IBM Cognos PowerPlay Client
Version 11.0

Macro Reference Guide

©

Product Information

This document applies to IBM Cognos Analytics version 11.0.0 and may also apply to subsequent releases.

Copyright

Contents

Introduction

This document includes information about using OLE automation for use with IBM® Cognos® PowerPlay® Client tasks.

Finding information

To find product documentation on the web, including all translated documentation, access IBM Knowledge Center (http://www.ibm.com/support/knowledgecenter).

Accessibility features

IBM Cognos PowerPlay Client does not currently support accessibility features that help users with a physical disability, such as restricted mobility or limited vision, to use this product.

Forward-looking statements

This documentation describes the current functionality of the product. References to items that are not currently available may be included. No implication of any future availability should be inferred. Any such references are not a commitment, promise, or legal obligation to deliver any material, code, or functionality. The development, release, and timing of features or functionality remain at the sole discretion of IBM.

Samples disclaimer

The Sample Outdoors Company, Great Outdoors Company, GO Sales, any variation of the Sample Outdoors or Great Outdoors names, and Planning Sample depict fictitious business operations with sample data used to develop sample applications for IBM and IBM customers. These fictitious records include sample data for sales transactions, product distribution, finance, and human resources. Any resemblance to actual names, addresses, contact numbers, or transaction values is coincidental. Other sample files may contain fictional data manually or machine generated, factual data compiled from academic or public sources, or data used with permission of the copyright holder, for use as sample data to develop sample applications. Product names referenced may be the trademarks of their respective owners. Unauthorized duplication is prohibited.

Chapter 1. PowerPlay OLE Automation

You can use OLE automation to automate tasks in IBM Cognos PowerPlay by creating macros using the IBM CognosScript language.

You can write macros using

- IBM Cognos Series 7 CognosScript Editor

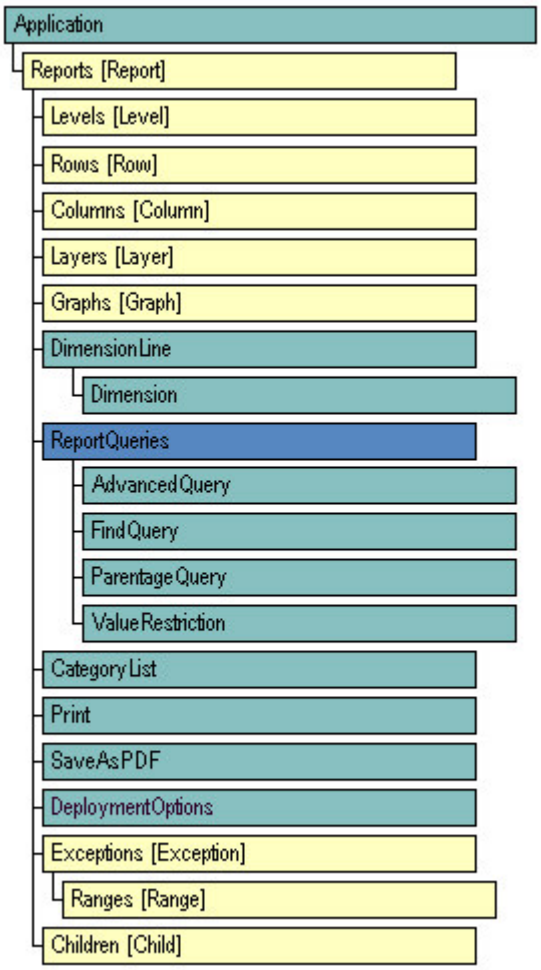
  IBM Cognos PowerPlay does not include CognosScript Editor. You can use IBM Cognos Series 7 CognosScript Editor to create and run macros for IBM Cognos PowerPlay. For more information, see the IBM Cognos Series 7 CognosScript Editor *Reference Guide*.
- Microsoft Visual Basic

  For more information, see "How the CognosScript Language Compares to Different Versions of Basic" on page 3
- a text editor, such as Notepad
- When you create a macro using a text editor, you cannot compile or run it from the editor. You must compile and run the macro in the IBM Cognos Series 7 CognosScript Editor or a compatible macro editor such as Microsoft Visual Basic for Applications.

PowerPlay Application Hierarchy

The PowerPlay application hierarchy is organized into a logical tree structure of objects and collections that depend on each other to operate. The structure shows which objects must exist before you can use the properties or methods to control the behavior and characteristics of another object.

The following diagram shows the relationship between the collections and objects and the order in which they fit in the hierarchy.



Related Topics

- "Macro Script Design"
- Chapter 4, "Methods," on page 71
- Chapter 5, "Properties," on page 259
- Chapter 6, "Administrative Macros," on page 433

Macro Script Design

Consider error handling, scheduling, and distribution when you create macro scripts.

Error Handling

A script can be thoroughly debugged and still encounter conditions that cause errors, such as running a script that attempts to open a file that does not exist or is unavailable. When a IBM Cognos PowerPlay user encounters errors, they can respond to the error and continue working. However, when a macro encounters an error, it stops. The remaining macro instructions are not executed unless your macro includes instructions to handle errors.

When you write macros, the goal is to write a macro that is not too dependent on a particular set of conditions. For example, you do not want to create a macro that

requires precise and accurate input from the user. By structuring your macros to include IF statements, you can try to anticipate your user's responses.

A macro that functions properly should

- close any documents the macro opened but the user no longer needs
- save, or prompt the user to save, a file that should be saved
- restore the settings of any options the macro may have changed
- delete any temporary files

Scheduling

If you use IBM Cognos Series 7 Scheduler to run a macro, estimate how long the macro will take to execute. For longer activities, you may want to schedule the macro to run during off-hours to minimize network activity. However, too many large macros, along with other routine network activities, may also affect the successful execution of your macros.

Distribution

You can use any file distribution method available to distribute models, cubes, reports, and macros.

When you define your PowerPlay strategies for automating model, cube, and report distribution, you need to

- identify repetitive tasks, such as identifying the models, views, cubes, and reports to change or update regularly
- list the users or user communities that require the files
- indicate how, and when, each user or user community will receive the updates

How the CognosScript Language Compares to Different Versions of Basic

The following information will help you understand the differences between CognosScript and different versions of Microsoft Basic. You must consider these differences if you plan to use Microsoft Basic instead of IBM Cognos Series 7 CognosScript Editor to create and run macros.

There are several versions of Basic with which you may be familiar, the most common being Microsoft Visual Basic and Word Basic. The CognosScript Language shares a substantial common core of functions and statements with these versions; however, each one has unique capabilities.

Differences Between the CognosScript Language and Visual Basic

The CognosScript Language is very similar to Microsoft Visual Basic; however, there are some critical differences. The topics below describe some of the differences you will notice between the CognosScript Language and Visual Basic.

Functions and Statements Unique to the CognosScript Language

The CognosScript Language offers a few statements and functions not found in the standard version of Visual Basic.

Some of these statements and functions are as follows:

- $CStrings metacommand
- $Include metacommand
- $NoCStrings metacommand
- GetField$ function
- SetFields$ function
- Assert statement

Control-Based Objects

The CognosScript Language does not include a Visual Basic form of control-based objects. As a result, a Visual Basic property like "BorderStyle" is not an intrinsic part of the CognosScript Language. This does not mean that you cannot define an object in the CognosScript Language that has BorderStyle as a property. You will probably define many objects that are intrinsic to your application in the process of integration.

Differences Between the CognosScript Language and Microsoft Visual Basic for Applications

Microsoft offers a modified version of Visual Basic in some of its products, called Visual Basic for Applications. In addition to the functions and statements unique to the CognosScript Language, there are differences you will notice between the CognosScript Language and Microsoft Visual Basic for Applications.

In Microsoft Visual Basic for Applications

- a Global Const is treated as a Const. An equivalent would be Public Const.
- there are differences in the behavior of the Declare statement:
  - Forward declarations to functions are not available.
  - The BasicLib attribute is not recognized and should be converted to Lib.
  - The Ordinal attribute will not be treated as an ordinal number of a procedure of an external dll unless it is prefixed by "#".
- prompted user input in the form of the Input or Line Input statement are not available.
- the Print statement when used without a file number should be changed to the equivalent Debug.Print statement.
- can not access clipboard objects are not accessible.
- the Erl function is not recognized.

Dialog Box Capabilities and Microsoft Visual Basic for Applications

Microsoft Visual Basic for Applications does not have a syntax to create or run dialog boxes. In response to this, the CognosScript Language has a set of functions and statements to enable the use of dialog boxes (they are similar to those in Word).

Microsoft Visual Basic for Applications does provide dialog box handling statements and functions. However, the CognosScript dialog functions and statements will not work directly in Visual Basic for Applications. You must port the dialog scripts to custom user forms.

Differences Between the CognosScript Language and Word Basic

Word Basic is a version of Visual Basic that was included in earlier versions Microsoft Word. Word Basic supports dialog boxes, but it does not support objects. The topics below describe some of the differences you will notice between the CognosScript Language and Word Basic.

Dialog Box Capabilities

The dialog box capabilities in the CognosScript Language and Word are very similar. Word does offer some statements and functions that the CognosScript Language does not, such as DlgFilePreview.

As well, the CognosScript Language offers some features that Word does not, such as:

- Button
- Button Group
- Caption
- DropComboBox
- StaticComboBox

In response to the need for certain types of dialog box support, the CognosScript Language offered some dialog box options before Word Basic did. Later, Word Basic came out with their own syntax for these options. As a result, there are differences in the way the two languages handle dialog boxes.

Button vs. PushButton

Button is the original CognosScript Language syntax; PushButton is the Word Basic syntax. The two are interchangeable, and the CognosScript Editor supports both. PushButton is preferred.

Dialog Box Units

The measurement units used in the two dialog box syntaxes are different. The CognosScript Editor supports both, and you can choose to use either.

Since many of our clients have built scripts based on the original CognosScript units, those are the ones used in the Examples. As a result, if you use Word units, some of the dialog boxes created in the Examples may look odd.

User Input Mechanisms

There are slight differences in some of the mechanisms for user input.

The following table shows these differences

The CognosScript Language	**Word Basic**
StaticComboBox or ComboBox (in the CognosScript Language, these are interchangeable)	ComboBox (Word Basic supports only this syntax)
DropComboBox	N/A

Chapter 2. Objects

You work with the following objects for IBM Cognos PowerPlay OLE automation.

Name	Description
AdvancedQuery Object	Represents an advanced query containing categories based on certain criteria specified in the subset definition.
Application Object	This is the main PowerPlay Application object.
CategoryList Object	Maintains a list of categories from a multidimensional cube.
Child Object	References a specific Child object in a collection.
Column Object	An object used to manipulate a column in a report.
Dimension Object	An object containing a dimension in the DimensionLine object from a PowerCube.
DimensionLine Object	Maintains a list of Dimension objects.
Exception Object	Defines a new exception for a report.
FindQuery Object	Searches categories for a particular string.
Graph Object	Enables the manipulation of displays in a PowerPlay report.
Layer Object	An object used to manipulate a layer in a report.
Level Object	An object used to return a level in a report.
ParentageQuery Object	Performs a query based on levels within a report.
Print Object	An object used to manipulate the printing parameters of a PowerPlay report and initiate a print job.
Range Object	An object used to specify the numerical ranges for exceptions in PowerPlay reports.
Report Object	An object that contains data from one or more cubes.

Name	Description
Row Object	An object used to manipulate a row in a report.
SaveAsPDF Object	Saves a report as a PDF or portable document format file (.pdf).
ValueRestriction Object	Applies restrictions to AdvancedQuery results based on values that exceed or fall below a specified number or values that belong to a specified range.

AdvancedQuery Object

Represents an advanced query containing categories based on certain criteria specified in the subset definition.

Discussion

Use the AdvancedQuery object to specify which categories are to appear in a report using one or more of the following criteria

- a dimension
- a drill-down path
- one or more levels
- one or more qualifiers (optional)
- one or more find queries (optional)

The combination of criteria is known as a query. When a query is placed in a report, the categories that are specified by the criteria are added to the report. Whenever the report is opened, the query is automatically re-executed. Any new categories that fit the specified criteria appear in the report and any categories that no longer match the criteria do not appear in the report.

To maximize the query capabilities, you can use the results of the FindQuery subset in the AdvancedQuery. First create the FindQuery object, and then use the subset data in the AdvancedQuery.

The Item index for the AdvancedQuery and FindQuery objects starts at 1.

The Find queries only apply to the lowest level.

When an AdvancedQuery has more than one level and contains a ValueRestriction query, the value restriction only applies to the lowest level. There can only be one value restriction per AdvancedQuery.

For an AdvancedQuery, do not change the dimension when the query is valid (this change invalidates all the levels).

For an AdvancedQuery, the order and structure of the subset definition are

Name

Dimension

Level

Find

Include

Exclude

ValueRestriction

Execute

AddToReport

Name	Description
AddToReport Method	Adds query results to a report.
Exclude Method	Sets the categories to exclude from the query.
Execute Method	Runs an advanced query on the cube.
Find Method	Specifies the name of the FindQuery object to include in an AdvancedQuery.
Include Method	Sets the categories to include in the query.
Item Method	Returns a category from the AdvancedQuery object.
Level Method	Sets the level used by the AdvancedQuery object to retrieve categories for the query.
Remove Method	Removes the AdvancedQuery object from the ReportQueries collection.
ValueRestriction Method	Returns the value restriction for an AdvancedQuery object.

Name	Description
Application Property	Returns the Application object.
Count Property	Returns a value giving the number of categories that satisfy the Subset.
Dimension Property	Sets or returns the dimension from which categories are returned.

Name	Description
LevelList Property	Returns the list of levels for a specified drill-down path.
Name Property	Sets or returns the name of the subset.
Type Property	Returns a query object type.

Example

This example creates a FindQuery (type1) subset definition which searches for all products that begin with "Star". Then an AdvancedQuery (type 3) subset definition is created using the Find subset definition results. The subset of "Products" beginning with the name "Star" is then added to the report as columns.

```
Sub Main()
   Dim strCubePath As String
   Dim objPPRep As Object
   Dim objFind As Object
   Dim objAdvanced As Object
   strCubePath = "C:\Cubes and Reports\Great Outdoors.mdc"
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.New strCubePath, 1
   objPPRep.ExplorerMode = False
   objPPRep.Visible = True
   Set objFind = objPPRep.ReportQueries.Add(1)
   With objFind
      .Name = "Find Star"
      .Dimension = "Products"
      .SearchShortName = False
      .SearchText = "Star"
      .Pattern = 2
   End With
   Set objAdvanced = objPPRep.ReportQueries.Add(3)
   With objAdvanced
      .Name = "Star Products"
      .Dimension = "Products"
      .Level "Product Id"
      .Find objFind.Name
      .Execute
      .AddToReport 1,1,3
   End With
   Set objAdvanced = Nothing
   Set objFind = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- “FindQuery Object” on page 23
- “ParentageQuery Object” on page 31
- “Report Object” on page 37
- “ReportQueries” on page 63
- “ReportQueries Method” on page 210

Application Object

This is the main IBM Cognos PowerPlay Application object.

Discussion

Use this object to gain control of the application. The Application object lets you start PowerPlay from within an OLE automation script. You can declare an object variable and then use the CreateObject method to create a PowerPlay Application object.

You can select an Application object in the following ways:

- CreateObject("CognosPowerPlay.Application") starts another instance of PowerPlay.
- GetObject (, "CognosPowerPlay.Application") selects the active application.
- Use the Application method with any other object.

Name	Description
Activate Method	Sets the focus on the Application object.
Active Method	Returns the active Report object.
“DeleteAllDataSourceInfo Method” on page 125	Deletes security access information from memory for all PowerCubes.
DeleteAllMDCAccessInfo Method	Deletes PowerCube security access information from memory for all cubes.
DeleteMDCAccessInfo Method	Deletes PowerCube security access information from memory for the specified cube.
Maximize Method	Maximizes the Application object window.
Minimize Method	Minimizes the Application object window.
Quit Method	Exits PowerPlay.
Reports Method	Returns one Report object or the entire collection.
Restore Method	Restores the Application object window to its original size and position.

Name	Description
SetDataSourceInfo Method	Stores security information for a data source in memory.

Name	Description
Application Property	Returns the Application object.
Caption Property	Returns the title of the Application object window.
DefaultAlternateDirectory Property	Sets or returns the directory to save updates to a read-only report.
DefaultCubeDirectory Property	Sets or returns the default path for multidimensional cube files (.mdc).
DefaultMacroDirectory Property	Sets or returns the default path for macro files.
DefaultReportDirectory Property	Sets or returns the default path for PowerPlay report files.
FullName Property	Returns the full name, including the location, of the Application object.
LogonPrompt Property	Sets or returns whether the application prompts for logon or security information.
Name Property	Returns the name of the Application object.
Path Property	Returns the path of the Application object.
RefreshSubCube Property	Sets or returns whether the sub-cube is refreshed automatically.
ShareDimensionLine Property	Sets or returns whether open reports share a dimension line.
UserControl Property	Sets or returns whether the Application object is under user control.
Version Property	Returns the version number of PowerPlay.
Visible Property	Sets or returns whether the Application object is visible to the user.

Example

This example creates an instance of the PowerPlay Application object and shows some its properties to the user. In order to link to a running PowerPlay application, replace the CreateObject function with the GetObject function.

```
Sub Main()
   Dim objPPlayApp as Object
   Set objPPlayApp = CreateObject("CognosPowerPlay.Application")
   objPPlayApp.Visible = 1
   MsgBox "The name of the Application is " &objPPLayApp.Name
   MsgBox "The location of the Application is " _
      &objPPLayApp.Path
   MsgBox "The Application version is " &objPPLayApp.Version
   Set objPPlayApp = Nothing
End Sub
```

CategoryList Object

Maintains a list of categories from a multidimensional cube.

Discussion

Use this object to select categories from a cube or create new categories, and add all or some of these categories to the report.

In order to add categories to a report, first create a CategoryList object by calling the CategoryList method, which is a Report method, and then use it to identify the required categories.

These categories can be existing ones which are found in a cube, or new ones created when the CategoryList object is passed to the report. Select existing categories from the MDC file using the Add method, or create new categories by setting one or more of the following properties to True: Average, Intersection, and Sum.

Name	Description
Add Method (CategoryList)	Adds one or more categories to a CategoryList object.
Remove Method	Removes all categories from the CategoryList object.

Name	Description
Application Property	Returns the Application object.
Average Property	Sets or returns whether to calculate the average of the selected categories in the CategoryList object.

Name	Description
Count Property	Returns the number of categories in the CategoryList object.
Each Property	Sets or returns whether all selected and new categories or just the new categories appear in the Report object.
Intersect Property	Sets or returns whether to determine the values at the intersection of selected categories from different dimensions.
ShareOf Property	Sets or returns whether to show the values in selected categories as a percentage of their higher-level category.
Sum Property	Sets or returns whether to calculate the sum of selected categories.

Example

This example adds categories to the columns and rows in the report.

```
Sub Main()
   Dim objPPRep as Object
   Dim objCatList as Object
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.New "C:\Cubes and Reports\Great " & _
      "Outdoors.mdc", False
   Set objCatList = objPPRep.CategoryList
   objPPRep.Visible = True
   objCatList.Add 1, "Products", "Outdoor Products"
   objPPRep.Columns.Add objCatList
   objCatList.Add 1, "Locations", "Far East"
   objPPRep.Rows.Add objCatList
   Set objCatList = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- Chapter 4, “Methods,” on page 71
- Chapter 5, “Properties,” on page 259

Child Object

References a specific Child object in a collection.

Discussion

Use this object when you want to obtain the name of a child from the collection. The child object lets you isolate categories one level below another category along a drill-down path without the need to know the name of the child category.

You can also use the Item method from the Children collection to obtain the name of a Child object.

Name	Description
Name Property	Sets or returns the name of the Child object.

Example

This example obtains the name of the child of the first category in the Rows collection, then displays its name.

```
Sub Main
   Dim objPPRep As Object
   Dim objFirstRow As Object
   Dim strChild As String
   Set objPPRep = GetObject(,"CognosPowerPlay.Report")
   Set objFirstRow = objPPRep.Rows.Item(1)
   strChild = objFirstRow.Children.Item(1).Name
   MsgBox strChild & " is a child of the " &
_
         objFirstRow.Name & " category.", , "Child
Object"
   Set objFirstRow = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "Children" on page 49
- "Children Method" on page 118

Column Object

An object used to manipulate a column in a report.

Discussion

In order to use this object, first add categories to the report from the CategoryList object or open an existing report.

You can perform calculations, drill-up, drill-down, hide, rank and remove Column objects. You use an index to refer to the position of a Column object in the Columns collection.

Certain methods are only available in Reporter mode.

Name	Description
Accumulation Method	Accumulates all the values of the categories in the Column object.
Activate Method	Sets the focus on the Column object.
Addition Method (Objects)	Adds a constant value or category to the Column object.
AddLowestLevelCategories Method (Reporter)	Adds the lowest-level categories to a report.
Average Method (Objects) (Reporter)	Determines the average between either a constant value or another category and the Column object.
CanDrillDown Method	Returns whether you can drill down the Column object.
CanDrillUp Method	Returns whether you can drill up the Column object.
Children Method	Returns the next child in the hierarchy for the Column object.
CumPercentOfBase Method	Adds a Cumulative Percent of Base Column object using a Row object as the base category.
Division Method	Divides the Column object by either a constant value or another category.
DrillDown Method	Drills down the Column object.
DrillUp Method	Drills up the Column object.
Exponentiation Method	Raises the Column object to the power of either another category or a constant value.
Hide Method	Hides the Column object.
Maximum Method (Objects) (Reporter)	Determines the maximum between the Column object and a constant value or another category.
Minimum Method (Objects) (Reporter)	Determines the minimum between the Column object and a constant value or another category.
Multiplication Method (Objects)	Multiplies a constant value or another category to the Column object.
Percent Method	Adds a percent Column object based on either another category or a constant value.

Name	Description
PercentGrowth Method	Calculates the percentage change between two categories or measures.
PercentOfBase Method	Adds a Percent of Base Column object using a Row object as the base category.
Rank2 Method	Ranks and sorts the Row objects based on the Column object.
Remove Method	Removes the Column object from the report.
Rollup Method	Groups categories containing calculated values to create a new, dynamic calculation.
Select Method	Selects the Column object.
SelectBlank Method	Selects a specific blank column.
Subtraction Method (Objects)	Subtracts a constant value or another category from the Column object, or subtracts the Column object from the category or constant value.
Unselect Method	Unselects the Column object.
UnselectBlank Method	Unselects a specific blank column.

Name	Description
Application Property	Returns the Application object.
CellText Property	Returns the text in a cell.
CellValueAlignment Property	Returns the alignment applied to a cell value in a report.
CellValueFontColor Property	Returns the font color applied to a cell value in a report.
CellValueFontName Property	Returns the name of the font applied to a cell value in a report.
CellValueFontSize Property	Returns the size of the font applied to a cell value in a report.
Exception Property	Sets or returns the exception for the Column object.
Index Property	Returns the position of the Column object in the Columns collection.

Name	Description
IsAlternate Property	Returns whether the drill-down path is primary or alternate.
IsCalculatedCategory Property	Returns whether the category is a calculated category.
LabelAlignment Property	Returns the alignment applied to a category label in a report.
LabelFontColor Property	Returns the font color applied to a category label in a report.
LabelFontName Property	Returns the name of the font applied to a category label in a report.
LabelFontSize Property	Returns the size of the font applied to a category label in a report.
Level Property	Returns the level of the category in a dimension.
Name Property	Sets or returns the name of the Column object.
NestedName Property	Returns the nested name for a category.
ParentCategory Property	Returns the name of the parent category for the object.
Precedence Property	Sets or returns the precedence used in complex calculations.
Style Property	Sets or returns the style used for the Column object.
TopLevelParentCategory Property	Returns the name of the dimension for the object.

Example

This example opens a report, looks for the Column object "Tents" in the Columns collection, changes the name to "Old Tents", and saves the report.

