	4D Chart area
name	String	→	Name of hot link
Function result	Longint	←	Object ID of the new object

Description

CT Subscribe subscribes to the already existing hot link name and returns the new object's ID. This function creates a new object in area.

If name is an empty string (""), the Subscribing to a Hot Link dialog box is displayed so that the user can choose a hot link. If name doesn't exist, CT Subscribe returns -32000.

If the specified hot link is a Picture hot link, the PICT is displayed as a picture in the 4D Chart area. If the specified hot link is a Values hot link, 4D Chart generates a graph from the data using the default hot link type specified in the Properties dialog box.

Example

This example asks the user for a name and then subscribes to the specified hot link. If the subscription is successful, it centers the hot link in the visible area.

```
      `Get the hot link
   $Name := Request ("Subscribe to which hot link?")
   If (OK = 1)  `If the request is accepted
⇒     $Hotlink := CT Subscribe (Area;$Name)      `Subscribe to the hot link
      If (CT Error=0)     `If there were no errors
                  `Get the clipped boundary of the area
         CT GET AREA BOUNDARY (Area;1;$left;$top;$right;$bottom)
                  `Get the hot link's boundary
         CT GET BOUNDARY (Area;$HotLink;$left2;$top2;$right2;$bottom2)
                  `Center the hot link in the area
         CT MOVE (Area;$HotLink;(($right-$left)-($right2-$left2))/2;
                                 (($bottom-$top)-($bottom2-$top2))/2)
      End if
   End if
```

See Also

CT PUBLISH, CT UNPUBLISH.

CT UNPUBLISH

CT UNPUBLISH (area; name)

Parameter	Type		Description
area	Longint	→	4D Chart area
name	String	→	Name of hot link

Description

The CT UNPUBLISH command stops publishing the hot link called name from area. This command is equivalent to choosing a hot link in the Unpublish Hot Link dialog box.

If name is an empty string (""), the Unpublish a Hot Link dialog box is displayed so that the user can choose a hot link to stop publishing.

If name does not exist or is not a hot link published from area, CT UNPUBLISH does nothing.

Example

This example asks the user for a name and then unpublishes the specified hot link.

```
      $Name := Request ("Enter the name of the hot link to unpublish.")
      If (OK = 1)
⇒        CT UNPUBLISH (Area;$Name)
         If (CT Error = 5)
            ALERT ("There is no hot link with that name!")
         End if
      End if
```

See Also

CT PUBLISH, CT Subscribe.

7 CT Printing

CT PRINT (area; cancellable; printDialog)

Parameter	Type		Description
area	Longint	→	4D Chart area
cancellable	Integer	→	Allow printing to be cancelled? 0 = Do not allow cancelling 1 = Allow cancelling
printDialog	Integer	→	Present Print dialog box? 0 = Without dialog box 1 = With dialog box

Description

The CT PRINT command prints the document in area. Calling CT PRINT performs a function similar to choosing Print from the File menu, except that the Print Setup dialog box is not presented to the user. To display the Print Setup dialog box before printing, use CT DO COMMAND.

If cancellable equals 1, 4D Chart displays a dialog box that allows the user to cancel printing by pressing Ctrl+Period (Command-Period on Macintosh). If the user cancels printing, CT Error returns error number 20. If cancellable equals 0, the dialog box is not displayed and the user cannot cancel printing.

If the printDialog parameter equals 0, the standard Print File dialog box does not appear and the print job begins immediately. If printDialog equals 1, the standard Print File dialog box appears.

See Also

CT DO COMMAND.