```
Sub Main()
   Dim objPPRep as Object
   Dim objPPCol as Object
   Set objPPRep = CreateObject ("CognosPowerPlay.Report")
   objPPRep.Open "C:\Cubes and Reports\sample1.ppr"
   Set objPPCol = objPPRep.Columns.Item("Tents")
   objPPCol.Name = "Old Tents"
   objPPRep.Save
```

```
   Set objPPCol = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "CategoryList Object" on page 13

Dimension Object

An object containing a dimension in the DimensionLine object from a PowerCube.

Discussion

Use this object to change the default settings for dimensions in the DimensionLine object and to determine the properties of the dimension. Each Dimension object includes different levels of categories. You use the object to reference an item in a row, column or layer.

Special categories and alternate drill paths are also recognized by OLE automation. For example, a dimension "Years" may have the two regular categories "1995" and "1996", and three other categories named "Current Month", "QTD", and "Last Month" that represent special categories and alternate drill paths.

Performing a count on the "Years" Dimension

```
Msgbox Report.DimensionLine.Item("Years").Count
```

will return "5" because the regular categories and special categories are all visible to OLE automation.

It is possible to filter on a special category by naming it as

```
Report.DimensionLine.Item("Years").Change("Current Month")
```

Name	Description
Change Method	Changes the current category for the Dimension object.
ChangeToParent Method	Changes the current category for the Dimension object to the category one level higher.
ChangeToTop Method	Changes the current category for the Dimension object to top-level category.
Children Method	Returns the next child category in the hierarchy for the current dimension.
HasParent Method	Returns whether the current category has a parent.
Levels Method	Returns all levels available in the dimension for a category.
Parent Method	Returns the name of the parent category.

Name	Description
Application Property	Returns the Application object.
BlankWhenDividedByZero Property	Set or returns whether a numeric value divided by zero appears as zero or blanks.
BlankWhenMissing Property	Sets or returns whether missing numeric values appear as zero or blanks.
BlankWhenZero Property	Sets or returns whether zero numeric values are shown as zeros or blanks.
Count Property	Returns the number of categories one level below the current category.
Index Property	Returns the position of the Dimension object in the DimensionLine object.
IsAlternate Property	Returns whether the drill-down path is primary or alternate.
IsCalculatedCategory Property	Returns whether the category is a calculated category.
Level Property	Returns the level of the category in a dimension.
Measure Property	Sets or returns the value and symbol for a specified currency.
Name Property	Sets or returns the name of the Dimension object.
Visible Property	Sets or returns whether the Dimension Line object is visible to the user.

Example

This example changes two of the current categories (one from the Year dimension, the other from the Product dimension) for the DimensionLine object in the current report.

```
Sub Main()
   Dim objPPRep as Object
   Dim objPPDim1 as Object
   Dim objPPDim2 as Object
   Set objPPRep = GetObject("c:\cognos\change.ppr")
   Set objPPDim1 = objPPRep.DimensionLine.Item("Years")
   Set objPPDim2 = objPPRep.DimensionLine.Item("Products")
   objPPDim1.Change ("1996")
   objPPDim2.Change ("Outdoor Products")
   Set objPPDim2 = Nothing
```

```
   Set objPPDim1 = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "DimensionLine Object"

DimensionLine Object

Maintains a list of Dimension objects.

Discussion

The DimensionLine object includes the categories used to filter data from each dimension in the cube. The DimensionLine object contains Dimension objects.

You can automate changes to the appearance of the dimension line for a report. You can make the DimensionLine invisible to the user either when you open a report or after the report opens.

Name	Description
Item Method	Returns a Dimension object from the DimensionLine object.

Name	Description
Application Property	Returns the Application object.
Count Property	Returns the number of Dimension objects in the DimensionLine object.
Visible Property	Sets or returns whether the DimensionLine object is visible to the user.

Example

This example changes two of the current categories for the DimensionLine object in the current report.

```
Sub Main()
   Dim objPPRep as Object
   Set objPPRep = GetObject("C:\Cubes and Reports\Sample.ppr")
   objPPRep.DimensionLine.Item("Years").Change ("1995")
   objPPRep.DimensionLine.Item("Products").Change _
       ("Go Sport Line")
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "Dimension Object" on page 19

Exception Object

Defines a new exception for a report.

Discussion

This object is the basis for identifying exceptions within the report data. An exception enables you to highlight information in a report that meets a specified criteria. You must define a range to which the exception is applied and the style to apply to the exception. Use the UpperBoundary and LowerBoundary methods to determine the range to apply formatting when the information in the report meets the conditions set by the exception range. After you define an Exception object, it can be applied to categories.

Name	Description
Ranges Method	Returns one Range object or the entire collection.
Remove Method	Removes the Exception object from the Report object.
SetDrivingCategory Method	Sets the driving category for the Exception object.
SetMacro Method	Sets the name and style of the macro used in the Exception object.

Name	Description
Application Property	Returns the Application object.
DrivingCategory Property	Returns the driving category for the Exception object.
DrivingDimension Property	Returns the driving dimension for the Exception object.
MacroName Property	Sets or returns the name of the macro associated with an Exception object.
MacroStyle Property	Sets or returns the name of the style associated with the macro.
Name Property	Sets or returns the name of the Exception object.

Example

This example opens a report and displays the driving fields for the first Exception object.

```
Sub Main()
   Dim objPPRep as Object
```

```
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.Open "C:\Cubes and Reports\Exception.ppr"
   MsgBox "Driving Category:" & _
      objPPRep.Exceptions.Item(1).DrivingCategory
   MsgBox "Driving Dimension:" & _
      objPPRep.Exceptions.Item(1).DrivingDimension
   Set objPPRep = Nothing
End Sub
```

Related Topics

- Chapter 4, "Methods," on page 71
- Chapter 5, "Properties," on page 259

FindQuery Object

Searches categories for a particular string.

Discussion

Use this object to locate an individual category or all categories matching a specified SearchText string. When specifying a SearchText string in the subset definition, you can also specify advanced patterns using the Patterns property.

When you use FindQuery to search for data within a report, you can search for the category that matches your search criteria. If you use FindQuery to search for data within a cube, it creates a query that enables you to locate all categories that match your specified search criteria. You can then use these FindQuery results within an AdvancedQuery object.

A FindQuery search does not include hidden categories inside the report.

The Item index for the FindQuery object starts at 1.

For a FindQuery, the order and structure of the subset definition (the components of the FindQuery object) are

- Name
- Dimension
- SearchShortName
- SearchDescription
- SearchText
- Pattern
- Execute
- AddToReport

If you want to use FindQuery in combination with AdvancedQuery object, define the FindQuery first.

Name	Description
AddToReport Method	Adds query results to a report.

Name	Description
Execute Method	Runs the find operation on a cube.
Item Method	Returns the FindQuery object from the ReportQueries collection.
Remove Method	Removes the FindQuery object from the ReportQueries collection.

Name	Description
Application Property	Returns the Application object.
Count Property	Returns a value giving the number of categories that satisfy the subset.
Dimension Property	Sets or returns the dimension from which categories are returned.
Name Property	Sets or returns the name of the subset.
Pattern Property	Sets search criteria for a subset definition.
SearchDescription Property	Sets or returns whether the FindQuery object searches the category descriptions in a cube.
SearchShortName Property	Sets or returns whether the FindQuery object searches short or long category names.
SearchText Property	Sets or returns the search string used in the subset definition of a FindQuery query.
Type Property	Returns a query object type.

Example

This example creates a FindQuery subset definition that searches for all products that begin with the name "Star".

```
Sub Main()
   Dim strCubePath As String
   Dim objPPRep As Object
   Dim objFind As Object
   strCubePath = "C:\Cubes and Reports\Great Outdoors.mdc"
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.New strCubePath, 1
   objPPRep.ExplorerMode = False
   objPPRep.Visible = True
   Set objFind = objPPRep.ReportQueries.Add(1)
   With objFind
```

```
      .Name = "Find Star"
      .Dimension = "Products"
      .SearchShortName = False
      .SearchText = "Star"
      .Pattern = 2
   End With
   Set objFind = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "AdvancedQuery Object" on page 8
- "ParentageQuery Object" on page 31
- "ReportQueries Method" on page 210
- "Reports" on page 65

Graph Object

Enables the manipulation of displays in a IBM Cognos PowerPlay report.

Discussion

Graphical objects communicate comparisons, relationships, and trends. When you have a large amount of data in a report, you can use a Graph object as a different means of representing the data. You can also use the Graph object attributes to present data more effectively, such as by using the swap methods to swap layers, rows and columns in the report. You can add and remove Graph objects and determine the type of display selected.

Name	Description
Activate Method	Sets the focus on the Graph object.
Depth Method	Returns whether the Graph object is three-dimensional (3D).
Remove Method	Removes the Graph object from the report.
SetType Method	Sets the Graph object type.
Vertical Method	Returns whether the Graph object is a vertical display.

Name	Description
Application Property	Returns the Application object.
DataGridlines Property	Sets or returns whether the gridline settings are on or off for a crosstab.

Name	Description
EnableUserColumnSummaryLabel Property	Sets or returns whether a user-defined label is used for the innermost summary column in a nested crosstab.
EnableUserRowSummaryLabel Property	Sets or returns whether a user-defined label is used for the innermost summary row in a nested crosstab.
HideRankCategory Property	Sets or returns whether the rank category is hidden.
IndentTotalsLevel Property	Sets or returns the current indent level for summary cells in a nested crosstab.
Index Property	Returns the position of the Graph object in the Graphs collection.
KeepSummaryVisible Property	Sets or returns whether the summary category will remain visible on all scrolled pages.
LabelGridlines Property	Sets or returns whether the gridlines are on or off for category labels in a nested crosstab.
Layout Property	Sets or returns the current layout style in a nested crosstab.
MaxPrintedBars Property	Sets or returns the maximum number of bars on a single printed page.
MaxVisibleBars Property	Sets or returns the maximum number of bars visible on a single page of scrolled data.
NamesShown Property	Sets or returns whether category names appear beside pie chart slices.
NestedCharts Property (Explorer)	Sets or returns whether multiple charts that represent summarized data appear in one display.
ShowSummaryBreakdown Property (Explorer)	Sets or returns whether to show the breakdown of summary rows and columns in a crosstab.
ShowSummaryColumn Property (Explorer)	Sets or returns whether to show the summary column.
ShowSummaryRow Property (Explorer)	Sets or returns whether to show the summary row.
ShowTies Property	Sets or returns whether to show label ties.

Name	Description
StatsLineCaption Property	Sets or returns the caption for a given statistical line on a graph.
StatsLineColor Property	Sets or returns the color for a given statistical line on a graph
StatsLineOn Property	Sets or returns a statistical line on a graph.
StatsLineStyle Property	Sets or returns the line style of a given statistical line on a graph.
StatsLineUserValue Property	Sets a custom value for a statistical line on a graph.
Type Property	Returns the Graph object type.
UserColumnSummaryLabel Property	Sets or returns the user-defined label for the innermost summary column in a nested crosstab.
UserRowSummaryLabel Property	Sets or returns the user-defined label for the innermost summary row in a nested-crosstab.
UseScrolling Property	Sets or returns whether scrolling is enabled.
ValuesAutoFit Property	Sets or returns whether value labels fit within graph bars and pie segments.
ValuesFontColor Property	Sets or returns the font color used for the graph value labels.
ValuesFontName Property	Sets or returns the font name used for the value labels.
ValuesFontSize Property	Sets or returns the font size used for the graph value labels.
ValuesFontStyle Property	Sets or returns the font style used for the graph value labels.
ValuesPosition Property	Sets or returns the position of value labels on some graph types.
ValuesShown Property	Sets or returns whether value labels appear next to pie chart slices.

Example

This example opens a report, and shows the display type for the first display in the report.

```
Sub Main()
```

```
   Dim objPPRep as Object
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.Open "C:\Cubes and Reports\Graph.ppr"
   MsgBox "The type of graph is" & _
      objPPRep.Graphs.Item(1).Type & "."
   Set objPPRep = Nothing
End Sub
```

Related Topics

- Chapter 4, "Methods," on page 71
- Chapter 5, "Properties," on page 259

Layer Object

An object used to manipulate a layer in a report.

Discussion

The Layer object represents the third set of categories, along with columns and rows that you can add to a report. You can perform calculations, drill-up, drill-down, hide, rank and remove Layer objects. You use an index to refer to the position of a Layer object in the Layers collection.

To use this object, first add categories to the report from the CategoryList object or open an existing report.

Certain methods are only available in Reporter mode.

Name	Description
Activate Method	Sets the focus on the Layer object.
Addition Method (Objects)	Adds a constant value or a category to the Layer object.
AddLowestLevelCategories Method (Reporter)	Adds the lowest-level categories to a report.
Average Method (Objects) (Reporter)	Determines the average between either a constant value or another category and the Layer object.
CanDrillDown Method	Returns whether you can drill down the Layer object.
CanDrillUp Method	Returns whether you can drill up the Layer object.
Children Method	Returns the next child in the hierarchy for the Layer object.
Division Method	Divides the Layer object by either a constant value or another category.

Name	Description
DrillDown Method	Drills down the Layer object.
DrillUp Method	Drills up the Layer object.
Exponentiation Method	Raises the Layer object to the power of either another category or a constant value.
Maximum Method (Objects) (Reporter)	Determines the maximum between the Layer object and a constant value or another category.
Minimum Method (Objects) (Reporter)	Determines the minimum between the Layer object and a constant value or another category.
Multiplication Method (Objects)	Multiplies a constant value or another category to the Layer object.
Percent Method	Adds a percent Layer object based on either another category or a constant value.
Remove Method	Removes the Layer object from the Report object.
Rollup Method	Groups categories containing calculated values to create a new, dynamic calculation.
Select Method	Selects the Layer object.
Subtraction Method (Objects)	Subtracts a constant value or another category from the Layer object and the reverse.
Unselect Method	Unselect the Layer object.

Name	Description
Application Property	Returns the Application object.
Index Property	Returns the position of the Layer object in the Layers collection.
IsAlternate Property	Returns whether the drill-down path is primary or alternate.
IsCalculatedCategory Property	Returns whether the category is a calculated category.
LabelFontColor Property	Returns the font color applied to a category label in a report.

Name	Description
LabelFontName Property	Returns the name of the font applied to a category label in a report.
LabelFontSize Property	Returns the size of the font applied to a category label in a report.
Level Property	Returns the level of the category in a dimension.
NestedName Property	Returns the nested name for a category.
ParentCategory Property	Returns the name of the parent category for the object.
Precedence Property	Sets or returns the precedence used in complex calculations.
Style Property	Sets or returns the style used for the Layer object.
TopLevelParentCategory Property	Returns the name of the dimension for the object.

Example

This example gets an open report, removes the first and second layer, subtracts the new second layer from the new first layer, and performs maximum calculation on all the layers.

```
Sub Main()
   Dim objPPRep AS Object
   Set objPPRep = GetObject("C:\Cubes and Reports\Sample.ppr"
)
   objPPRep.Visible = 1
   objPPRep.ExplorerMode = 0
   objPPRep.Layers.Subset(1,2).Remove
   objPPRep.Layers.Subset(1,2).Subtraction
   objPPRep.Layers.Maximum
   objPPRep.Save
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "CategoryList Object" on page 13

Level Object

An object used to return a level in a report.

Discussion

A level is an object containing common or default attributes for all of its member categories. Drilling down on a dimension drills down on categories from one level to another. Data in each Dimension object is organized into a series of Level objects. Each dimension may contain one or more levels of data. Use the Item method to access a Level object in the Levels collection.

Name	Description
Application Property	Returns the Application object.
Name Property	Returns the name of the Level object.

Example

This example uses the Item method from the Levels collection to access the Level object for a dimension.

```
Sub Main
   Dim objPPRep As Object
   Dim objDimension As Object
   Dim objLevel As Object
   Dim intx As Integer
   Dim strLevelList As String
   Set objPPRep = GetObject(,"CognosPowerPlay.Report")
   Set objDimension = objPPRep.DimensionLine.Item(1)
   For intx = 1 to objDimension.Levels.Count
      Set objLevel = objDimension.Levels.Item(intx)
      strLevelList = strLevelList & chr$(10) &
objLevel.Name
      Set objLevel = Nothing
   Next intx
   MsgBox "The levels in the " & objDimension.Name
& _
         " dimension are:" & chr$(10) & strLevelList
   Set objDimension = Nothing
   Set objPPRep = Nothing
End Sub
```

ParentageQuery Object

Performs a query based on levels within a report.

Discussion

Use this object to query a report based on the level specified in the subset definition.

There are several variations of the ReportQuery object: AdvancedQuery, FindQuery, and ParentageQuery objects. A subset is the result of a query. It is a group of selected categories added to a Report object. For the ParentageQuery object, a subset is all the categories for a specified level.

The Item index for the ParentageQuery object starts at 1.

For a ParentageQuery, the order and structure of the subset definition (the components of the ParentageQuery object) are

Name

Category

LevelsDown

LowestLevel

Execute

AddToReport

Name	Description
AddToReport Method	Adds query results to a report.
Category Method	Sets the parent category or dimension for the ParentageQuery subset definition.
Execute Method	Runs a parentage query on a cube.
Item Method	Returns the ParentageQuery object from the ReportQueries collection.
Remove Method	Removes the ParentageQuery object from the ReportQueries collection.

Name	Description
Application Property	Returns the Application object.
Count Property	Returns a value giving the number of categories that satisfy the subset.
LevelsDown Property	Sets the number of levels down the hierarchy for specifying the next level Parentage subsets.
LowestLevel Property	Sets whether the query uses the next lower level or lowest level of the parent category.
Name Property	Sets or returns the name of the subset.
Type Property	Returns a query object type.

Example

This example creates a ParentageQuery (type 2) subset definition that returns all categories one level below Channels. Then the categories that are one level below Channels are added to the report as the first nesting level of rows.

```
Sub Main()
   Dim strCubePath As String
   Dim objPPRep As Object
   Dim objCategory As Object
   Dim objParent As Object
   strCubePath = "C:\Cubes and Reports\Great Outdoors.mdc"
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.New strCubePath, 1
   objPPRep.ExplorerMode = False
   objPPRep.Visible = True
   Set objParent = objPPRep.ReportQueries.Add(2)
   With objParent
      .Name = "Sales Channels"
      .Category "Channels"
      .LowestLevel = False
      .LevelsDown = 1
      .Execute
      .AddToReport 0,1,6
   End With
   Msgbox "The first category added was " & _
      objParent.Item(1).Name & ".", ,"Subset"
   Set objParent = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "AdvancedQuery Object" on page 8
- "FindQuery Object" on page 23
- "Report Object" on page 37
- "ReportQueries" on page 63
- "ReportQueries Method" on page 210

Print Object

An object used to manipulate the printing parameters of a IBM Cognos PowerPlay report and initiate a print job.

Discussion

This object is the basis for determining how a report prints. Through automation, you can set printing options such as

- which layers to print
- which rows to print
- which selected displays to print when showing more than one display
- which parts of the display to print

- the page layout view
- colors as patterns
- which pages to print
- the maximum number of pages to print
- the number of copies to print

Name	Description
PrintOut Method	Prints the Report object.
ResetPrintOptionsToDefault Method	Resets print options back to the default settings.
SetChartToPrint Method	Specifies which Graph object of a report to print.
SetListOfLayersToPrint Method	Specifies the range of layers of the report to print.
SetListOfRowsToPrint Method	Specifies the range of rows of the report to print.

Name	Description
Application Property	Returns the Application object.
AxisOnAllPages Property	Sets or returns whether the axis and labels appear on every page of the printed report.
ChartTitleOnAllPages Property	Sets or returns whether titles appear on every page of the printed report.
Collate Property	Sets or returns whether the Report object collates during printing.
Copies Property	Sets or returns the number of copies to print.
FitToPage Property	Sets or returns whether the report is scaled to fit on one page.
IncludeLegend Property	Sets or returns whether the legend prints with the report.
PrintAllCharts Property	Sets or returns whether all displays print on the same page.
PrintColorsAsPatterns Property	Sets or returns whether colors print as patterns or as colors.
PrintEntireReport Property	Sets or returns whether to print the entire report, including all displays, layers, and rows.

Name	Description
PrintPageLayout Property	Sets or returns whether to print all displays visible in the page layout or page width view on the same page.
PrintSelectedDisplay Property	Sets or returns whether to print the selected or currently active Graph object.
SummaryColumnOnAllPages Property	Sets or returns whether to show the summary column category on every page of a printed report.
SummaryRowOnAllPages Property	Sets or returns whether to show the summary row category on every page of a printed report.

Example

This example opens a report and prints one copy of all the data for the first graphical display only.

```
Sub Main()
   Dim objPPRep as Object
   Dim objRepPrt as Object
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.Open "C:\Cubes and Reports\Sample1.ppr"
   Set objRepPrt = objPPRep.Print
   objRepPrt.PrintAllCharts = False
   objRepPrt.SetListOfRowsToPrint objPPrep.Rows
   objRepPrt.SetListOfLayersToPrint objPPRep.Layers
   objRepPrt.SetChartToPrint objPPRep.Graphs.Item(1)
   objRepPrt.IncludeLegend = False
   objRepPrt.ChartTitleOnAllPages = True
   objRepPrt.SummariesOnAllPages = True
   objRepPrt.AxisOnAllPages = True
   objRepPrt.Collate = True
   objRepPrt.PrintOut
   Set objRepPrt = Nothing
   Set objPPrep = Nothing
End Sub
```

Related Topics

- Chapter 4, “Methods,” on page 71
- Chapter 5, “Properties,” on page 259

Range Object

An object used to specify the numerical ranges for exceptions in IBM Cognos PowerPlay reports.

Discussion

You can set lowest and highest values for use with Exception objects. The values in this object determine whether a cell value has failed an exception.

You can group the Range objects in an Exception object into a Ranges collection. Use a Ranges collection to operate on the entire group of Range objects at once rather than one at a time.

Name	Description
Remove Method	Removes the Range object from the Exception object.

Name	Description
Application Property	Returns the Application object.
Index Property	Returns the position of the Range object in the Ranges collection.
LowerBoundary Property	Sets or returns the value defined for the lower boundary of the Range object.
Style Property	Sets or returns the name of the style defined for the Range object.
UpperBoundary Property	Sets or returns the value defined for the upper boundary of the Range object.

Example

This example shows how to return a Range object and returns the value defined for the lower boundary of the range in an Exception object.

```
Sub Main
   Dim objPPRep As Object
   Dim objPPRange As Object
   Set objPPRep = GetObject("C:\Cubes and Reports\Exception.ppr")
   objPPRep.Visible = 1
   Set objPPRange = objPPRep.Exceptions.item(1).Ranges.Item(1)
   MsgBox "Lower boundary is " &objPPRange.LowerBoundary
   Set objPPRange = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- Chapter 4, “Methods,” on page 71
- Chapter 5, “Properties,” on page 259

Report Object

An object that contains data from one or more cubes.

Syntax A

GetObject(pathname)

Syntax B

GetObject(, class)

Parameter	Description
PathName	Required. Specifies the path and file name for the object to retrieve. Type: String
Class	Required. Specifies a string containing the class of the object, "CognosPowerPlay.Report". Type: String

Name	Description
Activate Method	Sets the focus on the Report object.
AddBlanks Method (Reporter)	Adds a single blank row or column to a nested crosstab.
CategoryList Method	Creates a CategoryList object used to identify categories to be inserted in the Report object.
CellValue Method	Gets the value of a cell in a Report object.
Close Method	Closes the Report object.
Columns Method	Returns a collection that contains all the Column objects.
Copy Method	Copies the Row, Column, or Layer objects currently selected in the Report object to the Clipboard.
Cut Method (Reporter)	Moves the Row, Column, or Layer object currently selected in the Report object to the Clipboard.
DeleteSelected Method	Deletes selected objects in a collection.

Name	Description
DimensionLine Method	Returns a DimensionLine object for the current Report object.
Exceptions Method	Returns one Exception object or the entire collection.
FindNext Method	Finds the next matching category label in a report.
FindPrevious Method	Finds the previous matching category label in a report.
Forecast Method (Explorer)	Creates a specified number of forecast categories based on the existing time dimensions.
GetDataNow Method	Updates the data in the Report object.
Graphs Method	Returns one Graph object or the entire collection.
HideSelected Method	Hides selected objects in a collection.
HideUnselected Method	Hides any object in a collection that is not selected.
Layers Method	Returns one Layer object or the entire collection.
Maximize Method	Maximizes the Report object window.
Minimize Method	Minimizes the Report object window.
Paste Method (Reporter)	Pastes the contents of the Clipboard into the selected categories of the Report object.
PDFFile Method	Sets the file name of the PDF object when it is saved.
Print Method	Returns a Print object.
PublishToPortal Method	Creates a report in the IBM Cognos Analytics content store. Users access the report using the Cognos Analytics portal.
ReportQueries Method	Returns a ReportQueries collection.
Restore Method	Restores the Report object window to its original size and position.
Rows Method	Returns one Row object or the entire collection.

Name	Description
Save Method	Saves one or all Report objects.
Select Method	Selects all the categories in the Report object.
SizeSelected Method	Applies a size to selected objects.
StyleSelected Method	Applies a style to the selected object.
SwapColumnsAndLayers Method	Exchanges the positions of the Column objects and Layer objects.
SwapRowsAndColumns Method	Exchanges the positions of the Row objects and Column objects.
SwapRowsAndLayers Method	Exchanges the positions of the Row objects and Layer objects.
UnhideAllCategories Method	Makes all hidden categories visible.
Unselect Method	De-selects all the categories in the Report object.
UpdatePublishedReport Method	Updates a report previously published to the IBM Cognos Analytics content store.

Name	Description
Application Property	Returns the Application object.
AutomaticExceptions Property	Sets or returns whether automatic highlighting of exceptions is on or off.
AutomaticExceptionSensitivity Property	Sets or returns the exceptional highlighting sensitivity.
CalculatedCategories Property	Sets or returns whether calculated categories are on or off, or whether a PowerCube contains calculated categories.
CubeName Property	Returns the file name of the cube for the active report.
ExplorerMode Property	Sets or returns whether the Report object is an Explorer or Reporter report.
FooterText Property	Sets or returns the text in the footer of a report.
FullName Property	Returns the full name, including the location, of the Report object.

Name	Description
GetDataAutomatically Property	Sets or returns whether the Report object retrieves data automatically each time it is modified.
HeaderText Property	Sets or returns the text in the header of a report.
Index Property	Returns the position of the Report object in the Reports collection.
Name Property	Returns the name of the Report object.
Path Property	Returns the path of the Report object.
Saved Property	Returns whether the Report object has been saved.
ShareDimensionLine Property	Sets or returns whether open reports share a dimension line.
ShowValuesAs Property (Explorer)	Sets or returns how to show values in a report.
Suppress8020 Property (Explorer)	Sets or returns the 80/20 suppress mode for report dimensions.
SuppressZeros Property	Sets or returns the suppress mode for the Report object.
TitleText Property	Sets or returns the text in the title of a report.
Visible Property	Sets or returns whether the Report object is visible to the user.

Discussion

This object is the basis for building and managing a report. Use this object to set and operate on the attributes of a report. Each Report object is a separate PowerPlay report.

Use the GetObject function when there is a current instance of the Report object or if you want to create the object with a file already loaded. If there is no current instance and you do not want the object started with a file loaded, use the CreateObject function.

Use only the following syntax to retrieve an existing PowerPlay report with the GetObject function. Using other syntax will cause an error.

Example

This example opens an IBM Cognos PowerPlay report and sets a number of properties.

```
Sub Main()
   Dim objPPRep As Object
   Set objPPRep = CreateObject ("CognosPowerPlay.Report")
   objPPRep.Open ("C:\Cubes and Reports\Sample1.ppr")
   objPPRep.ExplorerMode = True
   objPPRep.Visible = True
   objPPRep.ShowValuesAs = 2
   objPPRep.ShareDimensionLine = 1
   objPPRep.Suppress8020 = 3
   objPPRep.SuppressZeros = 3
   objPPRep.CalculatedCategories = True
   Set objPPRep = Nothing
End Sub
```

Related Topics

- Chapter 4, "Methods," on page 71
- Chapter 5, "Properties," on page 259

Row Object

An object used to manipulate a row in a report.

Discussion

A Row object is a category that shows related information in a horizontal list. To use this object, first add categories to the report from the CategoryList object or open an existing report.

You can perform calculations, drill-up, drill-down, hide, rank and remove Row objects. You use an index to refer to the position of a Row object in the Rows collection.

Certain methods are only available in Reporter mode.

Name	Description
Accumulation Method	Accumulates all the values of the categories in the Row object.
Activate Method	Sets the focus on the Row object.
Addition Method (Objects)	Adds a constant value or a category to the Row object.
AddLowestLevelCategories Method (Reporter)	Adds the lowest-level categories to a report.

Name	Description
Average Method (Objects) (Reporter)	Determines the average between either a constant value or another category and the Row object.
CanDrillDown Method	Returns whether you can drill down the Row object.
CanDrillUp Method	Returns whether you can drill up the Row object.
Children Method	Returns the next child in the hierarchy for the Row object.
CumPercentOfBase Method	Adds a Cumulative Percent of Base Row object using a Column object as the base category.
Division Method	Divides the Row object by either a constant value or another category.
DrillDown Method	Drills down the Row object.
DrillUp Method	Drills up the Row object.
Exponentiation Method	Raises the Row object to the power of either another category or a constant value.
Hide Method	Hides the Row object.
Maximum Method (Objects) (Reporter)	Determines the maximum between either a constant value or another category and the Row object.
Minimum Method (Objects) (Reporter)	Determines the minimum between either a constant value or another category and the Row object.
Multiplication Method (Objects)	Multiplies a constant value or another category by the Row object.
Percent Method	Adds a percent Row object based on either another category or a constant value.
PercentGrowth Method	Calculates the percentage change between two categories or measures.
PercentOfBase Method	Adds a Percent of Base Row object using a Column object as the base category.
Rank2 Method	Ranks and sorts the Column objects based on the Row object.
Remove Method	Removes the Row object from the report.

Name	Description
Rollup Method	Groups categories containing calculated values to create a new, dynamic calculation.
Select Method	Selects the Row object.
SelectBlank Method	Selects a specific blank row.
Subtraction Method (Objects)	Subtracts a constant value or another category from the Row object, or subtracts the Row object from the category or constant value.
Unselect Method	De-selects the Row object.
UnselectBlank Method	Unselects a specific blank row.

Name	Description
Application Property	Returns the Application object.
CellText Property	Returns the text in a cell.
CellValueAlignment Property	Returns the alignment applied to a cell value in a report.
CellValueFontColor Property	Returns the font color applied to a cell value in a report.
CellValueFontName Property	Returns the name of the font applied to a cell value in a report.
CellValueFontSize Property	Returns the size of the font applied to a cell value in a report.
Exception Property	Sets or returns the exception for the Row object.
Index Property	Returns the position of the Row object in the Rows collection.
IsAlternate Property	Returns whether the drill-down path is primary or alternate.
IsCalculatedCategory Property	Returns whether the category is a calculated category.
LabelAlignment Property	Returns the alignment applied to a category label in a report.
LabelFontColor Property	Returns the font color applied to a category label in a report.

Name	Description
LabelFontName Property	Returns the name of the font applied to a category label in a report.
LabelFontSize Property	Returns the size of the font applied to a category label in a report.
Level Property	Returns the level of the category in a dimension.
Name Property	Sets or returns the name of the Row object.
NestedName Property	Returns the nested name for a category.
ParentCategory Property	Returns the name of the parent category for the object.
Precedence Property	Sets or returns the precedence used in complex calculations.
Style Property	Sets or returns the style used for the Row object.
TopLevelParentCategory Property	Returns the name of the dimension for the object.

Example

This example opens a report, looks for the Column object "1996" in the Columns collection, changes the name to "Last Year", and saves the report.

```
Sub Main()
   Dim objPPRep as Object
   Set objPPRep = CreateObject ("CognosPowerPlay.Report")
   objPPRep.Open "C:\Cubes and Reports\Sample1.ppr"
   objPPRep.Rows.Item("1996").Name = "Last Year"
   objPPRep.Save
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "CategoryList Object" on page 13

SaveAsPDF Object

Saves a report as a PDF or portable document format file (.pdf).

Discussion

This object is the foundation for saving a report in PDF format. Use this object to set and operate on the attributes of a report. A PDF is useful for distributing standard reports using Adobe Reader and provides quality multipage output. PDFs

are compact, portable, and platform-independent so you can capture the original look and feel of a document (including all fonts, images, graphics, and formatting).

Name	Description
Save Method	Saves the report as a PDF.
SetChartToSave Method	Specifies which Graph object to save in a PDF.
SetListOfLayersToSave Method	Specifies the range of layers to save in a PDF.
SetListOfRowsToSave Method	Specifies the range of rows to save in a PDF.

Name	Description
Application Property	Returns the Application object.
AxisOnAllPages Property	Sets or returns whether the axis and labels appear on every page of the printed report or PDF.
ChartTitleOnAllPages Property	Sets or returns whether the display titles appear on every page of the PDF.
IncludeLegend Property	Sets or returns whether the legend appears in a PDF.
SaveAllCharts Property	Sets or returns whether all Graph objects in a report are saved in portable document format (.pdf).
SaveEntireReport Property	Sets or returns whether to save the entire report as a PDF.
SummariesOnAllPages Property	Sets or returns whether to show the summary column category on every page of a PDF.
SummaryRowOnAllPages Property	Sets or returns whether to show the summary row category on every page of a PDF.

Example

This example opens a report, sets options for saving the report, and then saves the report as a PDF.

```
Sub Main()
   Dim objPDF as Object
   Dim objPPRep as Object
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
```

```
   objPPRep.Open( "c:\Cognos\sample.ppr" )
   objPPRep.visible( TRUE )
   Set objPDF = objPPRep.PDFFile( "c:\Cognos\PDFSample"
, True )
   With objPDF
      .SaveEntireReport = False
      .AxisOnAllPages = True
      .ChartTitleOnAllPages = False
      .IncludeLegend = True
      .SetChartToSave objPPRep.Graphs.Item(1)
      .SetListOfLayersToSave objPPRep.Layers
      .SetListOfRowsToSave objPPRep.Rows
   End With
   objPDF.Save
   Set objPPRep = Nothing
   Set objPDF = Nothing
End Sub
```

ValueRestriction Object

Applies restrictions to AdvancedQuery results based on values that exceed or fall below a specified number or values that belong to a specified range.

Discussion

Use this object to limit the number of categories for the results of an AdvancedQuery.

A ValueRestriction object must be defined before it can be used by the AdvancedQuery object. You can only use a ValueRestriction with an AdvancedQuery object. Once you define a ValueRestriction and use it with an AdvancedQuery object, running the advanced query will apply the ValueRestriction to the query results.

When an AdvancedQuery object that has more than one level uses a ValueRestriction query, the value restriction only applies to the lowest level.

For a ValueRestriction, the order of the components is important. The Dimension property must be set before the Measure, Operator, Operand1, Operand2, and Count properties and the DimensionFilter method. The Name property can be set anywhere within the filter definition.

If you use values from a measure based on a percentage, you must use the decimal format when entering a value. For example, if you restrict an AdvancedQuery to profit margin values greater than 20 percent, use .20.

Name	Description
DimensionFilter Method	Sets the filter category for an indexed dimension.

Name	Description
Application Property	Returns the Application object.
Count Property	Sets or returns the number of categories the ValueRestriction results should contain when the operator is either "Smallest" or "Largest".
Dimension Property	Sets or returns the dimension from which categories are returned.
DimensionSettings Property	Returns a comma-separated string of all the dimension line settings for the ValueRestriction.
Measure Property	Sets or returns the name of measure whose values are used for a value restriction.
Name Property	Sets or returns the name of the ValueRestriction object.
Operand1 Property	Sets or returns the value used to compare report cell values based on a specified operator.
Operand2 Property	Sets or returns the second value when the Between operator is used to specify a range.
Operator Property	Sets or returns the operator used for a value restriction.

Example

This example creates an advanced subset that selects countries or regions from the Locations dimension. The value restriction (type 4) limits the results to return only those countries or regions whose Revenue values for Sports Chains are between 25,000 and 100,000.

```
Sub Main()
   Dim strCubePath As String
   Dim objPPRep As Object
   Dim objValue As Object
   Dim objAdvanced As Object
   strCubePath = "C:\Cubes and Reports\Great Outdoors.mdc"
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.New strCubePath, 1
   objPPRep.ExplorerMode = False
   objPPRep.Visible = True
   Set objValue = objPPRep.ReportQueries.Add(4)
   With objValue
      .Name = "25000-100000"
      .Dimension = "Locations"
      .Measure = "Revenue"
```

```
      .Operator = "between"
      .Operand1 = 25000
      .Operand2 = 100000
      .DimensionFilter 4, "Sports Chain"
   End With
   Set objAdvanced = objPPRep.ReportQueries.Add(3)
   With objAdvanced
      .Name = "Locations"
      .Dimension = "Locations"
      .Level "Country or Region"
      .ValueRestriction objValue.Name
      .Execute
      .AddToReport 0,1,3
   End With
   Set objPPRep = Nothing
   Set objAdvanced = Nothing
   Set objValue = Nothing
End Sub
```

Related Topics

- "AdvancedQuery Object" on page 8
- "ParentageQuery Object" on page 31
- "ReportQueries Method" on page 210
- "Reports" on page 65

Chapter 3. Collections

You work with the following collections for IBM Cognos PowerPlay OLE automation.

Name	Description
Children	Maintains a list of child objects.
Columns	Maintains a list of Column objects in a report.
Exceptions	Maintains a list of Exception objects in a report.
Graphs	Maintains a list of Graph objects in a Report object.
Layers	Maintains a list of Layer objects in a report.
Levels	Maintains a list of Level objects in a report.
Ranges	Maintains a list of Range objects in a report.
ReportQueries	Maintains a list of query objects in a report.
Reports	Maintains a list of Report objects.
Rows	Maintains a list of Row objects in a report.

Related topics

- Chapter 2, "Objects," on page 7
- Chapter 5, "Properties," on page 259
- Chapter 4, "Methods," on page 71

Children

Maintains a list of child objects.

Discussion

Use this collection to access a specific child object in the hierarchy and to determine how many child objects are at a specific level within the hierarchy. For example, you can use this method to obtain the names of the children of the first category in the Columns collection. This collection applies to Columns, Dimensions, Layers, and Rows.

Calculated Categories cannot have children, while alternate categories can.

Name	Description
Item Method	Returns a child object from the Children collection.

Name	Description
Count Property	Returns the number of Child objects in the collection.

Example

These examples obtain the name of the children of the first category in the Columns, Dimensions, Layers, and Rows collections, then return the number of items and names of the children in each of the respective collections.

Column

```
Sub Main()
   Dim objPPRep As Object
   Dim objChildrenCols As Object
   Dim strColChild As String
   Dim strColChildren As String
   Dim intx As Integer
   Set objPPRep = GetObject(,"PowerPlay.Report")
   Set objChildrenCols = objPPRep.Columns.Item(1).Children
   For intx = 1 to objChildrenCols.Count
      strColChild = objChildrenCols.Item(intx).Name
      strColChildren = strColChildren & chr$(10)
& _
         strColChild
   Next intx
   MsgBox "The " & objPPRep.Columns.Item(1).Name
& _
      " category has " & objChildrenCols.Count &
_
      " children." & chr$(10) & chr$(10) &
"They are: " & _
      chr$(10) & strColChildren, , "Column Children"
   Set objChildrenCols = Nothing
   Set objPPRep = Nothing
End Sub
```

Dimension

```
Sub Main()
   Dim objPPRep As Object
   Dim objDimension As Object
   Dim strName As String
   Dim intx as Integer
   Set objPPRep = GetObject(,"CognosPowerPlay.Report")
```

```
   Set objDimension = objPPRep.DimensionLine.Item(1)
   objDimension.ChangeToTop
   If objDimension.HasParent = 0 Then
      Msgbox "The " & objDimension.Name & " dimension
has " & _
         "no parent."
      For intx = 1 to objDimension.Children.Count
          strName = objDimension.Children.Item(intx).Name
          objDimension.Change strName
          MsgBox "The parent for the " &objDimension.Name
& _
              " dimension is " & objDimension.Parent
& "."
      Next intx
   End If
   Set objDimension = Nothing
   Set objPPRep = Nothing
End Sub
```

Layer

```
Sub Main()
   Dim objPPRep As Object
   Dim objChildrenLayers As Object
   Dim strLayerChild As String
   Dim strLayerChildren As String
   Dim intx As Integer
   Set objPPRep = GetObject(,"CognosPowerPlay.Report")
   Set objChildrenLayers = objPPRep.Layers.Item(1).Children
   For intx = 1 to objChildrenLayers.Count
      strLayerChild = objChildrenLayers.item(intx).Name
      strLayerChildren = strLayerChildren & chr$(10)
& _
         strLayerChild
   Next intx
   MsgBox "The " & objPPRep.Layers.Item(1).Name &
_
      " category has " & objChildrenLayers.Count
& _
      " children." & chr$(10) & chr$(10) &
_
      "They are: " & chr$(10) & strLayerChildren,
' _
      "Layer Children"
   Set objChildrenLayers = Nothing
   Set objPPRep = Nothing
End Sub
```

Row

```
Sub Main()
   Dim objPPRep As Object
   Dim objChildrenRows As Object
```

```
   Dim strRowChild As String
   Dim strRowChildren As String
   Dim intx As Integer
   Set objPPRep = GetObject(,"CognosPowerPlay.Report")
   Set objChildrenRows = objPPRep.Rows.Item(1).Children
   For intx = 1 to objChildrenRows.Count
      strRowChild = objChildrenRows.item(intx).Name
      strRowChildren = strRowChildren & chr$(10)
& _
         strRowChild
   Next
   MsgBox "The " & objPPRep.Rows.Item(1).Name &
_
      " category has " & objChildrenRows.Count &
_
      " children." & chr$(10) & chr$(10) &
_
      "They are: " & chr$(10) & strRowChildren,
' _
      "Row Children"
   Set objChildrenRows = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- Chapter 4, "Methods," on page 71
- Chapter 5, "Properties," on page 259

Columns

Maintains a list of Column objects in a report.

Discussion

Collections are generic objects that group or list other types of objects. You can group the Column objects in a Report object into a Columns collection. Using a Columns collection, you can operate on the entire group of Column objects at once rather than one at a time.