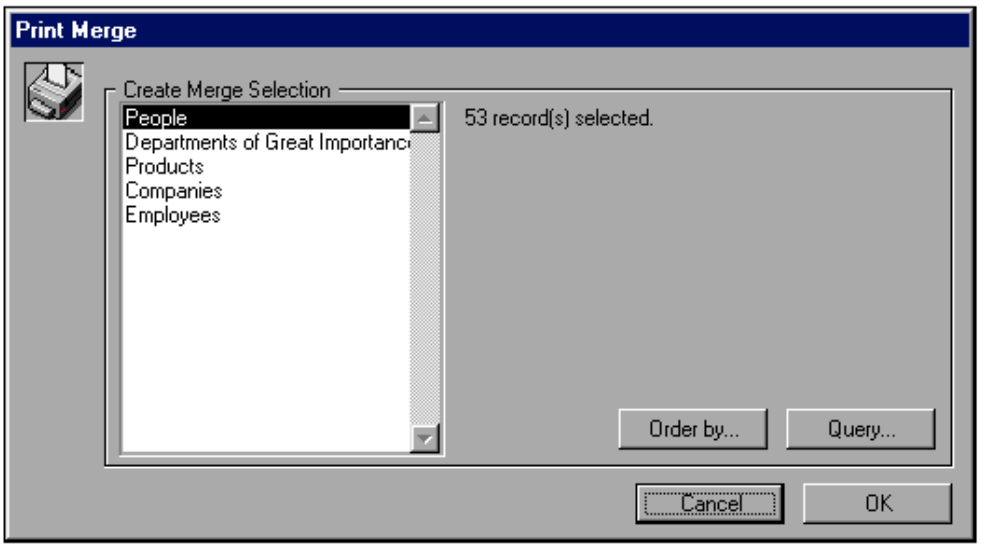
CT PRINT MERGE (area; table; cancellable; printDialog)

Parameter	Type		Description
area	Longint	→	4D Chart area
table	Integer	→	Table number
cancellable	Integer	→	Allow printing to be cancelled? 0 = Do not allow cancelling 1 = Allow cancelling
printDialog	Integer	→	Present Print dialog box? 0 = Without dialog box 1 = With dialog box

Description

The CT PRINT MERGE command allows you to perform a print merge for the current selection of the fileNum file. The document used for the print merge is specified by area.

If table equals 0, the Create Merge Selection dialog box is displayed.



If cancellable equals 1, a dialog box appears, allowing you to cancel the printing in progress by pressing Command-period (Macintosh) or Ctrl-period (Windows) . If the user cancels the print job, CT Error returns error number 20. If cancellable equals 0, this dialog box does not appear, and the user cannot cancel the print job.

printDialog determines whether or not the standard Print dialog box appears. If printDialog equals 0, the standard Print File dialog box does not appear and the print job begins immediately. If printDialog equals 1, the standard Print File dialog box appears.

Example

This example ensures that the 4D Chart area is set to Show Values mode and performs a print merge:

```
   `Generate selection for print merge
ALL RECORDS([MyFile])
   `Ensure that display mode is set to Show Values
CT MENU STATUS (Area;6006;$Checked;$Available;$Name)

If ($Name="Show Values")
   CT DO COMMAND (Area;6006)      `Set display to Show Values mode
End if

   `Perform the print merge with no user intervention
⇒  CT PRINT MERGE (Area;File(»[MyFile]);0;0)
```

See Also

No reference.

8 CT Utilities

There are three ways to specify a color in 4D Chart:

- RGB: Red, green, and blue values (3 longints).
- Index: Colors 1 through 256 in the 4th Dimension palette, which are displayed in the Fill Color, Line Color, and Text Color submenus if your monitor is set for 256 colors (1 integer).
- Color: An internal value used by 4D Chart (1 longint).

The utility commands discussed in this section convert color values between these three conventions. The Chart and Object theme commands in expect the Color longint as a parameter.

CT RGB to color

CT Utilities

version 6.0.5

CT RGB to color (red; green; blue) → Longint

Parameter	Type		Description
red	Longint	→	Red value (0 to 65535)
green	Longint	→	Green value (0 to 65535)
blue	Longint	→	Blue value (0 to 65535)
Function result	Longint	←	Value of the color (internal to 4D Chart)

Description

CT RGB to color returns a value that represents the red, green, and blue components. This number is used in several 4D Chart commands.

The red, green, and blue parameters are the same as those used in the Macintosh Color Picker.

The following table shows the values for red, green, and blue in three commonly used colors:

Color	Red	Green	Blue
red	56683	2242	1698
green	0	32768	4528
blue	0	0	54272

Example

This example sets a color variable for the color red:

```
⇒ $color:=CT RGB to color(56683;2242;1698)
```

See Also

CT Color to index, CT COLOR TO RGB, CT index to color.