You can perform calculations on and sort the Column objects that are in the Columns collection.

To use this collection, first add categories to the report from the CategoryList object, and then add Column objects, or just open an existing report that has columns.

Certain methods are only available in Reporter mode.

Name	Description
Accumulation Method	Accumulates all the values of the categories in an object or a collection.

Name	Description
Active Method	Returns the active Column object.
Add Method (Columns, Layers, Rows)	Adds a Column object to the Columns collection.
Addition Method (Collections)	Adds a constant value or a category to all Column objects in the Columns collection.
AddLevel Method	Adds a column level to the collection of columns in a nested crosstab.
AddLowestLevelCategories Method (Reporter)	Adds the lowest-level categories to a report.
Average Method (Collections) (Reporter)	Returns the average of a constant value or a column, and either one or more columns.
CumPercentOfBase Method	Adds Cumulative Percent of Base Column objects using a Row object as the base category.
DeleteExplorerRank Method	Deletes the rank category from an Explorer report.
Division Method	Divides all the Column objects in the collection by either a constant value or another category.
Exponentiation Method	Raises all the Column objects to the power of either another category or a constant value.
Hide Method	Hides all Column objects in the collection.
Item Method	Returns a Column object from the Columns collection.
ItemAtLevel Method	Returns a Column object from a nested report.
ItemAtLevel Method	Determines the maximum between each Column object in the collection and either a constant value or another category.
Minimum Method (Collections) (Reporter)	Determines the minimum between each Column object in the collection and either a constant value or another category.
Multiplication Method (Collections)	Multiplies a constant value or a category to all the Column objects in the collection.
Percent Method	Adds percent Column objects based on either a category or a constant value.

Name	Description
PercentOfBase Method	Adds Percent of Base Column objects using a Row object as the base category.
Remove Method	Removes all Column objects from the Report object.
Select Method	Selects all Column objects.
Sort Method	Sorts columns in ascending or descending order.
Subset Method	Returns a subset of the Column objects in the Columns collection.
Subtraction Method (Collections)	Subtracts a constant value or a category from all the Column objects in the collection or subtracts all the Column objects from a constant value or a category.
Unselect Method	De-selects all Column objects in the collection.

Name	Description
Application Property	Returns the Application object.
Count Property	Returns the number of Column objects in the collection.
DimensionLineIndex Property	Returns the position of a dimension line item to maintain a list of Layer, Row, and Column objects.
Exception Property	Sets the exception for all the Column objects in the collection.
Style Property	Sets the style used for all the Column objects in the collection.
TopLevelCategory Property (Explorer)	Returns the name of the dimension for the collection.

Example

This example creates a new report in Reporter mode. It multiplies all columns by 10%, and removes the old columns. Finally, it saves and closes the report.

```
Sub Main()
   Dim objPPRep As Object
   Dim objNewCol As Object
   Dim strOldCol As String
```

```
   Dim intx as Integer
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.New "C:\Cubes and Reports\Great Outdoors.mdc",
True
   objPPRep.ExplorerMode = False
   For intx = 1 TO objPPRep.Columns.Count
      strOldCol = objPPRep.Columns.Item(intx).Name
      Set objNewCol = objPPRep.Columns.Item _
          (intx).Multiplication(1.10)
      Msgbox "The new column name is " &objNewCol.Name
      objPPRep.Columns.Item(strOldCol).Remove
   Next intx
   objPPRep.SaveAs "C:\Cubes and Reports\10perc.ppr"
   objPPRep.Close
   Set objNewCol = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "Column Object" on page 15
- "Columns Method" on page 120

Exceptions

Maintains a list of Exception objects in a report.

Discussion

Collections are generic objects that group or list other types of objects. You can group the Exception objects in a Report object into an Exceptions collection. Use an Exceptions collection to operate on an entire group of Exception objects at once rather than one at a time.

You can use the Exceptions collection to return the number of Exception objects in a report.

Name	Description
Add Method (Exceptions)	Adds an Exception object to the Exceptions collection.
Item Method	Returns an Exception object from the Exceptions collection.
Remove Method	Removes all Exception objects from the Report object.

Name	Description
Application Property	Returns the Application object.

Name	Description
Count Property	Returns the number of Exception objects in the Exceptions collection.

Example

This example opens a report and displays the driving fields for the first Exception object.

```
Sub Main()
   Dim objPPRep as Object
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.Open "C:\Cubes and Reports\Exception.ppr"
   MsgBox "Driving Category:" & _
      objPPRep.Exceptions.Item(1).DrivingCategory
   MsgBox "Driving Dimension:" & _
      objPPRep.Exceptions.Item(1).DrivingDimension
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "Exception Object" on page 22

Graphs

Maintains a list of Graph objects in a Report object.

Discussion

Collections are generic objects that group or list other types of objects. You can group the Graphs objects in a Report object into a Graphs collection. Using a Graphs collection, you can operate on the entire group of Graph objects at once rather than one at a time.

Graphs present report data in a number of formats that emphasize different statistical features. For example, pie charts show the relationship of a category to the whole whereas clustered bar graphs show the trend of categories over time.

Name	Description
Active Method	Returns the active Graph object.
Add Method (Graphs)	Adds a Graph object to the Graphs collection.
Item Method	Returns one Graph object from the Graphs collection.
Remove Method	Removes all Graph objects from the Report object.

Name	Description
Application Property	Returns the Application object.
Count Property	Returns the number of Graph objects in the Graphs collection.
DataGridlines Property	Sets whether the gridlines settings are on or off for a crosstab
EnableUserColumnSummaryLabel Property	Sets whether a user-specified label is used for the innermost summary column in a nested crosstab.
EnableUserRowSummaryLabel Property	Returns whether a user-specified label is used for the innermost summary row in a nested crosstab.
HideRankCategory Property	Sets whether the rank category is hidden.
IndentTotalsLevel Property	Sets the current indent level for summary cells on nested crosstabs in the collection.
KeepSummaryVisible Property	Sets whether the summary category will remain visible on all scrolled pages.
LabelGridlines Property	Sets whether the gridlines are on or off for category labels in a nested crosstab.
Layout Property	Sets the current layout style for nested crosstabs in the collection.
MaxPrintedBars Property	Sets the maximum number of bars on a single printed page
MaxVisibleBars Property	Sets the maximum number of bars visible on a single page of scrolled data.
NamesShown Property	Sets or returns whether category names appear beside pie chart slices.
ShowSummaryBreakdown Property (Explorer)	Sets or returns whether to show the breakdown of summary rows and columns in a crosstab.
ShowSummaryColumn Property (Explorer)	Sets whether to show the summary column for crosstabs in the collection.
ShowSummaryRow Property (Explorer)	Sets whether to show the summary row for crosstabs in the collection.
ShowTies Property	Sets whether to show label ties.
StatsLineCaption Property	Sets or returns the caption for a given statistical line on a graph.

Name	Description
StatsLineColor Property	Sets or returns the color for a given statistical line on a graph.
StatsLineOn Property	Sets or returns a statistical line on a graph.
StatsLineStyle Property	Sets or returns the line style of a given statistical line on a graph.
StatsLineUserValue Property	Sets a custom value for a statistical line on a graph.
UseScrolling Property	Sets or returns whether scrolling is enabled.
UserColumnSummaryLabel Property	Sets or returns the user-defined label for the innermost summary column in a nested crosstab.
UserRowSummaryLabel Property	Sets or returns the user-defined label for the innermost summary row in a nested-crosstab.
ValuesAutoFit Property	Sets whether value labels fit within graph bars and pie segments.
ValuesFontColor Property	Sets the font color used for the value labels for graphs in the collection.
ValuesFontName Property	Sets the font name used for the value labels for graphs in the collection.
ValuesFontSize Property	Sets the font size used for the value labels for graphs in the collection.
ValuesPosition Property	Sets the position of value labels on some graph types in the collection.
ValuesShown Property	Sets or returns whether value labels appear next to pie chart slices.

Example

This example opens a report, adds a new Graph object to the Graphs collection, and saves the report.

```
Sub Main()
   Dim objPPRep as Object
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.Open "C:\Cubes and Reports\Trend.ppr"
   objPPRep.Graphs.Add 5
   objPPRep.Save
   objPPRep.Close
   Set objPPRep = Nothing
```

```
End Sub
```

Related Topics

- "Graph Object" on page 25
- "Report Object" on page 37

Layers

Maintains a list of Layer objects in a report.

Discussion

Collections are generic objects that group or list other types of objects. You can group the Layer objects in a Report object into a Layers collection. Use a Layers collection, to operate on the entire group of Layer objects at once rather than one at a time.

To use this collection, first add categories to the report from the CategoryList object, and then add Layer objects, or just open an existing report that has layers.

Certain methods are only available in Reporter mode.

Name	Description
Active Method	Returns the active Layer object.
Add Method (Columns, Layers, Rows)	Adds a Layer object to the Layers collection.
Addition Method (Collections)	Adds a constant value or a category to all Layer objects in the Layers collection.
AddLevel Method	Adds a layer level to the collection of layers in a nested crosstab.
AddLowestLevelCategories Method (Reporter)	Adds the lowest-level categories to a report.
Average Method (Collections) (Reporter)	Returns the average of a constant value or a layer, and either one or more layers.
Division Method	Divides all the Layers in the collection by either a constant value or another category.
Exponentiation Method	Raises all the Layer objects to the power of either a constant value or another category.
Item Method	Returns a Layer object from the Layers collection.
Maximum Method (Collections) (Reporter)	Determines the maximum between each Layer object in the collection and either a constant value or another category.

Name	Description
Minimum Method (Collections) (Reporter)	Determines the minimum between each Layer object in the collection and either a constant value or another category.
Multiplication Method (Collections)	Multiplies a constant value or a category to all the Layer objects in the collection.
Percent Method	Adds percent Layer objects based on either a category or a constant value.
Remove Method	Removes all Layer objects from the Report object.
Select Method	Selects all Layer objects in the collection.
Sort Method	Sorts layers in ascending or descending order.
Subset Method	Returns a subset of the Layer objects from the Layers collection.
Subtraction Method (Collections)	Subtracts a constant value or category from the Layer objects in the collection or subtracts all the Layer objects from a constant value or a category.
Unselect Method	De-selects all Layer objects in the collection.

Name	Description
Application Property	Returns the Application object.
Count Property	Returns the number of Layer objects in the collection.
DimensionLineIndex Property	Returns the position of a dimension line item to maintain a list of layer, row, and column objects.
Exception Property	Sets the exception for all Layer objects in the collection.
Style Property	Sets the style used for all Layer objects in the collection.
TopLevelCategory Property (Explorer)	Returns the name of the dimension for the collection.

Example

This example gets an open report, removes the first and second layer, subtracts the new second layer from the new first layer, and performs maximum calculation on all the layers.

```
Sub Main()
   Dim objPPApp As Object
   Dim objPPRep As Object
   Set objPPApp = GetObject( , "CognosPowerPlay.Application"
)
   Set objPPRep = GetObject("C:\Cubes and Reports\Sample1.ppr")
   objPPRep.Visible = True
   objPPApp.Reports(1).ExplorerMode = False
   objPPApp.Reports(1).Layers.Subset(1,2).Remove
   objPPApp.Reports(1).Layers.Subset(1,2).Subtraction
   objPPApp.Reports(1).Layers.Maximum
   Set objPPRep = Nothing
   Set objPPApp = Nothing
End Sub
```

Related Topics

- "CategoryList Object" on page 13
- "Layer Object" on page 28
- "Report Object" on page 37

Levels

Maintains a list of Level objects in a report.

Discussion

Use a Levels collection to operate on the group of Level objects. For example, you can group the Level objects in a Dimension object into a Levels collection.

Use the Items method to access a Level object in a Levels collection. The index for the item method starts at 1.

There is a Levels collection for the Dimension object.

Name	Description
Item Method	Returns a Level object at a given index in the collection.

Name	Description
Application Property	Returns the Application object.
Count Property	Returns the number of Level objects in the collection.

Example

This example displays all levels for the first dimension of the dimension line.

```
Sub Main()
   Dim objPPRep As Object
   Dim objDimension As Object
   Dim objLevel As Object
   Dim strLevelList As String
   Dim intx As Integer
   Set objPPRep = GetObject(,"CognosPowerPlay.Report")
   Set objDimension = objPPRep.DimensionLine.Item(1)
   For intx = 1 to objDimension.Levels.Count
      Set objLevel = objDimension.Levels.Item(intx)
      strLevelList = strLevelList & chr$(10) &
objLevel.Name
      Set objLevel = Nothing
   Next intx
   MsgBox "The levels in the " & objDimension.Name
& _
         " dimension are:" & chr$(10) & strLevelList
   Set objDimension = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "Level Object" on page 30

Ranges

Maintains a list of Range objects in a report.

Discussion

Collections are generic objects that group or list other types of objects. You can group the Range objects in an Exception object into a Ranges collection. Use a Ranges collection to operate on the entire group of Range objects at once rather than one at a time.

Name	Description
Add Method (Ranges)	Adds a Range object to the Ranges collection.
Item Method	Returns a Range object from the Ranges collection.
Remove Method	Removes all Range objects from the Exception object.

Name	Description
Application Property	Returns the Application object.
Count Property	Returns the number of Range objects in the collection.
MaximumNumberOfRanges Property	Returns the maximum number of Range objects definable for an Exception object.

Example

This example shows how to return a Ranges collection and returns the maximum number of Range objects that you can define for the Exception object.

```
Sub Main()
   Dim objPPRep As Object
   Dim objPPRange As Object
   Set objPPRep = GetObject("C:\Cubes and Reports\Exception.ppr")
   objPPRep.Visible = True
   Set objPPRange = objPPRep.Exceptions.item(1).Ranges
   MsgBox "The maximum number of ranges: " _
      &objPPRange.MaximumNumberOfRanges
   Set objPPRange = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "Exception Object" on page 22
- "Range Object" on page 35

ReportQueries

Maintains a list of query objects in a report.

Discussion

ReportQueries is a collection of query objects. Collections are generic objects that group or list other types of objects. You can group query objects in a Report object into a ReportQueries collection. Using a ReportQueries collection, you can operate on the entire group of ReportQueries objects at once rather than one at a time.

To use this collection, you can add categories to the report from the CategoryList object, and then create query objects; or just open an existing report that has queries.

The ReportQueries collection is a special object that belongs to the Report object. There are several variations of the query objects that belong to the ReportQueries collection; the FindQuery, ParentageQuery, and AdvancedQuery.

Name	Description
Add Method (ReportQueries)	Adds a query object to the ReportQueries collection.
Item Method	Returns a query object from the ReportQueries collection.
Remove Method	Removes all query objects from the ReportQueries collection.

Name	Description
Application Property	Returns the Application object.
Count Property	Returns the number of objects in the collection.

Example

This example creates a FindQuery subset definition that searches for all products that begin with the name "Star".

```
Sub Main()
   Dim strCubePath As String
   Dim objPPRep As Object
   Dim objFind As Object
   strCubePath = "C:\Cubes and Reports\Great Outdoors.mdc"
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.New strCubePath, 1
   objPPRep.ExplorerMode = False
   objPPRep.Visible = True
   Set objFind = objPPRep.ReportQueries.Add(1)
   With objFind
      .Name = "Find Star"
      .Dimension = "Products"
      .SearchShortName = False
      .SearchText = "Star"
      .Pattern = 2
   End With
   Set objFind = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- “AdvancedQuery Object” on page 8
- “FindQuery Object” on page 23
- “ParentageQuery Object” on page 31
- “ValueRestriction Object” on page 46

Reports

Maintains a list of Report objects.

Discussion

Collections are generic objects that group or list other types of objects. You can group the Report objects in an Application object into a Reports collection. Use a Reports collection to operate on an entire group of Report objects at once rather than one at a time.

When you use the Add method or the Open method for the Reports collection, you must capture the new Report object or it will terminate. For example, you must use the following to capture the new report created from the OUTDOORS.MDC using the object, PPRep1:

```
Set PPRep1 = PPApp.Reports.Add("c:\cognos\outdoors.mdc")
```

Using

```
PPApp.Reports.Add("c:\cognos\outdoors.mdc")
```

will terminate the report.

Name	Description
Active Method	Returns the active Report object.
Add Method (Reports)	Adds a Report object to the Reports collection.
Close Method	Closes all Report objects in the collection.
Item Method	Returns a Report object from the Reports collection.
Open Method (Reports)	Opens a collection of Reports objects.

Name	Description
Application Property	Returns the Application object.
Count Property	Returns the number of Report objects in the collection.

Example

This example adds a two reports to the collection and then and makes the first report in the collection visible.

```
Sub Main()
   Dim objPPApp As Object
   Dim objTest As Object
   Set objPPApp = CreateObject("CognosPowerPlay.Application")
   objPPApp.Visible = True
```

```
   Set objTest = objPPApp.Reports.Add _
      ("C:\Cubes and Reports\Great Outdoors.mdc")
   Set objTest = objPPApp.Reports.Open _
      ("C:\Cubes and Reports\Exception.ppr")
   Msgbox "The number of reports in the collection is
" _
      &objPPApp.Reports.Count
   Msgbox "The name of the first report in the collection
is " _
      &objPPApp.Reports(1).Name
   objPPApp.Reports(1).Visible = True
   Set objTest = Nothing
   Set objPPApp = Nothing
End Sub
```

Related Topics

- "Report Object" on page 37

Rows

Maintains a list of Row objects in a report.

Discussion

Collections are generic objects that group or list other types of objects. You can group Row objects in a Report object into a Rows collection. Use a Rows collection to operate on an entire group of Row objects at once rather than one at a time.

To use this collection, first add categories to the report from the CategoryList object, and then add Row objects, or just open an existing report that has rows.

Certain methods are only available in Reporter mode.

Name	Description
Accumulation Method	Accumulates all the values of the categories in the Rows collection.
Active Method	Returns the active Row object.
Add Method (Columns, Layers, Rows)	Adds a Row object to the Rows collection.
Addition Method (Collections)	Adds a constant value or a category to all Row objects in the Rows collection.
AddLevel Method	Adds a row level to the collection of rows in a nested crosstab.
AddLowestLevelCategories Method (Reporter)	Adds the lowest-level categories to a report.
Average Method (Collections) (Reporter)	Returns the average of a constant value or a row, and either one or more rows.

Name	Description
CumPercentOfBase Method	Adds Cumulative Percent of Base Row objects using a Column object as the base category.
DeleteExplorerRank Method	Deletes the rank category from an Explorer report.
Division Method	Divides all the Row objects in the collection by either a constant value or another category.
Exponentiation Method	Raises all the Rows to the power of either another category or a constant value.
Hide Method	Hides all Row objects in the collection.
Item Method	Returns a Row object from the Reports collection.
ItemAtLevel Method	Returns a Row object from a nested report.
Maximum Method (Collections) (Reporter)	Determines the maximum between each Row object in the collection and either a constant value or another category.
Minimum Method (Collections) (Reporter)	Determines the minimum between each Row object in the collection and either a constant value or another category.
Multiplication Method (Collections)	Multiplies a constant value or a category to all the Row objects in the collection.
Percent Method	Adds percent Row objects based on either a category or a constant value.
PercentOfBase Method	Adds Percent of Base Row objects using a Column object as the base category.
Remove Method	Removes all Row objects from the Report object.
Select Method	Selects all Row objects in the collection.
Sort Method	Sorts rows in ascending or descending order.
Subset Method	Returns a subset of the Row objects in the Rows collection.
Subtraction Method (Collections)	Subtracts a constant value or a category from the Row objects in the collection, or subtracts all the Row objects from a constant value or a category.

Name	Description
Unselect Method	De-selects all Row objects in the collection.

Name	Description
Application Property	Returns the Application object.
Count Property	Returns the number of Row objects in the collection.
DimensionLineIndex Property	Returns the position of an dimension line item to maintain a list of layer, row, and column objects.
Exception Property	Sets the exception for all Row objects in the collection.
Style Property	Sets the style used for all Row objects in the collection.
TopLevelCategory Property (Explorer)	Returns the name of the dimension for the collection.

Example

This example adds the first three rows, and also adds column one and column four.

```
Sub Main()
   Dim objPPRep As Object
   Dim objNewCol As Object
   Dim objNewRows As Object
   Set objPPRep = GetObject( , "CognosPowerPlay.Report")
   objPPRep.ExplorerMode = False
   Set objNewRows = objPPRep.Rows.Subset(1,3).Addition
   MsgBox "The sum of the first three rows is " _
      &objPPRep.CellValue(objNewRows(1).Index,1)
   Set objNewCol = objPPRep.Columns.Item(4).Addition
_
      (objPPRep.Columns.Item(1))
   MsgBox "The sum of columns 1 and 4 is " _
      &objPPRep.CellValue(1,objNewCol.Index)
   Set objNewRows = Nothing
   Set objNewCol = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "Report Object" on page 37
- "Row Object" on page 41

Chapter 4. Methods

You work with the following methods for IBM Cognos PowerPlay OLE automation.

Name	Description
Accumulation Method	Accumulates all the values of the categories in an object or a collection.
Activate Method	Sets the focus on an object.
Active Method	Returns the active object in the collection.
ActiveReport Method	Returns the active Report object for the Application object.
Add Method (CategoryList)	Adds one or more categories to a CategoryList object.
Add Method (Columns, Layers, Rows)	Adds one or more objects a collection, or to an object that maintains a list of other objects.
Add Method (Exceptions)	Adds an Exception object to the Exceptions collection.
Add Method (Graphs)	Adds a Graph object to the Graphs collection.
Add Method (Ranges)	Adds a Range object to the Ranges collection.
Add Method (ReportQueries)	Adds a query object to the ReportQueries collection.
Add Method (Reports)	Adds a Report object to the Reports collection.
AddBlanks Method (Reporter)	Adds a single blank row or column to a nested crosstab.
Addition Method (Collections)	Adds a constant value or a category to one or more objects in the collection.
Addition Method (Objects)	Adds a constant value or a category to an object.
AddLevel Method	Adds a row, column, or layer level to a nested crosstab.

Name	Description
AddLowestLevelCategories Method (Reporter))	Adds the lowest-level categories to a report.
AddToReport Method	Adds query results to a report.
AddToReportAtSpecificNestingLevel Method	Adds query results to a report at a specific nesting level.
Add Method (CategoryList)	Returns the average of a constant value or a category, and either one or more categories.
Average Method (Objects) (Reporter)	Determines the average between a constant value or another category and an object.
CanDrillDown Method	Returns whether you can drill down the category object.
CanDrillUp Method	Returns whether you can drill up the category object.
Category Method	Sets the parent category or dimension for the ParentageQuery subset definition.
CategoryList Method	Creates a CategoryList object which is used to identify categories to be inserted in the Report object.
CellValue Method	Gets the value of a cell in a Report object.
Change Method	Changes the current category for the Dimension.
ChangeToParent Method	Changes the current category for the Dimension object to the category one level higher in the hierarchy.
ChangeToTop Method	Changes the current category for the Dimension object to top-level category.
Children Method	Returns the next child in the hierarchy for an object.
Close Method	Closes one or all the Report objects.
Columns Method	Returns a collection that contains all the Column objects.
Copy Method	Copies the Row, Column, or Layer objects currently selected in the report to the Clipboard.

Name	Description
CumPercentOfBase Method	Adds one or more Cumulative Percent of Base category using a category from a different dimension as the base.
Cut Method (Reporter)	Moves the Row, Column, or Layer objects currently selected to the Clipboard.
DeleteAllMDCAccessInfo Method	Deletes PowerCube security access information from memory for all cubes.
DeleteExplorerRank Method	Deletes the rank category from an Explorer report.
DeleteDataSourceInfo Method	Deletes PowerCube security access information from memory for the specified cube.
DeleteAllDataSourceInfo Method	Deletes security access information from memory for all PowerCubes.
DeleteMDCAccessInfo Method	Deletes security access information for a local PowerCube from memory.
DeleteSelected Method	Deletes selected objects from a Report collection.
DeploymentOptions Method	Returns the distribution options for a report published to the IBM Cognos portal.
Depth Method	Returns whether the Graph object is three-dimensional (3D).
DimensionFilter Method	Sets the filter category for an indexed dimension.
DimensionLine Method	Returns a DimensionLine object for the current Report object.
Division Method	Divides one or more categories by either a constant value or another category.
DrillDown Method	Drills down the category object.
DrillUp Method	Drills up the category object.
Exceptions Method	Returns one Exception object or the entire collection.
Exclude Method	Sets the categories to exclude from the query.
Execute Method	Runs a query on the cube.

Name	Description
Exponentiation Method	Raises one or more categories to the power of either another category or a constant value.
Find Method	Specifies the name of the FindQuery object to include in an AdvancedQuery.
FindNext Method	Finds the next matching category label in a report.
FindPrevious Method	Finds the previous matching category label in a report.
Forecast Method (Explorer)	Creates a specified number of forecast categories, based on the existing time dimensions.
GetDataNow Method	Updates the data in the Report object.
Graphs Method	Returns one Graph object or the entire collection.
HasParent Method	Returns whether the current category has a parent.
Hide Method	Hides the category.
HideSelected Method	Hides selected objects in a collection.
HideUnselected Method	Hides any object in a collection that is not selected.
Include Method	Sets the categories to include in the query.
Item Method	Returns an object from the collection or object that maintains a list of other objects.
ItemAtLevel Method	Returns either a row or column object from a nested report.
Layers Method	Returns one Layer object or the entire collection.
Level Method	Sets the level used by the AdvancedQuery object to retrieve categories for the query.
Levels Method	Returns all levels available in the dimension for a category.
Logon Method	Logs on to IBM Cognos Analytics as an authenticated user to provide the application object access to secured IBM Cognos Analytics resources such as packages.

Name	Description
Logoff Method	Revokes authentication to all IBM Cognos Analytics namespaces for the application object. Even if you used multiple namespaces in the session, you logoff only once.
Maximize Method	Maximizes the object window.
Maximum Method (Collections) (Reporter)	Determines the maximum between either a constant value or a category, and one or more categories.
Maximum Method (Objects) (Reporter)	Determines the maximum between either a constant value or a category, and one or more categories.
Minimize Method	Minimizes the object window.
Minimum Method (Collections) (Reporter)	Determines the minimum between either a constant value or a category, and one or more categories.
Minimum Method (Objects) (Reporter)	Determines the minimum between either a constant value or a category, and one or more categories.
Multiplication Method (Collections)	Multiplies a constant value or a category by one or more categories.
Multiplication Method (Objects)	Multiplies a constant value or a category by an object.
New Method	Creates a new Report object.
Open Method (Report)	Opens an existing Report object.
Open Method (Reports)	Opens a collection of Report objects.
“OpenRemoteReport Method” on page 189	Opens an existing remote report object.
Parent Method	Returns the name of the parent category.
Paste Method (Reporter)	Pastes the contents of the Clipboard into the selected categories of the Report object.
PDFFile Method	Sets the file name of the PDF object when it is saved.
Percent Method	Adds one or more percent categories based on either a category or a constant value.
PercentGrowth Method	Calculates the percentage change between two categories or measures.

Name	Description
PercentOfBase Method	Adds one or more percent of base categories using a category from a different dimension as the base.
Print Method	Returns a Print object.
PrintOut Method	Prints the Report object.
PublishToPortal Method	Creates a report in the IBM Cognos Analytics content store. Users access the report using the Cognos Analytics portal.
Quit Method	Exits PowerPlay.
Ranges Method	Returns one Range object or the entire collection.
Rank2 Method	Ranks and sorts Row or Column objects.
Remove Method	Removes a single object or all objects from a collection.
Remove Method (ReportQueries)	Removes all query objects from the ReportQueries collection.
RemoveLevel Method	Removes a column, layer, or row level from a nested crosstab in a report.
ReportQueries Method	Returns a ReportQueries collection.
Reports Method	Returns one Report object or the entire collection.
ResetPrintOptionsToDefault Method	Resets print options back to the default settings.
Restore Method	Restores the object window to its original size and position.
Rollup Method	Groups categories containing calculated values to create a new, dynamic calculation.
Rows Method	Returns one Row object or the entire collection.
Save Method	Saves one or all Report objects.
SaveAs Method	Saves the Report object with a different name, and if desired, a different format.
Select Method	Selects categories.

Name	Description
SelectAllDimensions Method	Selects all the dimension objects in the dimension line that can be filtered when a report is opened in the IBM Cognos portal.
SelectBlank Method	Selects a specific blank row or column.
SetChartToPrint Method	Specifies which Graph object of a report to print.
SetChartToSave Method	Specifies which Graph object to save in a PDF.
SetDataSourceInfo Method	Stores security information for a data source in memory.
SetDrivingCategory Method	Sets the driving category for the Exception object.
SetListOfLayersToPrint Method	Specifies the range of layers of the report to print.
SetListOfLayersToSave Method	Specifies the range of layers to save in a PDF.
SetListOfRowsToPrint Method	Specifies the range of rows of the report to print.
SetListOfRowsToSave Method	Specifies the range of rows to save in a PDF.
SetMacro Method	Sets the name and style for the macro used by the Exception object.
SetMDCAccessInfo Method	Stores PowerCube security access information in memory.
SetType Method	Sets the Graph object type.
SizeSelected Method	Applies a size to selected objects.
Sort Method	Sorts columns, layers, or rows in ascending or descending order.
StyleSelected Method	Applies a style to the selected object.
Subset Method	Returns a subset of objects from the current collection.
Subtraction Method (Collections)	Subtracts a constant value or a category from one or more categories in the collection, or subtracts all categories from a constant value or another category.

Name	Description
Subtraction Method (Objects)	Subtracts a constant value or another category from an object, or subtracts an object from the category or constant value.
SwapColumnsAndLayers Method	Exchanges the positions of the Column objects and Layer objects.
SwapRowsAndColumns Method	Exchanges the positions of the Row objects and Column objects.
SwapRowsAndLayers Method	Exchanges the positions of the Row objects and Layer objects.
UnhideAllCategories Method	Makes all hidden categories visible.
Unselect Method	De-selects categories.
UnselectBlank Method	Unselects a specific blank row or column.
UnselectAllDimensions Method	Clears all selected dimension objects in the dimension line that can be filtered when a report is opened in the IBM Cognos portal.
UpdatePublishedReport Method	Updates a report previously published to the IBM Cognos Analytics content store.
ValueRestriction Method	Returns the value restriction for an AdvancedQuery object.
Vertical Method	Returns whether the Graph object is a vertical display.

Accumulation Method

Accumulates all the values of the categories in an object or a collection.

Syntax

object.**Accumulation** [Operand]

Applies To

Column Object

Columns

Row Object

Rows

Discussion

You cannot accumulate values over layers.

Depending on whether the method was applied to an object or a collection, the results are returned respectively as either an object or a collection.

The result of each calculation is stored in a new category which is added beside the Row or Column object, or to the end of the Rows or Columns collection.

When you accumulate the values in a collection, a new category is created for each object in the collection. For example, five columns are created when the Accumulation method is used in a collection of five columns.

References to the position of an object in the collection are not valid after you use this method.

Crossing accumulated categories produces an undefined result.

Notes

When you accumulate values in Explorer mode, the Summary Categories are exempt and therefore show N/A values. N/A values are also shown at the intersection point of a rank, other accumulation, or a business calculation such as Cumulative Percent of Base.

In Explorer mode, if more than one category is selected, the Accumulate calculation is disabled.

Parameters	**Description**
Operand	Optional. Specifies either a constant value or a category object. Type: Variant

Return Type

Object

Example

This example returns an open report, accumulates all columns and all rows, and returns new rows and new columns.

```
Sub Main()
   Dim objPPRep As Object
   Set objPPRep = GetObject("C:\Cubes and Reports\Sample2.ppx")
   objPPRep.ExplorerMode = False
   objPPRep.Columns.Accumulation
   objPPRep.Rows.Accumulation
   Set objPPRep = Nothing
End Sub
```

Related Topics

- Chapter 4, "Methods," on page 71
- Chapter 5, "Properties," on page 259
- "Column Object" on page 15
- "Columns" on page 52
- "Row Object" on page 41
- "Rows" on page 66

Activate Method

Sets the focus on an object.

Syntax

object.**Activate**

Applies To

Application Object

Column Object

Graph Object

Layer Object

Report Object

Row Object

Discussion

Use this method to

- activate the Graph object
- activate the Layer object, and moves the cursor to that category
- activate the Column or Row object and moves the cursor to that category
- activate a Report object, and brings it to the front of all the Report objects in the same Application object
- activate the application, and brings it in front of all the other applications if the Visible method is True

For example, when you create an automation procedure that involves several different reports, you can set the focus on the different objects involved. For example, if the macro opens four different reports, use the Activate method to shift the focus to one that is not currently active.

Return Type

Nothing

Example

This example establishes a link to an already opened report in order to modify its content. Using the Application method, the IBM Cognos PowerPlay application is brought to the front and made active.

```
Sub Main()
   Dim objRep As Object
   Set objRep = GetObject( ,"CognosPowerPlay.Report")
   MsgBox "The name of the current report is " &objRep.Name
   If objRep.Saved = False Then
      objRep.Save
      MsgBox "Changes to the report have been saved."
   Else
      MsgBox "No changes have been made to the report."
   End If
   objRep.Application.Activate
   Set objRep = Nothing
End Sub
```

Active Method

Returns the active object in the collection.

Syntax

object.**Active**

Applies To

Columns

Graphs

Layers

Reports

Rows

Discussion

Use this method to gain control of the active object in a collection. For example, when you have multiple Graph objects in a report, use this method to identify the single active graph object in the collection of Graph objects.

Related Topics

- "Columns" on page 52
- "Graphs" on page 56
- "Layers" on page 59
- "Reports" on page 65
- "Rows" on page 66

Return Type

Object

Example

This example adds a stacked bar graph to the report and shows the graph type of the active graph.

```
Sub Main()
   Dim objPPRep as Object
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.Open "C:\Cubes and Reports\Trend.ppx"
   objPPRep.Graphs.Add 5
   MsgBox "The active graph type is " _
      &objPPRep.Graphs.Active.Type
   objPPRep.Save
   objPPRep.Close
   Set objPPRep = Nothing
End Sub
```

ActiveReport Method

Returns the active Report object for the Application object.

Syntax

Application.**ActiveReport**

Applies To

Application Object

Discussion

Use this method to gain control of the active report in the application. For example, when you open multiple reports, use this method to return the active Report object, without the need to specify the name of the Report object.

Return Type

Object

Example

This example adds a report to the Reports collection so that there are multiple reports open. The ActiveReport method sets the focus on the last report that you opened or added to the collection.

```
Sub Main()
   Dim objPPRep As Object
   Dim objPPApp As Object
   Dim objReports As Object
   Set objPPApp = GetObject(, "CognosPowerPlay.Application")
   Set objReports = objPPApp.Reports
```

```
   Set objPPRep = objReports.Add("C:\Cubes and Reports\"
& _
      "Sample1.ppx")
   objPPRep.Visible = True
   MsgBox "There are " & objReports.Count & "
reports open."
   Set objPPRep = objPPApp.ActiveReport
   MsgBox "The name of the active report is " &objPPRep.Name
   Set objPPRep = Nothing
   Set objReports = Nothing
   Set objPPApp = Nothing
End Sub
```

Related Topics

- "Application Object" on page 11

Add Method (CategoryList)

Adds one or more categories to a CategoryList object.

Syntax

CategoryList.**Add** Level, Dimension[, Category]

Applies To

CategoryList Object

Discussion

To add categories to a report, first create a CategoryList object by calling the CategoryList method, which is a Report method, and then use it to identify the required categories. Then use this method to select existing categories from the multidimensional cube file (.mdc).

For CategoryList objects, the Add method is used to select categories out of all the possible categories in order to add them to the CategoryList object. The added categories are the ones found on Level n of the specified Dimension or Category.

In ReporterMode, the Level parameter can be any number. In ExplorerMode, if the Level parameter isn't zero, the value will be ignored. In ExplorerMode, the "Add" method is a drill down or filter action where the macro replaces categories, whereas in ReporterMode, it is an additive action where the macro adds categories without replacing existing ones.

If no Category names are specified, then all categories at the specified level below the dimension are added to the CategoryList object.

If one Category name is specified, then all categories at the specified level below the Category name are added to the CategoryList object.

A subsequent Category name must be child of the previous Category name. Up to 10 Category names can be specified.

For example, in

- Catlist.Add 0, "Years"

the Years dimension category (and none of its children) are added to the report.

- Catlist.Add 2,"Dimension","Category1","Category2"

categories in the second level below Category2 are added to the CategoryList object. Category1 specifies a category from the first level of Dimension, and Category2 identifies a child of Category1.

A Dimension is required before categories can be added to it.

References to the position of an object in the collection are not valid after you use this method.

Notes

If category names are unique within the specified dimension, then lower-level category names can be specified without having to specify their parent category. For example, in

```
Catlist.Add 2, "Dimension","Category1"
```

categories in the second level below Category1 are added to the CategoryList object and Category1 could be a category of the third level of dimension Dimension. Although syntactically correct, this approach does not make it obvious to someone reading the macro exactly which level the categories added are from.

Parameters	**Description**
Level	Required. Specifies the number below a specified dimension or category, in which to find the categories to be added. Zero means the specified dimension or category. Type: Integer See the Notes in Discussion regarding this parameter.
Dimension	Required. Specifies the dimension name or index to add, or in which to find the categories to be added. Type: String
Category	Optional. Zero or more category names can be specified in order to identify the exact category desired or to identify the starting category from which to add children. The maximum is 10. Type: String

Return Type

Object

Example

This example adds rows to a Reporter report. The rows are categories below the 2008 Q1 category of the Years dimension.

```
Sub Main()
   Dim objMDCCats as Object
   Dim objPPRep as Object
   Set objPPRep = GetObject(,"CognosPowerPlay.Report")
   Set objMDCCats = objPPRep.CategoryList
   objMDCCats.Add 1, "Years","2008","2008 Q1"
   objPPRep.Rows.Add objMDCCats
   objPPRep.Save
   Set objPPRep = Nothing
   Set objmdcCats = Nothing
End Sub
```

Related Topics

- Chapter 4, "Methods," on page 71
- Chapter 5, "Properties," on page 259

Add Method (Columns, Layers, Rows)

Adds one or more objects a collection, or to an object that maintains a list of other objects.

Syntax

collection.**Add** CategoryList [, NestingLevel]

Applies To

Columns

Layers

Rows

Discussion

Use this method to add the categories identified in a CategoryList object to the columns, layers, or rows of the report.

References to the position of an object in the collection are not valid after you use this method.

Parameters	Description
CategoryList	Required. Specifies the CategoryList object without having to identify the dimension if the category label is unique. Type: Object

Parameters	Description
NestingLevel	Optional: A value indicating the nesting level to add the categories to. Default: 0 (top level) Type: Variant

Return Type

Object

Example

This example adds a new level of categories as layers to a report.

```
Sub Main()
   Dim objCubeCategories As Object
   Dim objPPRep As Object
   Const level_0 = 0
   Const level_1 = 1
   Const add_to_current = 0
   Const add_to_all = 1
   Const as_parent = 0
   Const as_child = 1
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.New "C:\Cubes and Reports\Great Outdoors.mdc",
-1
   objPPRep.ExplorerMode = False
   objPPRep.Visible = True
   Set objCubeCategories = objPPRep.CategoryList
   objCubeCategories.Add level_1, "Locations"
   objPPRep.Layers.Add objCubeCategories
   Set objCubeCategories = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- Chapter 4, “Methods,” on page 71
- Chapter 5, “Properties,” on page 259

Add Method (Exceptions)

Adds an Exception object to the Exceptions collection.

Syntax

Exceptions.**Add** Identifier

Applies To

Exceptions

Discussion

The Add method adds an Exception object to an Exceptions collection. The Identifier is the name to be given to the added object.

When you create an exception using the Add method, it is not a shared exception, and will not be added to the ppexcept.ini file.

References to the position of an object in the collection are not valid after you use this method.

Parameters	Description
Identifier	Required. Specifies the exception name. Type: String

Return Type

Object

Example

This example creates an exception with one range and then applies it to a new report. The style that is applied to the exception must be predefined.

```
Sub Main()
   Dim objPPRep As Object
   Dim strExceptionName As String
   strExceptionName = "At Least 400 Thousand"
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.New "C:\Cubes and Reports\Great Outdoors.mdc",True
   objPPRep.ExplorerMode = False
   objPPRep.Visible = True
   objPPRep.Exceptions.Add strExceptionName
   objPPRep.Exceptions.Item(strExceptionName).Ranges.Add
_
       "Minimum",399999,"Good News"
   objPPRep.Columns.Exception =strExceptionName
   Set objPPRep = Nothing
End Sub
```

Related Topics

- Chapter 4, “Methods,” on page 71
- Chapter 5, “Properties,” on page 259

Add Method (Graphs)

Adds a Graph object to the Graphs collection.

Syntax

Graphs.**Add** GraphType[, Depth][, Vertical]

Applies To

Graphs

Discussion

A Graph object is the same as a display in the user interface.

Use the following list to set the Graph object type:

- 0 (Crosstab)
- 1 (Pie)
- 2 (3-D)
- 3 (Bar)
- 4 (Cluster)
- 5 (Stack)
- 6 (Line)
- 7 (Multi-Line)
- 8 (Correlated)
- 9 (Scatter)

References to the position of an object in the collection are not valid after you use this method.

Parameters	**Description**
GraphType	Required. Specifies the Graph object type to add. Type: Integer
Depth	Optional. Specifies whether the Graph object is three-dimensional (3D). Applies only to graph types 1, 3, 4, and 5. Default: True (for Type 1, 3, 4 and 5) Type: Boolean
Vertical	Optional. Specifies whether the Graph object is a vertical display. If the property is False, it applies only to graph type 3. Default: True Type: Boolean

Return Type

Object

Example

This example adds a new display type to the active report. The display added is a horizontal, three-dimensional, bar graph.

```
Sub Main()
   Dim objPPRep As Object
   Dim objPPGraph As Object
   Dim objGraphs As Object
   Const type_bar = 3
   Const three_d = True
   Const horizontal = False
   Set objPPRep = GetObject(, "CognosPowerPlay.Report")
   Set objGraphs = objPPRep.Graphs
   Set objPPGraph = objGraphs.Add _
      (type_bar, three_d, horizontal)
   Set objPPGraph = Nothing
   Set objGraphs = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- Chapter 4, "Methods," on page 71
- Chapter 5, "Properties," on page 259

Add Method (Ranges)

Adds a Range object to the Ranges collection.

Syntax

Ranges.**Add** LowerBoundary, UpperBoundary, Style

Applies To

Ranges

Discussion

You must define a range to which an exception is applied and where a specified style is also applied to the exception. Use the UpperBoundary and LowerBoundary methods of the Range object to determine the range to apply formatting when the information in the report meets the conditions set by the exception range.

You can define multiple value ranges for one exception, and attach a formatting style to each range.

References to the position of an object in the collection are not valid after you use this method.

Parameters	Description
LowerBoundary	Required. Specifies the lower boundary of the range. Type: Variant
UpperBoundary	Required. Specifies the upper boundary of the range. Type: Variant
Style	Required. Specifies the name of the style to be used for the range. Type: Variant

Return Type

Object

Example

This example creates an exception with one range and then applies it to a new report. The style that is applied to the exceptions must be predefined.

```
Sub Main()
   Dim objPPRep As Object
   Dim strExceptionName As String
   strExceptionName = "At Least 400 Thousand"
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.New "C:\Cubes and Reports\Great Outdoors.mdc",True
   objPPRep.ExplorerMode = False
   objPPRep.Visible = True
   objPPRep.Exceptions.Add strExceptionName
   objPPRep.Exceptions.Item(strExceptionName).Ranges.Add
_
       "Minimum",399999,"Good News"
   objPPRep.Columns.Exception =strExceptionName
   Set objPPRep = Nothing
End Sub
```

Related Topics

- Chapter 4, “Methods,” on page 71
- Chapter 5, “Properties,” on page 259

Add Method (ReportQueries)

Adds a query object to the ReportQueries collection.

Syntax

ReportQueries.**Add** Type

Applies To

ReportQueries

Discussion

Use this method to create and add a new query, and return the new query object. You can group query objects from a Report object into a ReportQueries collection. You can use this method to add categories to the report from the CategoryList object, and then create query objects.