CT COLOR TO RGB

CT COLOR TO RGB (color; red; green; blue)

Parameter	Type		Description
color	Longint	→	Color value (internal to 4D Chart)
red	Longint	←	Receives red value (0 to 65535)
green	Longint	←	Receives green value (0 to 65535)
blue	Longint	←	Receives blue value (0 to 65535)

Description

CT COLOR TO RGB breaks down the color value in color and returns values in the variables representing the red, green, and blue components.

Example

This example gets the component RGB values for a color variable and places them in several local variables:

```
      $color:=100000
⇒     CT COLOR TO RGB($color;$red;$green;$blue)
```

See Also

CT Color to index, CT Index to color, CT RGB to color.

CT Index to color

CT Utilities

version 6.0.5

CT Index to color (index) → Longint

Parameter	Type		Description
index	Integer	→	Palette index
Function result	Longint	←	Color described by index

Description

CT Index to color returns the color described by index.

Index is an integer that specifies a particular color on the 4th Dimension color palette. Colors on the 4th Dimension palette are numbered from 1 to 256.

CT Index to color is a convenient way to specify a color without knowing the individual components.

Example

This example places the color value of the color Cyan (palette index #8) into the vColor variable:

⇒ vColor:=***CT Index to color***(8)

See Also

CT Color to index, CT COLOR TO RGB, CT RGB to color.

CT Color to index

CT Utilities

version 6.0.5

CT Color to index (color) → Integer

Parameter	Type		Description
color	Longint	→	Color value (internal to 4D Chart)
Function result	Integer	←	Index of the closest color in the 4D palette

Description

CT Color to index returns the index of the color closest to color on the 4th Dimension color palette.

For instance, if a particular shade of blue is specified, CT Color to index returns the closest shade of blue in the 4th Dimension palette. Colors on the 4th Dimension palette are numbered from 1 to 256.

Example

This example places the palette index of the color closest to red in the vColor variable:

```
⇒ vColor:=CT Color to index(CT RGB to color(56683;2242;1698))
```

See Also

CT COLOR TO RGB, CT Index to color, CT RGB to color.

CT Font name

CT Utilities

version 6.0.5

CT Font name (fontNumber) → String

Parameter	Type		Description
fontNumber	Integer	→	Font ID number
Function result	String	←	Name of the font

Description

CT Font name returns the name of the font whose ID is fontNumber.

If fontNumber does not exist, CT Font name returns an empty string.

Example

This example returns the name of the font whose ID number is 3:

```
⇒ vName:=CT Font name(3)
```

See Also

CT Font name.

CT Font number

CT Font number (fontName) → Integer

Parameter	Type		Description
fontName	String	→	Name of font
Function result	Integer	←	ID number of font

Description

CT Font number returns the ID number of the font whose name is fontName.

Example

This example returns the number of the font "Times":

```
⇒ vNumber:=CT Font number("Times")
```

See Also

CT Font name.

CT Clipboard to picture

CT Utilities

version 6.0.5

CT Clipboard to picture → Picture

Parameter	Type		Description
This command does not require any parameters			
Function result	Picture	←	A copy of contents of the Clipboard

Description

CT Clipboard to picture returns a 4th Dimension picture that is a copy of the contents of the Clipboard.

If the Clipboard does not contain a picture, CT Clipboard to picture returns an empty picture.

Example

This example copies a picture from the Clipboard to the vPict variable:

⇒ vPict:=***CT Clipboard to picture***

See Also

CT PICTURE TO CLIPBOARD.

CT PICTURE TO CLIPBOARD

CT PICTURE TO CLIPBOARD (picture)

Parameter	Type		Description
picture	Picture	→	Picture

Description

The CT PICTURE TO CLIPBOARD command copies picture to the Clipboard.

Once on the Clipboard, picture can be pasted anywhere pictures are allowed.

Example

This example copies the contents of the field [Chart]Object to the Clipboard:

```
⇒ CT PICTURE TO CLIPBOARD([Chart]Object)
```

See Also

CT Clipboard to picture.