The ReportQueries collection is a special object that belongs to the Report object. There are several variations of the query objects that belong to the ReportQueries collection; the FindQuery, ParentageQuery, and AdvancedQuery.

Parameter	Description
Type	Required. Specifies a numerical value that identifies the type of query to add to the query object. 1 = Find Query 2 = Parentage Query 3 = Advanced Query 4 = ValueRestriction Query Type: Integer

Return Type

Object

Example

This example creates an AdvancedQuery (type 3) subset definition that retrieves all categories belonging to Europe. This subset is then added to the report as rows.

```
Sub Main()
   Dim strCubePath As String
   Dim objPPRep As Object
   Dim objAdvanced As Object
   strCubePath = "C:\Cubes and Reports\Great Outdoors.mdc"
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.New strCubePath, 1
   objPPRep.ExplorerMode False
   objPPRep.Visible = True
   Set objAdvanced = objPPRep.ReportQueries.Add(3)
   With objAdvanced
      .Name "European Countries"
      .Dimension "Locations"
      .Level "Country"
      .Include "Europe"
      .Execute
      .AddToReport 1,1,3
   End With
```

```
   Msgbox "Name:  " & objAdvanced.Name & chr$(10)
& _
      "Dimension:  " & objAdvanced.Dimension &
chr$(10) & _
      "Level List:  " & objAdvanced.LevelList &
chr$(10) & _
      "Query Type Code:  " & objAdvanced.Type &
chr$(10) & _
      "Number of Categories:  " & objAdvanced.Count
& _
      chr$(10) & _
      "First Category:  " & objAdvanced.Item(1).Name,
_
      ,"Subset"
   Set objAdvanced = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "AddToReport Method" on page 101
- "AdvancedQuery Object" on page 8
- "FindQuery Object" on page 23
- "ParentageQuery Object" on page 31

Add Method (Reports)

Adds a Report object to the Reports collection.

Syntax

Reports.**Add** MDCName

Applies To

Reports

Discussion

The Add method opens an existing report or cube and adds it to the Reports collection.

When you use the Add method for the Reports collection, you must capture the new Report object or it will terminate. For example, you must use the following to capture the new report created from the outdoors.mdc using the object, objPPRep1:

```
Set objPPRep1 = objPPApp.Reports.Add("c:\cognos\outdoors.mdc")
```

Using

```
objPPApp.Reports.Add(c:\cognos\outdoors.mdc")
```

terminates the report.

Parameters	Description
MDCName	Required. Specifies the name of the MDC (cube) to be opened. Type: String

Return Type

Object

Example

This example adds an existing report to the Reports collection and makes it visible in the active instance of IBM Cognos PowerPlay.

```
Sub Main()
   Dim objPPRep As Object
   Dim objPPApp As Object
   Dim objReports As Object
   Set objPPApp = GetObject(, "CognosPowerPlay.Application")
   Set objReports = objPPApp.Reports
   Set objPPRep = objReports.Add("C:\Cubes and Reports\"
& _
      "Sample1.ppx")
   objPPRep.Visible = True
   MsgBox "There are " & objReports.Count & "
reports open."
   Set objPPRep = Nothing
   Set objReports = Nothing
   Set objPPApp = Nothing
End Sub
```

Related Topics

- Chapter 4, “Methods,” on page 71
- Chapter 5, “Properties,” on page 259

AddBlanks Method (Reporter)

Adds a single blank row or column to a nested crosstab.

Syntax

Report.**AddBlanks**

Applies To

Report Object

Discussion

Use this method to improve the appearance of a crosstab by separating groups of rows or columns with blank spaces. Blank rows and columns can be resized, moved, deleted, or formatted.

Blank rows and columns are not intended to contain values. To add columns, layers, and rows that will contain values, use the Add or AddLevel methods.

AddBlanks is used in conjunction with the Select method. After objects are selected in a report using Select, you can use AddBlanks to add a blank (row or column) immediately after each selected member. The hierarchy of the nested crosstab is respected; that is, an inserted blank will appear at the same level as the corresponding selected member. Blank rows or columns belong to a list that in turn, belong to the selected member.

Using Rows as an example, all selected rows and selected blank rows will have new blank rows inserted below them.

This example uses the ItemAtLevel property to avoid blank rows at a specific position:

```
Sub Main()
   Dim objPPRep as Object
   Set objPPRep = GetObject(,"CognosPowerPlay.Report")
   objPPRep.Rows.ItemAtLevel("Environmental Line",1).Select
   objPPRep.Rows.ItemAtLevel("Recycled Products",0).Select
   objPPRep.Rows.ItemAtLevel("Outdoor Products",1).Select
   objPPRep.Rows.AddBlanks
   objPPRep.Save
   Set objPPRep = Nothing
End Sub
```

Blank rows are not matched with blank layers when rows and layers are swapped; however, when the rows and layers are swapped again, the blank rows will reappear.

AddBlanks applies only to nested crosstabs in Reporter. If the display is not a nested crosstab, the method generates an error.

AddBlanks returns True if successful and False if unsuccessful.

Return Type

Boolean

Example

This example uses the AddBlanks method to add a blank row before the last row and a blank column before the last column in the active report.

```
Sub Main()
   Dim objPPRep As Object
   Dim intRow As Integer
   Dim intColumn As Integer
```

```
   Set objPPRep = GetObject(,"CognosPowerPlay.Report")
   objPPRep.ExplorerMode = False
   intRow = objPPRep.Rows.Count - 1
   intColumn = objPPRep.Columns.Count - 1
   objPPRep.Rows.Item(intRow).Select
   objPPRep.AddBlanks
   objPPRep.Rows.Unselect
   objPPRep.Columns.Item(intColumn).Select
   objPPRep.AddBlanks
   objPPRep.Columns.Unselect
   objPPRep.Rows.ItemAtLevel(intRow,0).SelectBlank(1)
   objPPRep.Columns.ItemAtLevel(intColumn,0).SelectBlank(1)
   Msgbox " A blank row and column have been added "
& _
      "and selected.",64,"Blanks"
   objPPRep.Rows.ItemAtLevel(intRow,0).UnselectBlank(1)
   objPPRep.Columns.ItemAtLevel(intColumn, _
      0).UnselectBlank(1)
   Msgbox " The blank row and column have now been "
& _
      "unselected.",64,"Blanks"
   objPPRep.Save
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "Select Method" on page 220

Addition Method (Collections)

Adds a constant value or a category to one or more objects in the collection.

Syntax

collection.**Addition** [Operand]

Applies To

Columns

Layers

Rows

Discussion

This method can also add multiple categories together.

Depending on whether the method was applied to an object or a collection, the results are returned respectively as an object or a collection.

References to the position of an object in the collection are not valid after you use this method.

Parameters	Description
Operand	Optional. Specifies either a constant value or a category object. If you specify this parameter, the method calculates the addition for each category and operand pair and creates a new category for each result. Type: Variant

Return Type

Object

Example

This example adds the first three rows and also adds column one and column four.

```
Sub Main()
   Dim objPPRep As Object
   Dim objNewCol As Object
   Dim objNewRow As Object
   Set objPPRep = GetObject( , "CognosPowerPlay.Report")
   Set objNewRow = objPPRep.Rows.Subset(1, 3).Addition
   MsgBox "The sum of the first three rows is " _
      &objPPRep.CellValue(objNewRow(1).Index,1)
   Set objNewCol = objPPRep.Columns.Item(4).Addition
_
      (objPPRep.Columns.Item(1))
   MsgBox " The sum of column one and column four is
" _
      &objPPRep.CellValue(1,objNewCol.Index)
   objPPRep.Save
   Set objNewRow = Nothing
   Set objNewCol = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- “Column Object” on page 15
- “Columns” on page 52
- “Layer Object” on page 28
- “Layers” on page 59
- “Row Object” on page 41
- “Rows” on page 66

Addition Method (Objects)

Adds a constant value or a category to an object.

Syntax

object.**Addition**(Operand)

Applies To

Column Object

Layer Object

Row Object

Discussion

Use this method to calculate the addition for the category and the operand, and creates a new category for each result.

In Explorer mode, the operands of the calculation must be from the same dimension and from the same axis.

Depending on whether the method was applied to an object or a collection, the results are returned respectively as an object or a collection.

In Explorer mode, the new calculation is inserted directly after the last operand. In Reporter mode, the new calculation is inserted directly after the active row or column.

In Explorer mode, if you change a report by removing a level, drilling, filtering or nesting, then all calculations that can not be created in the changed report disappear.

References to the position of an object in the collection are not valid after you use this method.

Parameters	**Description**
Operand	Required. Specifies either a constant value or a category object. Type: Variant

Return Type

Object

Example

This example adds 6 to a column and then returns the sum in a new column in the report.

```
Sub Main()
   Dim objPPRep as Object
   Dim objPPCol as Object
   Set objPPRep = CreateObject ("CognosPowerPlay.Report")
   objPPRep.Open "C:\Cubes and Reports\Sample1.ppx"
```

```
    Set objPPCol = objPPRep.Columns.Item("Tents")
    objPPCol.Addition(6)
    objPPRep.Save
    Set objPPCol = Nothing
    Set objPPRep = Nothing
End Sub
```

Related Topics

- "Column Object" on page 15
- "Columns" on page 52
- "Layer Object" on page 28
- "Layers" on page 59
- "Row Object" on page 41
- "Rows" on page 66

AddLevel Method

Adds a row, column, or layer level to a nested crosstab.

Syntax

collection.**Addlevel** CubeCategories, Level, Action, Position

Applies To

Columns

Layers

Rows

Discussion

Use this method to insert a collection of Category objects into a nested crosstab at the current position of the cursor at the specified level. The new level contains the categories in the category list.

The level can be inserted at a nesting level or group level (child categories for a parent category), depending on the setting of the Action parameter. Adding a group level adds the categories to all the categories at the given level that are part of the same group. The level can be either a parent or a child, depending on the setting of the Position parameter.

The level is set using the Level parameter, where 0 indicates the level closest to the actual data (that is, the lowest level row or column), 1 indicates the next level up, and so on. For example, where a financial crosstab has levels for Years and Months and then data, level 0 is Months and level 1 is Years.

AddLevel applies only to nested crosstabs. If the graph displayed is not a nested crosstab, the method generates an error.

In Explorer mode, when you add a level from the same dimension, the Action parameter is ignored.

Parameter	Description
CubeCategories	Required. Specifies a collection of Category objects to be added to the given Dimension (rows, columns, or layers) and the specified level. Type: Object
Level	Required. Specifies the nesting level in which to add the categories. Type: Long
Action	Required. Specifies where to add the level: 1 = add at the group level. 0 = add at the nesting level Note: Set this parameter to 1 (group) when the Position parameter is set to False (parent). Type: Integer
Position	Required. Specifies whether the category is added as a parent or child. False = parent level True = child level Type: Boolean

Return Type

Object

Example

This example adds a new nesting level of categories, as rows, to a report and then adds a new nesting level of columns.

```
Sub Main()
   Dim objCubeCategories As Object
   Dim objPPRep As Object
   Const level_0 = 0
   Const level_1 = 1
   Const add_to_current = 0
   Const add_to_all = 1
   Const as_parent = 0
   Const as_child = 1
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.New "C:\Cubes and Reports\Great Outdoors.mdc",
-1
   objPPRep.ExplorerMode = False
   objPPRep.Visible = True
   Set objCubeCategories = objPPRep.CategoryList()
```

```
   objCubeCategories.Add level_1, "Channels"
   objPPRep.Rows.AddLevel objCubeCategories, level_0,
_
         add_to_all, as_child
   objCubeCategories.Remove
   objCubeCategories.Add level_1, "Locations"
   objPPRep.Columns.AddLevel objCubeCategories, level_0,
_
         add_to_all, as_child
   Set objCubeCategories = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "AddBlanks Method (Reporter)" on page 93

AddLowestLevelCategories Method (Reporter)

Adds the lowest-level categories to a report.

Syntax

object.**AddLowestLevelCategories** [Rollup]

Applies To

Column Object

Columns

Layer Object

Layers

Row Object

Rows

Discussion

If a Reporter report contains a parent category, this method is used to insert the lowest-level child categories after the specified parent category.

This method cannot be performed on a category that does not currently exist in the report, however it can operate on a category previously added in the macro. In this case, the lowest-level child categories are inserted after the current cursor position.

Tip: This method can be useful within the After Doc Open administrative macro if the user wants to open a report that shows the lowest level categories.

Parameters	Description
Rollup	Optional. Specifies whether to add rollup categories to a report. Valid values are True = add rollup categories False = do not add rollup categories Default: True Type: Boolean

Return Type

Nothing

Example

This example adds the lowest-level children of the first row category that is in the report.

```
Sub Main()
   Dim objReport As Object
   Set objReport = GetObject ("c:\Cubes and Reports\Sample2.ppx")
   objReport.ExplorerMode = False
   objReport.Rows.Item(1).AddLowestLevelCategories True
   objReport.Rows.Item(1).Remove
   MsgBox "The parent category has been removed."
   Set objReport = Nothing
End Sub
```

Related Topics

- "Column Object" on page 15
- "Columns" on page 52
- "Layer Object" on page 28
- "Layers" on page 59
- "Row Object" on page 41
- "Rows" on page 66

AddToReport Method

Adds query results to a report.

Syntax

object.**AddToReport** InsertItem, InsertPoint, LevelAction

Applies To

AdvancedQuery Object

FindQuery Object

Discussion

Use this method to add query results to the rows, columns, or layers in a report. You can specify where to insert the subset in the report and how it is to be inserted.

You define a subset using the AdvancedQuery, FindQuery, or ParentageQuery objects, and then use the Execute method to run the query based on the properties specified for the subset definition. This method inserts the subset into the report at a specified location.

The AddtoReport method does not allow you to specify at which level the row or column will be added. A new method, AddToReportAtSpecificNestingLevel, allows you to specify the desired nesting level.

The AddToReport method should be the last component within the subset definition for a query.

Parameter	**Description**
InsertItem	Required. Specifies where to insert a subset in a report. 0 = Row 1 = Column 2 = Layer Type: Integer
InsertPoint	Required. Specifies the index in the report to add the subset. Rows, Columns, and Layers start at index 1. Type: Long
LevelAction	Required. Specifies the action to perform when adding a row, column, or layer into a set. 0 = None (nothing is allowed) 1 = (reserved for future use) 2 = Insert Before (short drop zone nesting (parent)) 3 = Insert After (short drop zone nesting (parent)) 4 = Add sibling category Before (same level) 5 = Add sibling category After (same level) 6 = Group Before (insert a new nesting group - long drop zone) 7 = Group After (insert a new nesting group - long drop zone) Type: Integer Note: Using 0 inserts the subset into the rows, columns or layers where no rows, columns or layers currently exist. If the rows, columns or layers exist, then using this value does nothing.

Return Type

Integer

Example

This example creates a FindQuery (type 1) subset definition that searches for all products that begin with the name "Star". The subset of Products that the query finds beginning with Star is then added to the report as columns.

```
Sub Main()
   Dim strCubePath As String
   Dim objPPRep As Object
   Dim objFind As Object
   Dim objAdvanced As Object
   strCubePath = "C:\Cubes and Reports\Great Outdoors.mdc"
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.New strCubePath, 1
   objPPRep.ExplorerMode = False
   objPPRep.Visible = True
   Set objFind = objPPRep.ReportQueries.Add(1)
   With objFind
      .Name = "Find Star"
      .Dimension = "Products"
      .SearchShortName = False
      .SearchText = "Star"
      .Pattern = 2
   End With
   Set objAdvanced = objPPRep.ReportQueries.Add(3)
   With objAdvanced
      .Name = "Star Products"
      .Dimension = "Products"
      .Level "Product Id"
      .Find objFind.Name
      .Execute
      .AddToReport 1,1,3
   End With
   Set objAdvanced = Nothing
   Set objFind = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "Execute Method" on page 143

AddToReportAtSpecificNestingLevel Method

Adds query results to a report at a specific nesting level.

Syntax

object.**AddToReportAtSpecificNestingLevel** InsertItem, InsertPoint, LevelAction

Applies To

AdvancedQuery Object

FindQuery Object

ParentageQuery Object

Discussion

Use this method to add query results to the rows or columns in a report. You can specify where to insert the subset in the report and how it is to be inserted. Also, you can specify at which nesting level to insert the query results.

You define a subset using the AdvancedQuery, FindQuery, or ParentageQuery objects, and then use the Execute method to run the query based on the properties specified for the subset definition. This method inserts the subset into the report at a specified location.

The AddToReportAtSpecificNestingLevel method should be the last component within the subset definition for a query.

Parameter	**Description**
InsertItem	Required. Specifies where to insert a subset in a report. 0 = Row 1 = Column Type: Integer
InsertPoint	Required. Specifies the index in the report to add the subset. Rows, Columns, and Layers start at index 1. Type: Long

Parameter	Description
LevelAction	Required. Specifies the action to perform when adding a row or column into a set. 0 = None (nothing is allowed) 1 = (reserved for future use) 2 = Insert Before (short drop zone nesting (parent)) 3 = Insert After (short drop zone nesting (parent)) 4 = Add sibling category Before (same level) 5 = Add sibling category After (same level) 6 = Group Before (insert a new nesting group - long drop zone) 7 = Group After (insert a new nesting group - long drop zone) Type: Integer Note: Using 0 inserts the subset into the rows or columns where no rows or columns currently exist. If the rows or columns exist, then using this value does nothing.
InsertLevel	Required. Specifies the nesting level at which to insert the query results. The numbering for the OLE nesting levels starts at the category furthest from the report edge. For example, for a report that contains the following nesting on the row axis: Products -> Retailers -> Locations The corresponding OLE nesting levels will be: Locations - 1 Retailers - 2 Products - 3 Type: Integer

Return Type

Integer

Example

This example creates a FindQuery (type 1) subset definition that searches for all products that begin with the name "Star". The subset of Products that the query finds beginning with Star is then added to the report as columns.

```
Sub Main()
   Dim strCubePath As String
   Dim objPPRep As Object
   Dim objFind As Object
   Dim objAdvanced As Object
```

```
   strCubePath = "C:\Cubes and Reports\Great Outdoors.mdc"
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.New strCubePath, 1
   objPPRep.ExplorerMode = False
   objPPRep.Visible = True
   Set objFind = objPPRep.ReportQueries.Add(1)
   With objFind
      .Name = "Find Star"
      .Dimension = "Products"
      .SearchShortName = False
      .SearchText = "Star"
      .Pattern = 2
   End With
   Set objAdvanced = objPPRep.ReportQueries.Add(3)
   With objAdvanced
      .Name = "Star Products"
      .Dimension = "Products"
      .Level "Product Id"
      .Find objFind.Name
      .Execute
      .AddToReportAtSpecificNestingLevel 1,1,3
   End With
   Set objAdvanced = Nothing
   Set objFind = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "Execute Method" on page 143
- "AddToReport Method" on page 101

Average Method (Collections) (Reporter)

Returns the average of a constant value or a category, and either one or more categories.

Syntax

collection.**Average** [Operand]

Applies To

Columns

Layers

Rows

Discussion

This method is only available if the Report object is in Reporter mode (ExplorerMode property set to False).

Depending on whether the method was applied to an object or a collection, the results are returned respectively as an object or a collection. The new calculation is inserted directly after the active row or column.

References to the position of an object in the collection are not valid after you use this method.

Parameters	Description
Operand	Required. Specifies either a constant value or a category object. Type: Variant

Return Type

Object

Example

This example determines the average of the first three rows and the average of column one and four in a Reporter report.

```
Sub Main()
   Dim objPPRep As Object
   Dim objNewCol As Object
   Dim objNewRow As Object
   Set objPPRep = GetObject( , "CognosPowerPlay.Report")
   objPPRep.ExplorerMode = False
   Set objNewRow = objPPRep.Rows.Subset(1, 3).Average
   MsgBox "The average of the first three rows is " _
      &objPPRep.CellValue(objNewRow(1).Index,1)
   Set objNewCol = objPPRep.Columns.Item(4).Average _
      (objPPRep.Columns.Item(1))
   MsgBox " The average of column one and column four
is " _
      &objPPRep.CellValue(1,objNewCol.Index)
   objPPRep.Save
   Set objNewRow = Nothing
   Set objNewCol = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "Column Object" on page 15
- "Columns" on page 52
- "Layer Object" on page 28

- "Layers" on page 59
- "Row Object" on page 41
- "Rows" on page 66

Average Method (Objects) (Reporter)

Determines the average between a constant value or another category and an object.

Syntax

object.**Average**(Operand)

Applies To

Column Object

Layer Object

Row Object

Discussion

The method calculates the average for each category and operand pair, and creates a new category for the result.

This method is only available if the Report object is in Reporter Mode (ExplorerMode property set to False).

Depending on whether the method was applied to an object or a collection, the results are returned respectively as an object or a collection. The new calculation is inserted directly after the active row or column.

References to the position of an object in the collection are not valid after you use this method.

Parameters	Description
Operand	Required. Specifies either a constant value or a category object. Type: Variant

Return Type

Object

Example

This example returns the average of values in one column and a constant value in a new column.

```
Sub Main()
   Dim objPPRep as Object
   Dim objPPCol as Object
```

```
   Set objPPRep = CreateObject ("CognosPowerPlay.Report")
   objPPRep.Open "C:\Cubes and Reports\Sample1.ppx"
   objPPRep.ExplorerMode = False
   Set objPPCol = objPPRep.Columns.Item("Tents")
   objPPCol.Average(67000)
   objPPRep.Save
   Set objPPCol = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "Column Object" on page 15
- "Columns" on page 52
- "Layer Object" on page 28
- "Layers" on page 59
- "Row Object" on page 41
- "Rows" on page 66

CanDrillDown Method

Returns whether you can drill down the category object.

Syntax

object.**CanDrillDown**

Applies To

Column Object

Layer Object

Row Object

Discussion

Use this method to determine whether you can show child categories. A drill down category is the immediate descendant (child) of a category that defines the properties of a drill-down path.

If this method is True, you can drill down the category. If False, you will receive an error when you attempt to drill down the category. Use the DrillDown method on the category to drill down to the next level.