9 Control Codes

The following list contains the codes of the command parameter:

MENU	Codes	COMMANDS
Tool Palette	1	Pointer
	2	Text
	3	Line
	4	Rectangle
	5	Round rectangle
	6	Ovale
	7	Polygon
File	1001	New
	1002	Open
	1003	Save
	1004	Save as
	1006	Save as Template
	1008	Page Setup
	1009	Print
	1010	Print Merge
	1012	Go to Full Window/Return to Form
	1013	Import
	1014	Export selection as
	1015	Export
Edit	2001	Undo
	2003	Cut
	2004	Copy
	2005	Paste
	2006	Clear
	2007	Duplicate
	2009	Select All
	2011	Properties
	4016	Display menu
Texte	3001	Font menu

Chart	4002	New chart
	4003	Axis menu
	4004	Grid Lines menu
	4005	Titles menu
	4007	Legends
	4012	View
	4009	Values
	4017	Tips
	4010	Options
	4014	Update
Object	5001	Fill Pattern menu
	5002	Fill Color menu
	5004	Line Pattern menu
	5005	Line Color menu
	5007	Line Width menu
	5008	Arrowhead menu
	5010	Round corners
	5012	Arrange menu
Database	6001	Paste Field
	6002	Format
	6003	Reference
	6004	Unreference
	6006	Show Values/Show References
	6008	Subscribe to Hot Link
	6009	Publish Hot Link
	6010	Unsubscribe to Hot Link
	6011	Unpublish Hot Link
	6012	Add to Hot Link
Font	7001 – 7999	Names of individual fonts
Size	8001 – 8009	Sizes of individual fonts
	8010	Other Size
Style	9001	Plain
	9002	Bold
	9003	Italic
	9004	Underline
	9005	Outline
	9006	Shadow

New 2D Chart	12001	2D Area
	12002	2D Column
	12003	2D Line
	12005	2D Pie
	12006	2D Picture
	12007	2D Polar
	12008	2D XY
New 3D Chart	13001	3D Column
	13002	3D Line
	13003	3D Area
	13004	3D Surface
	13005	3D Triangle
	13006	3D Spike
Axis	14001	Category (X axis)
	14002	Values (Z axis) (2D chart) Series (Y axis) (3D chart)
	14003	Values (Z axis) (3D chart)
Grid Lines	15001	Category (X axis)
	15002	Values (Z axis) (2D chart)) Series (Y axis) (3D chart)
	15003	Values (Z axis) (3D chart)
Titles	16001	Category (X axis)
	16002	Values (Z axis) (2D chart) Series (Y axis) (3D chart)
	16003	Values (Z axis) (3D chart)
Display	17001	Menu Bar
	17002	Object Tools
	17003	Chart Tools
	17004	Scroll Bars
	17005	Tulers
Fill Pattern	18001 – 18036	Individual fill patterns
Fill Color	19001 – 19256	Individual fill colors

Line Pattern	20001 – 20036	Individual line patterns
Line Color	21001 – 21256	Individual line colors
Line Width	22001	0,25 point
	22002	1 point
	22003	2 points
	22004	4 points
	22005	6 points
	22006	Other Width
Arrowheads	23001	No arrowhead
	23002	Start
	23003	End
	23004	Both ends
Arrange	24001	Bring to Front
	24002	Send to Back
	24003	Move Forward
	24004	Move Backward
	24006	Align Objects
	24008	Group
	24009	Ungroup

The following are the codes for 4D Chart error messages:

Error	Message
1	Invalid 4D Chart area reference.
2	Unable to load segment to complete operation.
3	Invalid command number.
4	Item disabled.
5	Hot link was not found.
6	Invalid 4D table number.
7	Invalid 4D field number.
8	Picture was not created.
9	Invalid scope in CT AREA TO AREA.
10	Source and destination areas must be different.
11	Invalid scope.
12	Invalid object ID.
13	This operation cannot be performed on this type of object.
14	Invalid object index.
15	There are no selected objects.
16	There are no objects in this document.
17	Invalid file type.
18	The version of this 4D Chart document is no longer supported.
19	This 4D Chart document was created with a later version.
20	User canceled the dialog box.
21	Invalid hot link type.
22	Adding to this hot link would generate recursion in the hot link chain.
23	This operation would cause objects to be moved outside the document.
24	There are no 4D tables.
25	Bitmapped image too large.
26	This operation would cause the maximum number of objects to be exceeded.
27	There are not enough categories to create a chart.
28	There are too many categories to create a chart.
29	There are not enough series to create a chart.