Return Type

Boolean

Example

This example determines whether you can drill down on a row, and if you can, drills down to the next level.

```
Sub Main()
```

```
   Dim objPPRep as Object
   Dim objPPCol as Object
   Dim objPPRow as Object
   Set objPPRep = GetObject( , "CognosPowerPlay.Report")
   Set objPPCol = objPPRep.Columns.Item("Outdoor Products")
   Set objPPRow = objPPRep.Rows.Item("1997")
   If objPPCol.CanDrillDown Then
      objPPCol.DrillDown
   End If
   If objPPRow.CanDrillUp Then
      objPPRow.DrillUp
   End If
   Set objPPRow = Nothing
   Set objPPCol = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "Column Object" on page 15
- "Layer Object" on page 28
- "Row Object" on page 41

CanDrillUp Method

Returns whether you can drill up the category object.

Syntax

object.**CanDrillUp**

Applies To

Column Object

Layer Object

Row Object

Discussion

Use this method to determine whether you can remove the child categories and add the parent and sibling categories. You can drill up in any hierarchy where you have drilled down. For example, you can drill up to Locations from its child category Europe. Drilling up gives you a broader perspective of a Dimension object.

If this method returns True, you can drill up on the category. If False, you will receive an error when you attempt to drill up on the category. Use the DrillUp method on the category to drill up to the next level.

Return Type

Boolean

Example

This example determines whether you can drill up on a column, and if you can, drills down to the next level.

```
Sub Main()
   Dim objPPRep as Object
   Dim objPPCol as Object
   Dim objPPRow as Object
   Set objPPRep = GetObject( , "CognosPowerPlay.Report")
   Set objPPCol = objPPRep.Columns.Item("Outdoor Products")
   Set objPPRow = objPPRep.Rows.Item("1997")
   If objPPCol.CanDrillDown Then
      objPPCol.DrillDown
   End If
   If objPPRow.CanDrillUp Then
      objPPRow.DrillUp
   End If
   Set objPPRow = Nothing
   Set objPPCol = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "Column Object" on page 15
- "Layer Object" on page 28
- "Row Object" on page 41

Category Method

Sets the parent category or dimension for the ParentageQuery subset definition.

Syntax

ParentageQuery.**Category**(Category[,AncestorOfCategory])

Applies To

ParentageQuery Object

Discussion

Use this method to specify the dimension name or category name for the ParentageQuery to start the query. The name of the category or dimension used is considered to be the parent.

Note: The order of the components in the subset definition is important. First, specify the Category method, followed by the LevelsDown and LowestLevels properties, and then the Execute and AddToReport methods. Set the Name property anywhere within the subset definition.

Parameter	Description
Text	Required. Specifies the dimension or category name of the string to search for, or you can use it to return the name of the category found based on the name you specified. Type: String
AncestorOfCategory	Optional. Specifies the ancestor of the category when the category is not unique. Type: Variant

Return Type

None

Example

This example creates a ParentageQuery (type 2) subset definition that returns all categories one level below Channels. The categories that are one level below Channels are then added to the report as the first nesting level of rows.

```
Sub Main()
   Dim strCubePath As String
   Dim objPPRep As Object
   Dim objCategory As Object
   Dim objParent As Object
   strCubePath = "C:\Cubes and Reports\Great Outdoors.mdc"
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.New strCubePath, 1
   objPPRep.ExplorerMode = False
   objPPRep.Visible = True
   Set objParent = objPPRep.ReportQueries.Add(2)
   With objParent
      .Name = "Sales Channels"
      .Category "Channels"
      .LowestLevel = False
      .LevelsDown = 1
      .Execute
      .AddToReport 0,1,6
   End With
   Msgbox "The first category added was " & _
      objParent.Item(1).Name & ".", ,"Subset"
   Set objParent = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "ParentageQuery Object" on page 31

CategoryList Method

Creates a CategoryList object used to identify categories to be inserted in the Report object.

Syntax

Report.**CategoryList**

Applies To

Report Object

Discussion

To add categories to a report, first create a CategoryList object by using the CategoryList method in the report and then use it to identify the required categories. These can be existing categories found in a cube, or new ones created when the CategoryList object is passed to the report. Use the Add method to select existing categories from the MDC file. When you set the Average, Intersection, or Sum properties to True, each property creates a new category to determine the average, intersection, and sum of the selected category.

Use the Add method to add the identified categories to the report.

References to the position of an object in the collection are not valid after you use this method.

Return Type

Object

Example

This example creates a report and attaches a cube to it. Then it adds existing categories from the YEARS dimension to the rows, adds existing categories from the PRODUCTS dimension to the columns, and adds a new average category to the columns.

```
Sub Main()
   Dim objReport as Object
   Dim objCatList as Object
   Set objReport = CreateObject("CognosPowerPlay.Report")
   objReport.New "C:\Cubes and Reports\Great Outdoors.mdc",
False
   objReport.Visible = True
   Set objCatList = objReport.CategoryList()
   objCatlist.Add 1, "Years", "2008"
   objCatlist.Each = True
   objReport.Rows.Add objCatList
   objCatlist.Remove
   objCatlist.Add 1, "Products"
   objCatlist.Each = True
   objCatList.Average = True
   objReport.columns.Add objCatList
```

```
    objReport.SaveAs "New Report.ppx"
    Set objCatList = Nothing
    Set objReport = Nothing
End Sub
```

Related Topics

- "CategoryList Object" on page 13
- "ActiveReport Method" on page 82
- "Report Object" on page 37

CellValue Method

Gets the value of a cell in a report.

Syntax

Report.**CellValue**([Row][, Column][, Layer])

Applies To

Report Object

Discussion

If any parameter is left out, the method uses the currently active row, column, or layer. You can use the Activate method to move the cursor to the appropriate cell.

If the current cursor position is not on a cell containing data, the method can fail.

Parameters	Description
Row	Optional. Specifies the row number as an index or Row object the cell is in. Type: Integer or Object
Column	Optional. Specifies the column number or Column object the cell is in. Type: Integer or Object
Layer	Optional. Specifies the layer number or Layer object the cell is in. Type: Integer or Object

Return Type

Double

Example

This example opens a report, and determines if independent stores met expectations for Outdoor products in the first quarter of 2008 and shows a message to indicate whether it did.

```
Sub Main()
   Dim objPPRep As Object
   Dim dValue As Double
   Set objPPRep = GetObject("C:\Cubes and Reports\Sample1.ppx")
   dValue = objPPRep.CellValue(objPPRep.Rows.item("2008
Q 1"), _
      objPPRep.Columns.item("Outdoor Products"), _
      objPPRep.Layers.item("Independent")  )
   If dValue > 100000 Then
      MsgBox "Results met expectations"
  Else
      MsgBox "Results are below expectations"
  End If
  Set objPPRep = Nothing
End Sub
```

Related Topics

- "Column Object" on page 15
- "Layer Object" on page 28
- "ActiveReport Method" on page 82
- "Report Object" on page 37
- "Row Object" on page 41

Change Method

Changes the current category for the Dimension object.

Syntax

Dimension.**Change**(CategoryLabel)

Applies To

Dimension Object

Discussion

Use this method to focus on a different category level. The report then shows either more or less detail about the key performance indicator. Changing the current category for the Dimension object, is the same as changing the filter for the Dimension.

References to the position of an object in the collection are not valid after you use this method.

Parameters	Description
CategoryLabel	Required. Specifies the category label or index to be made active. Type: String

Return Type

Nothing

Example

This example changes two of the current categories for the DimensionLine object in the current report.

```
Sub Main()
   Dim objPPRep as Object
   Set objPPRep = GetObject ("C:\Cubes and Reports\Sample1.ppx")
   objPPRep.DimensionLine.Item("Years").Change("1997")
   objPPRep.DimensionLine.Item("Products").Change _
      ("Go Sport Line")
   objPPRep.Save
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "Dimension Object" on page 19

ChangeToParent Method

Changes the current category for the Dimension object to the category one level higher in the hierarchy.

Syntax

Dimension.**ChangeToParent**

Applies To

Dimension Object

Discussion

Use this method to identify a higher-level category in the hierarchy that summarizes the current-level categories, first. If it does not find any categories, it looks for the first higher-level category in the hierarchy that did not summarize the current-level categories.

References to the position of an object in the collection are not valid after you use this method.

Return Type

Nothing

Example

This example changes the current category for the Measures dimension to the category one level higher in the hierarchy.

```
Sub Main()
   Dim objPPRep As Object
   Dim objDimension as Object
   Set objPPRep = GetObject(,"CognosPowerPlay.Report")
   Set objDimension = objPPRep.DimensionLine.Item("Measures")
   objDimension.ChangeToParent
   objDimension.Change "Product Cost"
   objDimension.BlankWhenZero = True
   objDimension.BlankWhenMissing = False
   objDimension.BlankWhenDividedByZero = False
   Set objDimension = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "Dimension Object" on page 19

ChangeToTop Method

Changes the current category for the Dimension object to top-level category.

Syntax

Dimension.**ChangeToTop**

Applies To

Dimension Object

Discussion

Use this method to go directly to the top-level category for the dimension.

References to the position of an object in the collection are not valid after you use this method.

Return Type

Nothing

Example

This example changes the current category of the Measures dimension to the category at the top of the hierarchy.

```
Sub Main()
```

```
   Dim objPPRep As Object
   Dim objDimension as Object
   Set objPPRep = GetObject(,"CognosPowerPlay.Report")
   Set objDimension = objPPRep.DimensionLine.Item("Measures")
   objDimension.ChangeToTop
   objDimension.Change "Product Cost"
   objDimension.BlankWhenZero = True
   objDimension.BlankWhenMissing = False
   objDimension.BlankWhenDividedByZero = False
   Set objDimension = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "Dimension Object" on page 19

Children Method

Returns the next child in the hierarchy for an object.

Syntax

object.**Children**

Applies To

Column Object

Dimension Object

Layer Object

Row Object

Discussion

Use this method to obtain the children for the Column, Dimension, Layer, and Row objects. The Children method accesses the different children for the current category. For example, the 1998 Q1 category has three children; 1998/January, 1998/February, and 1998/March.

For Dimensions, use the Change method to point to a specific category, the ChangeToTop method to point to the top level, and the ChangeToParent method to point one level higher in the hierarchy.

Return Type

Object

Example

This example gets the name of the children of the first category in the columns collection, then returns the number of items and names of the children in the collection.

```
Sub Main()
   Dim objPPRep As Object
   Dim objChildrenCols As Object
   Dim strColChild As String
   Dim strColChildren As String
   Dim intx As Integer
   Set objPPRep = GetObject(,"CognosPowerPlay.Report")
   Set objChildrenCols = objPPRep.Columns.Item(1).Children
   For intx = 1 to objChildrenCols.Count
      strColChild = objChildrenCols.Item(intx).Name
      strColChildren = strColChildren & chr$(10)
& _
         strColChild
   Next intx
   MsgBox "The " & objPPRep.Columns.Item(1).Name
& _
      " category has " & objChildrenCols.Count &
_
      " children." & chr$(10) & chr$(10) &
"They are: " & _
      chr$(10) & strColChildren, , "Column Children"
   Set objChildrenCols = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "Change Method" on page 115
- "ChangeToParent Method" on page 116
- "ChangeToTop Method" on page 117
- "Child Object" on page 14
- "Children" on page 49
- "Column Object" on page 15
- "Dimension Object" on page 19
- "Layer Object" on page 28
- "Row Object" on page 41

Close Method

Closes one or all the Reports objects.

Syntax

object.**Close**

Applies To

Report Object

Reports

Discussion

When you use the Close method on a Report object, it closes without saving any modifications made since it was last saved. When you use the Close method on a Reports collection, it closes all reports without saving any modifications. If the application or reports are visible and have been modified, you are prompted to save before closing. If the application and reports are invisible, you are not prompted to save before closing.

This method breaks any OLE connections to the report. To open the report again or any other report, you must use CreateObject("CognosPowerPlay.Report") followed by the New or Open method.

Return Type

Nothing

Example

This example closes the open report without saving it.

```
Sub Main()
   Dim objPPRep As Object
   Dim objPPCol As Object
   Set objPPRep = GetObject("C:\Cubes and Reports\Sample1.ppx")
   Set objPPCol = objPPRep.Columns.Item(1)
   Set objPPCol = objPPRep.Columns(1)
   MsgBox "The name of the first column is " _
      &objPPRep.Columns.Item(1).Name & " using
the Item method."
   MsgBox "The name of the first column is " _
      &objPPRep.Columns(1).Name & " using column
index."
   objPPRep.Close
   Set objPPCol = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "ActiveReport Method" on page 82
- "Report Object" on page 37
- "Reports" on page 65
- "Reports Method" on page 212

Columns Method

Returns a collection that contains all the Column objects.

Syntax

Report.**Columns**

Applies To

Report Object

Discussion

If no index is specified all the Column objects in the collection are returned, otherwise the method returns the specified Column object.

Return Type

Object

Example

This example shows two different ways of returning the same Column object from an existing report.

```
Sub Main()
   Dim objPPRep As Object
   Dim objPPCol As Object
   Set objPPRep = GetObject("C:\Cubes and Reports\Sample1.ppx")
   Set objPPCol = objPPRep.Columns.Item(1)
   Set objPPCol = objPPRep.Columns(1)
   MsgBox "The name of the first column is " _
      &objPPRep.Columns.Item(1).Name & " using
the Item method."
   MsgBox "The name of the first column is " _
      &objPPRep.Columns(1).Name & " using column
index."
   objPPRep.Close
   Set objPPCol = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "Column Object" on page 15
- "Columns" on page 52
- "ActiveReport Method" on page 82
- "Report Object" on page 37

Copy Method

Copies the Row, Column, or Layer objects currently selected in the Report object to the Clipboard.

Syntax

Report.**Copy**

Applies To

Report Object

Discussion

Use the Select method to select the categories you want to copy.

Return Type

Nothing

Example

This example opens two reports, copies columns from one report, pastes them into the second report, and saves the second report.

```
Sub Main()
   Dim objPPRep1 as Object
   Dim objPPRep2 as Object
   Set objPPRep1 = CreateObject("CognosPowerPlay.Report")
   Set objPPRep2 = CreateObject("CognosPowerPlay.Report")
   objPPRep1.Open "C:\Cubes and Reports\Sample1.ppx"
   objPPRep2.Open "C:\Cubes and Reports\Sample2.ppx"
   objPPRep1.ExplorerMode = False
   objPPRep2.ExplorerMode = False
   objPPRep1.Columns.Item("Back Packs").Select
   objPPRep1.Copy
   objPPRep2.Paste
   objPPRep2.Save
   objPPRep1.Close
   objPPRep2.Close
   Set objPPRep1 = Nothing
   Set objPPRep2 = Nothing
End Sub
```

Related Topics

- "Column Object" on page 15
- "Layer Object" on page 28
- "Report Object" on page 37
- "Row Object" on page 41
- "Select Method" on page 220

CumPercentOfBase Method

Adds one or more cumulative percent of base categories using a category from a different dimension as the base.

Syntax

object.**CumPercentOfBase(BaseCategory)**

Applies To

Column Object

Columns

Row Object

Rows

Discussion

To determine the cumulative percent of base for a row, you can only use a column as the base category. To determine the cumulative percent of base for a column, you can only use a row as the base category.

Depending on whether the method was applied to an object or a collection, the results are returned respectively as an object or a collection.

In Explorer mode, the new calculation is inserted directly after the last operand. In Reporter mode, the new calculation is inserted directly after the active row or column.

References to the position of an object in the collection are not valid after you use this method.

In Explorer mode, if you change a report by removing a level, drilling, filtering or nesting, then all calculations that can not be created disappear.

Parameters	Description
BaseCategory	Required. Specifies the category on which the calculation is based. Type: Object

Return Type

Object

Example

This example calculates the cumulative percent of base for the Columns collection, using 2008 as the base category.

```
Sub Main()
   Dim objPPRep as Object
   Dim objPPRes as Object
   Set objPPRep = GetObject(, "CognosPowerPlay.Report")
   Set objPPRes = objPPRep.Columns.CumPercentOfBase _
      (objPPRep.Rows.Item("2008"))
   objPPRep.SaveAs "MyNewReport"
   Set objPPRep = Nothing
End Sub
```

Related Topics

- “Column Object” on page 15
- “Columns” on page 52
- “Row Object” on page 41

- "Rows" on page 66

Cut Method (Reporter)

Moves the Row, Column, or Layer objects currently selected in the Report object to the Clipboard.

Syntax

Report.**Cut**

Applies To

Report Object

Discussion

Use the Select method to select the categories you want to cut.

References to the position of an object in the collection are not valid after you use this method.

This method is only available if the Report object is in Reporter Mode (the ExplorerMode property is False).

Return Type

Nothing

Example

This example opens a report, cuts the column "Outdoor Products", and saves the report.

```
Sub Main()
   Dim objPPRep as Object
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.Open "C:\Cubes and Reports\Sample1.ppx"
   objPPRep.ExplorerMode = False
   objPPRep.Columns.Item ("Outdoor Products").Select
   objPPRep.Visible = True
   objPPRep.Cut
   objPPRep.Save
   objPPRep.Close
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "Column Object" on page 15
- "Layer Object" on page 28
- "Report Object" on page 37
- "Row Object" on page 41
- "Select Method" on page 220

DeleteExplorerRank Method

Deletes the rank category from an Explorer report.

Syntax

collection.**DeleteExplorerRank**

Applies To

Columns

Rows

Discussion

Use this method to remove the row or column that shows the rank ordinals for a category. You can remove rank categories when you update a report and no longer require this detail, or if you add nested categories to the report and must recalculate the category rank ordinals.

If an Explorer report includes a column that shows the rank ordinals for a category, use the Columns collection and this property to permanently delete the category. If an Explorer report includes a row that shows the rank ordinals for a category, use the Rows collection and this property to permanently delete the category.

This property does not delete the rank category if the report is in Reporter mode. To delete a rank category from a Reporter report, use the DeleteSelected method.

Return Type

Integer

Example

This example removes the column that shows the rank ordinals for rows in the active report.

```
Sub Main()
   Dim objPPRep As Object
   Dim objRslt As Object
   Set objPPRep = GetObject( , "CognosPowerPlay.Report")
   objPPRep.Columns.DeleteExplorerRank
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "Rank2 Method" on page 204

DeleteAllDataSourceInfo Method

Deletes security access information for all PowerCube data sources from memory.

Syntax

Application.**DeleteAllDataSourceInfo**

Applies To

Application Object

Discussion

Use this method to remove the access information records for all data sources stored in memory by the SetDataSourceInfo method.

To delete access information for a single data source from memory, use the DeleteDataSourceInfo method.

This method can be used to prevent unauthorized users from opening subsequent reports based on the same data source after all desired reports are open. A good situation to use this method is AfterDocOpen macro for the last report to be opened.

Return Type

Nothing

Example

This example opens two reports, each based on a different data source, and then removes all security access information that has been stored for both of these data sources.

```
Sub Main()
     Dim objPPApp As Object
     Dim objPPRep1 As Object
     Dim objPPRep2 As Object
     Dim strMDCName As String
     Set objPPRep1 = CreateObject("CognosPowerPlay.Report")
     Set objPPRep2 = CreateObject("CognosPowerPlay.Report")
     Set objPPApp = objPPRep.Application
     strMDCName = "C:\Cubes and Reports\Sample1.mdc"
     objPPApp.SetDataSourceInfo "local", strMDCName,
"cube_password1"
     objPPRep1.Open("C:\Cubes and Reports\Sample1.ppx")
     strMDCName = "C:\Cubes and Reports\Sample2.mdc"
     objPPApp.SetDataSourceInfo "local", strMDCName,
"cube_password2"
     objPPRep2.Open("C:\Cubes and Reports\Sample2.ppx")
     objPPApp.DeleteAllDataSourceInfo
     Set objPPRep = Nothing
     Set objPPApp = Nothing
End Sub
```

Related Topics

- "Application Object" on page 11

DeleteAllMDCAccessInfo Method

Deletes security access information for all local PowerCubes from memory.

Syntax

Application.**DeleteAllMDCAccessInfo**

Applies To

Application Object

Discussion

Use this method to remove all access information records stored in memory by the SetMDCAccessInfo method.

To delete the access information record for a single PowerCube, use the DeleteMDCAccessInfo method.

Use the DeleteAllMDCAccessInfo method to prevent unauthorized users from opening subsequent reports based on the same PowerCube after all desired reports are open. A good situation to use this method is in the after doc open macro for the last report to be opened.

Return Type

Nothing

Example

This example opens two reports, each based on a different PowerCube, and then removes all security access information that has been stored for both cubes.

```
Sub Main()
     Dim objPPApp As Object
     Dim objPPRep1 As Object
     Dim objPPRep2 As Object
     Dim strMDCName As String
     Set objPPRep1 = CreateObject("CognosPowerPlay.Report")
     Set objPPRep2 = CreateObject("CognosPowerPlay.Report")
     Set objPPApp = objPPRep.Application
     strMDCName = "C:\Cubes and Reports\Sample1.mdc"
     objPPApp.SetMDCAccessInfo strMDCName, ", "cube_password1"
     objPPRep1.Open("C:\Cubes and Reports\Sample1.ppx")
     strMDCName = "C:\Cubes and Reports\Sample2.mdc"
     objPPApp.SetMDCAccessInfo strMDCName, ", "cube_password2"
     objPPRep2.Open("C:\Cubes and Reports\Sample2.ppx")
     objPPApp.DeleteAllMDCAccessInfo
     Set objPPRep = Nothing
     Set objPPApp = Nothing
End Sub
```

DeleteDataSourceInfo Method

Deletes security access information for a PowerCube data source from memory.

Syntax

Application.**DeleteDataSourceInfo**(ConnectionType, Location)

Applies To

Application Object

Discussion

Use this method to remove the access information records, such as cube password, stored in memory by the SetDataSourceInfo method.

To delete all data source access information records from memory, use the DeleteAllDataSourceInfo method.

This method can be used to prevent unauthorized users from opening subsequent reports based on the same data source after all desired reports are open. A good situation to use this method is AfterDocOpen macro for the last report to be opened.

Parameter	Description
Connection Type	Required. Identifies whether the connection type is for a local cube or a remote package. This parameter can be either "local" or "remote". Type: String
Location	Required. If the connection type is local, then a fully qualified local cube is expected. For example, "C:\Cubes\Great Outdoors.mdc" If the connection type is remote, then a package search path in native encoding or a store ID is expected. Search path example, "/content/package[@name=Great Outdoors]" Store ID example, "storeID('iAA1ECBF2EA9B46F78651D4787F219509')" Type: String

Return Type

Boolean

Example

This example sets the cube security access information record and then opens a report based on the password protected cube. Next, the data source security access record is deleted from memory.