30	There are too many series to create a chart.
31	Invalid chart type.
32	Invalid chart sizing.
33	Invalid field type for category.
34	Invalid field type for values.
35	Invalid array type.
36	Multiple values.
37	A hot link with that name and type already exists.
38	Need at least one character.
39	Invalid object boundary.
40	Invalid corner rounding amount.
41	Invalid RGB value.
42	Invalid color value.
43	Invalid color index.
44	The Clipboard does not contain a picture.
45	All values are out of range.
46	Not in text editing mode.
47	Invalid axis index.
48	Invalid grid index.
49	Invalid number of vertices for polygon.
50	Invalid coordinate(s).
51	Invalid object size.
52	Object(s) specified do not have this attribute.
53	File pathname exceeds 255 characters.
54	Invalid field type.
55	Does not apply to this kind of chart.
56	Invalid value from the Clipboard.
57	Invalid dimensions from the Clipboard.
58	Invalid selection values.
59	Could not initialize printer resources.
60	Insufficient memory to allocate an offscreen area.
61	Insufficient memory to use color resources.
62	Invalid display item number.
63	Insufficient memory to copy a text object.
64	Invalid font.
65	There are no selected records.
66	Incorrect value for this type of chart.

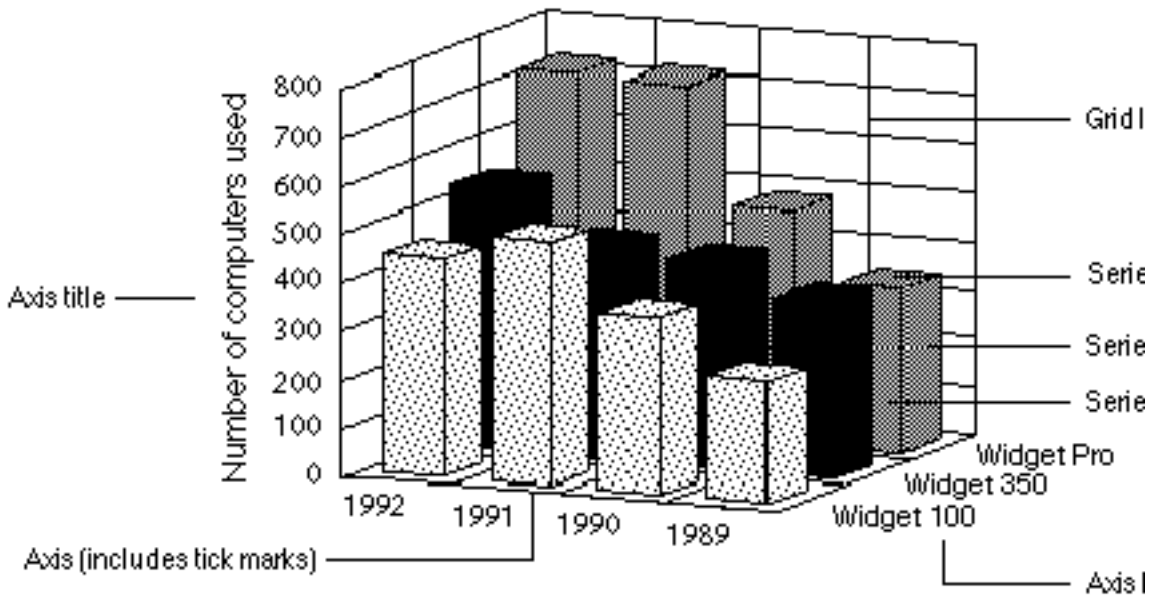
This section contains the codes for frequently used parameters of 4D Chart. These parameters are:

- partType
- partSpecifics
- pattern
- options

Part Type and Part Specifics Codes

When you modify parts of a graph programmatically, such as the legend or one of the axes, you specify which part of the graph to modify using the partType and partSpecifics parameters.

Some of the major parts of a graph are labelled in the following illustration:



• The following are the codes for the partType parameter:

Code	Graph Part
1	Plot rectangle
2	Legend
3	Axis
4	Label
5	Title
6	Major gridlines
7	Minor gridlines
8	Series
9	Value labels for series

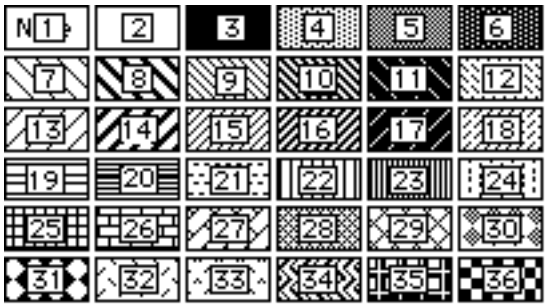
• The following are the codes for the partType parameter:

Graph Part	Specific Part Code
Plot rectangle	For 2D graphs: 0 = Entire rectangle, For 3D graphs: 1 = Back, 2 = Side, 3 = Bottom
Legend	0 = Legend rectangle
Axis	0 = Category, 1 = Series, 2 = Values
Labels	0 = Category, 1 = Series, 2 = Values
Titles	0 = Category, 1 = Series, 2 = Values
Major gridlines	0 = Category, 1 = Series, 2 = Values
Minor gridlines	0 = Category, 1 = Series, 2 = Values
Series/Value labels for series	(Series number * 100) + (Side number)
Side numbers (for sides of series)	0 = All sides (also use for all 2D graphs) 1 = Avant, 2 = Left, 3 = Right, 4 = Top, 5 = Bottom

Note: Series in 2D graphs have one side (Front); series in 3D graphs can display up to three sides at a time.

Pattern Codes

Following are the codes for the pattern parameter:



Option Codes

Following are the codes for the options parameter used in the CT GET CHART OPTIONS and CT SET CHART OPTIONS commands:

Graph Type	Element Number	Codes
Area	1	Orientation: 0 = Vertical, 1 = Horizontal
	2	Stacked: 0 = Normal, 1 = Stacked, 2 = Stacked Proportional
Column	1	Orientation: 0 = Vertical, 1 = Horizontal
	2	Stacked: 0 = Normal, 1 = Stacked, 2 = Stacked Proportional
	3	Overlap percent (0 through 100)
	4	Gap percent (0 through 100)
Line	1	Orientation: 0 = Vertical, 1 = Horizontal
	2	Stacked: 0 = Normal, 1 = Stacked, 2 = Stacked Proportional
	3	Controls display of squares on a 2D line: 0 = Invisible, 1 = Visible
Scatter	1	Orientation: 0 = Vertical, 1 = Horizontal
	2	Stacked: 0 = Normal, 1 = Stacked, 2 = Stacked Proportional
Pie	1	Start angle (0 through 360)

Picture	1	Orientation: 0 = Vertical, 1 = Horizontal
	2	Stacked: 0 = Normal, 1 = Stacked, 2 = Stacked Proportional
	3	Overlap percent (0 through 100)
	4	Gap percent (0 through 100)
	5	Horizontal alignment: 0 = Left, 1 = Center, 2 = Right
	6	Vertical alignment: 0 = Top, 1 = Center, 2 = Bottom
	7	Horizontal display: 0 = Clipped, 1 = Stretched, 2 = Stacked
	8	Vertical display: 0 = Clipped, 1 = Stretched, 2 = Stacked
Polar		None
XY Chart	1	Form of point: 0 = No point display, 1 = Circles, 2 = Squares, 3 = Stars
	2	Type of line: 0 = No lines, 1 = Straight lines, 2 = Arrow lines
	3	Display regression line: 0 = No, 1 = Yes
3D Column	1	Specify category percent as percent of: 0 = Width, 1 = Gap
	2	Specify series percent as percent of: 0 = Width, 1 = Gap
	3	Category percent (0 through 100)
	4	Series percent (0 through 100)
	5	Visible sides: 0 = All, 1 = Tops only
3D Line	1	Specify series percent as percent of: 0 = Width, 1 = Gap
	2	Series percent (0 through 100)
3D Area	1	Specify series percent as percent of: 0 = Width, 1 = Gap
	2	Series percent (0 through 100)
3D Surface	1	Visible sides: 0 = All, 1 = Tops only
3D Triangle	1	Flipped: 0 = False, 1 = True
	2	Plot zero values: 0 = False, 1 = True
	3	Specify series percent as percent of: 0 = Width, 1 = Gap
	4	Series percent (0 through 100)
3D Spike	1	Spike head: 0 = Oval, 1 = Square

See Also

No reference.

Command Index