```
Sub Main()
     Dim objPPApp As Object
     Dim objPPRep As Object
     Dim strMDCName As String
     Set objPPRep = CreateObject("CognosPowerPlay.Report")
     Set objPPApp = objPPRep.Application
     strMDCName = "C:\Cubes and Reports\Sample1.mdc"
     objPPApp.SetDataSourceInfo "local", strMDCName,
"cube_password"
     objPPRep.Open "C:\Cubes and Reports\Sample1.ppx"
     objPPApp.DeleteDataSourceInfo("local", strMDCName)
     Set objPPRep = Nothing
     Set objPPApp
End Sub
```

Related Topics

- "Application Object" on page 11

DeleteMDCAccessInfo Method

Deletes security access information for a local PowerCube from memory.

Syntax

Application.**DeleteMDCAccessInfo(MDCName)**

Applies To

Application Object

Discussion

Use this method to remove the access information records for a local PowerCube stored in memory by the SetMDCAccessInfo method.

To delete all access information records from memory, use the DeleteAllMDCAccessInfo method.

This method can be used to prevent unauthorized users from opening subsequent reports based on the local PowerCube after all desired reports are open. A good situation to use this method is AfterDocOpen macro for the last report to be opened.

Parameter	Description
MDCName	Required. Specifies the MDC (cube) name of a local PowerCube. This string must match the name that was used in the SetMDCAccessInfo Method. The name is not case sensitive. Type: String

Return Type

Boolean

Example

This example sets the cube security access information record and then opens a report based on the password protected cube. Next, the security access record is deleted from memory.

```
Sub Main()
     Dim objPPApp As Object
     Dim objPPRep As Object
     Dim strMDCName As String
     Set objPPRep = CreateObject("CognosPowerPlay.Report")
     Set objPPApp = objPPRep.Application
     strMDCName = "C:\Cubes and Reports\Sample1.mdc"
     objPPApp.SetMDCAccessInfo strMDCName, ", "cube_password"
     objPPRep.Open "C:\Cubes and Reports\Sample1.ppx"
     objPPApp.DeleteMDCAccessInfo(strMDCName)
     Set objPPRep = Nothing
     Set objPPApp
End Sub
```

Related Topics

- "Application Object" on page 11

DeleteSelected Method

Deletes selected objects from a Report collection.

Syntax

Report.**DeleteSelected**

Applies To

Report Object

Discussion

Use this method in conjunction with the Select method. After objects are selected in a collection using Select, you can remove them using DeleteSelected.

Return Type

Boolean

Example

This example searches for and deletes all rows that begin with "Star."

```
Sub Main()
   Dim objPPRep As Object
   Dim intFound As Integer
   Const begins_with = 2
   Const current_layer = False
   Const rows = 1
   Set objPPRep = GetObject(, "CognosPowerPlay.Report")
   objPPRep.Rows.Item(1).Activate
   intFound = 1
   Do
      intFound = objPPRep.FindNext("Star", begins_with,
_
         current_layer, rows)
      If intFound = -1 Then
         objPPRep.Rows.Active.Select
         objPPRep.DeleteSelected
      End If
   Loop While intFound <> 0
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "Select Method" on page 220

DeploymentOptions Method

Returns the distribution options for a report published to the IBM Cognos portal.

Syntax

Report.**DeploymentOptions**

Applies To

Report Object

Discussion

Use this method so that a report author can set the prompt properties and PDF options for a report published to the IBM Cognos portal. The report author must explicitly set up the deployment options for a report. These options are saved with the report.

Return Type

Object

Example

This example uses the DeploymentOptions method to set up the prompt properties for a report.

```
Sub Main()
   Dim objPPRep as Object
   Dim objDeploymentOptions as Object
   Set objPPRep = GetObject(,"CognosPowerPlay.Report")
   Set objDeploymentOptions = objPPRep.DeploymentOptions
   objDeploymentOptions.PromptForCurrency = True
   objDeploymentOptions.PromptForLongShortNames = True
   objDeploymentOptions.PromptForZeroSuppression = True
   objDeploymentOptions.PromptForSwapRowsAndColumns =
True
   objDeploymentOptions.PromptForDimension(1) = True
   objDeploymentOptions.PromptForDimension(2)= True
   objDeploymentOptions.PromptForDimension("Years")=
True
   objPPRep.Save
   Set objDeploymentOptions = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "PromptForCurrency Property" on page 370
- "PromptForDimension Property" on page 371
- "PromptForLongShortNames Property" on page 372
- "PromptForZeroSuppression Property" on page 374

Depth Method

Returns whether the Graph object is three-dimensional (3D).

Syntax

Graph.**Depth**

Applies To

Graph Object

Discussion

This method applies specifically to Graph object Type 1 (Pie), 3 (Simple Bar), 4 (Clustered Bar), 5 (Stacked Bar), and 8 (Correlation). To set this method, use the Add method for Graph collections, or the SetType method for Graph objects. When the display type is 0, 6, 7 or 9, the depth method is always False. For display type 2, it is always True.

Default: True (Pie, Simple Bar, Clustered Bar, Stacked Bar, Correlation)

Return Type

Boolean

Example

This example changes the display type for the first Graph object to a three-dimensional cluster bar, and displays the settings for the Graph object for an open report.

```
Sub Main()
   Dim objPPRep as Object
   Dim objPPGph as Object
   Set objPPRep = GetObject( , "CognosPowerPlay.Report")
   Set objPPGph = objPPRep.Graphs.Item(1)
   objPPGph.SetType 4, 1, 1
   MsgBox "The Graph object type is " & objPPGph.Type
& "."
   If objPPGph.Depth = -1 Then
      MsgBox "The graph is 3D."
   Else
       MsgBox "The graph is not 3D."
   End If
   Set objPPGph = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "Graph Object" on page 25

DimensionFilter Method

Sets the filter category for an indexed dimension.

Syntax

ValueRestriction.**DimensionFilter** Index, CategoryName

Applies To

ValueRestriction Object

Discussion

Use this method to specify the category name from a specific dimension to filter a value restriction. For a ValueRestriction query, only specify the DimensionFilter method when you require another dimension other than the current dimension line.

Note: For a ValueRestriction, the order of the components is important. The Dimension property must be set before the Measure, Operator, Operand1, Operand2, and Count properties and the DimensionFilter method. The Name property can be set anywhere within the filter definition.

Default: First dimension

Parameter	Description
Index	Required. Specifies the dimension index to return from the dimension line. The DimensionFilter index starts at 1 and increments by 1, from left to right, for each dimension in the dimension line as they appear in the dimension line of the user interface. Type: Integer
CategoryName	Required. Specifies the name of the category to filter for the specified dimension. Type: String

Return Type

Nothing

Example

This example creates an advanced subset that selects countries or regions from the Locations dimension. The value restriction (type 4) limits the results to return only those countries or regions whose Revenue values for Sports Chains are between 25,000 and 100,000.

```
Sub Main()
   Dim strCubePath As String
   Dim objPPRep As Object
   Dim objValue As Object
   Dim objAdvanced As Object
   strCubePath = "C:\Cubes and Reports\Great Outdoors.mdc"
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.New strCubePath, 1
   objPPRep.ExplorerMode = False
   objPPRep.Visible = True
   Set objValue = objPPRep.ReportQueries.Add(4)
   With objValue
      .Name = "25000-100000"
      .Dimension = "Locations"
      .Measure = "Revenue"
      .Operator = "between"
      .Operand1 = 25000
      .Operand2 = 100000
      .DimensionFilter 4, "Sports Chain"
   End With
   Set objAdvanced = objPPRep.ReportQueries.Add(3)
   With objAdvanced
      .Name = "Locations"
```

```
      .Dimension = "Locations"
      .Level "Country or Region"
      .ValueRestriction objValue.Name
      .Execute
      .AddToReport 0,1,3
   End With
   Msgbox "The Dimension Line Settings for this " &
_
         "report are:" & chr$(10) & chr$(10)
& _
         objValue.DimensionSettings, , "Dimension Line"
   Set objAdvanced = Nothing
   Set objValue = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- Chapter 4, "Methods," on page 71
- Chapter 5, "Properties," on page 259

DimensionLine Method

Returns a DimensionLine object for the current Report object.

Syntax

Report.**DimensionLine**

Applies To

Report Object

Discussion

Use this method to find out where you are in the data as the dimension line shows the category used to filter the data from each dimension in the cube. For Explorer reports, the dimension line changes when you drill down or up, or when you filter out unnecessary information. For Reporter reports, the dimension line only changes when you use automation to filter out unnecessary information.

Use the DimensionLine object to get each individual dimension before setting up filters.

Return Type

Object

Example

This example changes two of the current categories for the DimensionLine object in the current report.

```
Sub Main()
   Dim objPPRep as Object
```

```
   Set objPPRep = GetObject ("C:\Cubes and Reports\Sample1.ppx")
   objPPRep.DimensionLine.Item("Years").Change("1997")
   objPPRep.DimensionLine.Item("Products").Change _
      ("Go Sport Line")
   objPPRep.Save
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "DimensionLine Object" on page 21
- "ActiveReport Method" on page 82
- "Report Object" on page 37

Division Method

Divides one or more categories by either a constant value or another category.

Syntax

object.**Division**(Operand [, Reverse])

Applies To

Column Object

Columns

Layer Object

Layers

Row Object

Rows

Discussion

This method always requires an operand since the calculation is done on a pair of values. A category is created for the results of the division. You can reverse the division so the Operand is the divisor.

In Explorer mode, the operands of the calculation must be from the same dimension and from the same axis.

Depending on whether the method was applied to an object or a collection, the results are returned respectively as an object or a collection.

In Explorer mode, the new calculation is inserted directly after the last operand. In Reporter mode, the new calculation is inserted directly after the active row or column.

References to the position of an object in the collection are not valid after you use this method.

In Explorer mode, if you change a report by removing a level, drilling, filtering or nesting, then all calculations that can not be created in the changed report disappear.

Parameters	Description
Operand	Required. Specifies either a constant value or a category object. Type: Variant
Reverse	Optional. Specifies whether the Operand is the divisor or the dividend. Possible values are False = Operand is the divisor True = Operand is the dividend Default: False Type: Boolean

Return Type

Object

Example

This example divides the first five rows by a constant value and divides column one by column four.

```
Sub Main()
   Dim objPPRep As Object
   Dim objRslt As Object
   Set objPPRep = GetObject( , "CognosPowerPlay.Report")
   objPPRep.Rows.SubSet(1,5).Division 6
   Set objRslt = objPPRep.Rows.Item(1).Division _
      (objPPRep.Rows.Item(4))
   MsgBox "The result of dividing column 1 by column
4 is " _
      &objPPRep.CellValue(objRslt,1)
   Set objRslt = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "Column Object" on page 15
- "Columns" on page 52
- "Layer Object" on page 28
- "Layers" on page 59
- "Row Object" on page 41
- "Rows" on page 66

DrillDown Method

Drills down the specified category object.

Syntax

object.**DrillDown**

Applies To

Column Object

Layer Object

Row Object

Discussion

If you can't drill down a category object, you will receive an error message. Use the CanDrillDown property to determine whether you can drill down the category object.

References to the position of an object in the collection are not valid after you use this method.

Return Type

Nothing

Example

This example determines in an open report whether

- the specified Column object can drill down, and if possible, drills down to the next level
- the specified Row object can drill up, and if possible, drills up to the next level.

```
Sub Main()
   Dim objPPRep as Object
   Dim objPPCol as Object
   Dim objPPRow as Object
   Set objPPRep = GetObject( , "CognosPowerPlay.Report")
   Set objPPCol = objPPRep.Columns.Item("Outdoor Products")
   Set objPPRow = objPPRep.Rows.Item("1997")
   If objPPCol.CanDrillDown Then
      objPPCol.DrillDown
   End If
   If objPPRow.CanDrillUp Then
      objPPRow.DrillUp
   End If
   Set objPPRow = Nothing
   Set objPPCol = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- Chapter 4, “Methods,” on page 71
- Chapter 5, “Properties,” on page 259
- “Column Object” on page 15
- “Layer Object” on page 28
- “Row Object” on page 41

DrillUp Method

Drills up the category object.

Syntax

object.**DrillUp**

Applies To

Column Object

Layer Object

Row Object

Discussion

In Explorer mode, this method shows a higher-level category. In Reporter mode, this method removes lower-level categories from a report. If you can't drill up a category object, you will receive an error message. Use the CanDrillUp property to determine whether you can drill up the category object.

References to the position of an object in the collection are not valid after you use this method.

Return Type

Nothing

Example

This example determines in an open report whether

- the specified Column object can drill down, and if possible, drills down to the next level
- the specified Row object can drill up, and if possible, drills up to the next level.

```
Sub Main()
   Dim objPPRep as Object
   Dim objPPCol as Object
   Dim objPPRow as Object
   Set objPPRep = GetObject( , "CognosPowerPlay.Report")
   Set objPPCol = objPPRep.Columns.Item("Outdoor Products")
   Set objPPRow = objPPRep.Rows.Item("1997")
   If objPPCol.CanDrillDown Then
      objPPCol.DrillDown
```

```
   End If
   If objPPRow.CanDrillUp Then
      objPPRow.DrillUp
   End If
   Set objPPRow = Nothing
   Set objPPCol = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "Column Object" on page 15
- "Layer Object" on page 28
- "Row Object" on page 41

Exceptions Method

Returns one Exception object or the entire collection.

Syntax

Report.**Exceptions**

Applies To

Report Object

Discussion

If no index is specified, an Exceptions collection is returned, otherwise the method returns the requested Exception object.

Return Type

Object

Example

This example shows two different ways of returning the same Exception object from an existing report.

```
Sub Main()
   Dim objPPRep As Object
   Dim objPPExp As Object
   Set objPPRep = GetObject("C:\Cubes and Reports\Exception.ppx")
   Set objPPExp = objPPRep.Exceptions.Item(1)
   MsgBox "The exception is " &objPPRep.Exceptions.Item(1).Name
   Set objPPExp = objPPRep.Exceptions(1)
   MsgBox "The exception is " &objPPRep.Exceptions(1).Name
   Set objPPExp = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "Exception Object" on page 22
- "Exceptions" on page 55
- "ActiveReport Method" on page 82
- "Report Object" on page 37

Exclude Method

Sets the categories to exclude from the query.

Syntax

AdvancedQuery.**Exclude** CategoryName [,ParentName]

Applies To

AdvancedQuery Object

Discussion

Use this method in the subset definition to identify the categories to exclude from the query. To exclude more than one category from the query, include multiple Exclude statements in the subset definition for AdvancedQuery object.

If the resulting query finds more than one category that matches the specified label, specify the optional parameter, which can be the parent category or drill-down path of the desired category. The functionality is added to identify the category. For example:

```
Exclude("CategoryName", "ParentName or drill-down path
name")
```

Note: The order of the components in the subset definition is important. First, specify the Dimension property, followed by the Level method. The Include, Exclude, and Find methods are optional and can appear in any order following the Dimension and Level properties. The Name property is required and can be set anywhere within the subset definition. Specify Execute, followed by the AddToReport method last.

Parameter	Description
CategoryName	Required. Specifies the name of the category to exclude from the query. Type: String
ParentName	Optional. Specifies the name of the ancestor category of the category to exclude from the subset. For example, two countries or regions may have the same city name. Use the country or region name (the ancestor) to differentiate between the two cities. Type: String

Return Type

Nothing

Example

This example creates an AdvancedQuery (type 3) subset definition that retrieves all categories except those belonging to Europe. This subset is then added to the report as layers.

```
Sub Main()
   Dim strCubePath As String
   Dim objPPRep As Object
   Dim objCatList As Object
   Dim objAdvanced As Object
   strCubePath = "C:\Cubes and Reports\Great Outdoors.mdc"
      Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.New strCubePath, 1
   objPPRep.ExplorerMode = False
   objPPRep.Visible = True
   Set objCatList = objPPRep.CategoryList()
   objCatList.Add 0,"Locations"
   objPPRep.Layers.Add objCatList
   Set objAdvanced = objPPRep.ReportQueries.Add(3)
   With objAdvanced
      .Name = "Americas & Far East"
      .Dimension = "Locations"
      .Level "Country or Region"
      .Exclude "Europe"
      .Execute
      .AddToReport 2,1,4
   End With
   Msgbox "Name: " & objAdvanced.Name & chr$(10)
& _
      "Dimension: " & objAdvanced.Dimension &
chr$(10) & _
      "Level List: " & objAdvanced.LevelList &
chr$(10) & _
      "Query Type Code: " & objAdvanced.Type &
chr$(10) & _
      "Number of Categories: " & objAdvanced.Count
& _
      chr$(10) & _
      "First Category: " & objAdvanced.Item(1).Name,
_
      ,"Subset"
   Set objAdvanced = Nothing
   Set objCatList = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "AdvancedQuery Object" on page 8
- "Include Method" on page 162

Execute Method

Runs a query on the cube.

Syntax

object.**Execute**

Applies To

AdvancedQuery Object

FindQuery Object

ParentageQuery Object

Discussion

Use this method to run the query to create a data subset. Subsets are defined by creating the object (AdvancedQuery, FindQuery, or ParentageQuery) and setting the applicable properties for the type of query. This resulting subset definition is run using the Execute method.

Return Type

Nothing

Example

This example runs an AdvancedQuery (type 3) subset definition that retrieves all categories except those belonging to Europe. This subset is then added to the report as layers.

```
Sub Main()
   Dim strCubePath As String
   Dim objPPRep As Object
   Dim objCatList As Object
   Dim objAdvanced As Object
   strCubePath = "C:\Cubes and Reports\Great Outdoors.mdc"
      Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.New strCubePath, 1
   objPPRep.ExplorerMode = False
   objPPRep.Visible = True
   Set objCatList = objPPRep.CategoryList()
   objCatList.Add 0,"Locations"
   objPPRep.Layers.Add objCatList
   Set objAdvanced = objPPRep.ReportQueries.Add(3)
   With objAdvanced
      .Name = "Americas & Far East"
      .Dimension = "Locations"
```

```
      .Level "Country"
      .Exclude "Europe"
      .Execute
      .AddToReport 2,1,4
   End With
   Msgbox "Name: " & objAdvanced.Name & chr$(10)
& _
      "Dimension: " & objAdvanced.Dimension &
chr$(10) & _
      "Level List: " & objAdvanced.LevelList &
chr$(10) & _
      "Query Type Code: " & objAdvanced.Type &
chr$(10) & _
      "Number of Categories: " & objAdvanced.Count
& _
      chr$(10) & _
      "First Category: " & objAdvanced.Item(1).Name,
_
      ,"Subset"
   Set objAdvanced = Nothing
   Set objCatList = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "AddToReport Method" on page 101

Exponentiation Method

Raises one or more categories to the power of either another category or a constant value.

Syntax

object.**Exponentiation**(Operand [, Reverse])

Applies To

Column Object

Columns

Layer Object

Layers

Row Object

Rows

Discussion

This method always requires an Operand since the calculation is done on a pair of values. A category is created for the result of the exponentiation. You can reverse the exponentiation.

Depending on whether the method was applied to an object or a collection, the results are returned respectively as an object or a collection. The new calculation is inserted directly after the active row or column.

This method can also raise a constant value to the power of one or more categories.

References to the position of an object in the collection are not valid after you use this method. If you change a report by removing a level, drilling, filtering or nesting, then all calculations that can not be created in the changed report disappear.

Note: In Explorer mode, you cannot perform this calculation on a group of categories. If more than one category is selected, the Exponentiate calculation is disabled.

Parameters	Description
Operand	Required. Specifies either a constant value or a category object. Type: Variant
Reverse	Optional. Specifies whether the Operand is the value to be exponentiated or the exponent. If True, the Operand is the value to be exponentiated. If False, it is the exponent. Default: False. Type: Boolean

Return Type

Object

Example

This example raises the values in one column to the power of a constant value and returns the results in a new column.

```
Sub Main()
   Dim objPPRep as Object
   Dim objPPCol as Object
   Set objPPRep = CreateObject ("CognosPowerPlay.Report")
   objPPRep.Open "C:\Cubes and Reports\Sample1.ppx"
   objPPRep.ExplorerMode = False
   Set objPPCol = objPPRep.Columns.Item("Tents")
```

```
   objPPCol.Exponentiation(2)
   objPPRep.Save
   Set objPPCol = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "Column Object" on page 15
- "Columns" on page 52
- "Layer Object" on page 28
- "Layers" on page 59
- "Row Object" on page 41
- "Rows" on page 66

Find Method

Specifies the name of the FindQuery object to include in an AdvancedQuery.

Syntax

AdvancedQuery.**Find** FindSubsetName

Applies To

AdvancedQuery Object

Discussion

Use this method to attach a FindQuery object to the AdvancedQuery object. First, you must create the FindQuery object, then it must be assigned a name. Find filters the query for the AdvancedQuery object by matching the query name and validating its dimension name.

Find can be broken down into two main functional areas: finding within cube data and finding within report data. When dealing with large amounts of data in a report, Find can locate an individual category matching all specified search criteria. If searching for data within a cube, Find creates a query that locates all categories matching all specified criteria.

A search does not include hidden categories inside a report.The Find Method is an optional component within the subset definition for AdvancedQuery.

You must create the FindQuery before adding it to the AdvancedQuery subset.

Note: The order of the components in the subset definition is important. First, specify the Dimension property, followed by the Level method. The Include, Exclude, and Find methods are optional and can appear in any order following the Dimension and Level properties. The Name property is required and can be set anywhere within the subset definition. Specify Execute, followed by the AddToReport method last.

Parameter	Description
FindSubsetName	Required. Specifies the name of the FindQuery subset to include in the AdvancedQuery query. Type: String

Return Type

String

Example

This example creates a FindQuery (type 1) subset definition that searches for all products that begin with the name "Star". Then an AdvancedQuery (type 3) subset is created using the FindQuery subset definition. The subset of Products that the query finds beginning with Star is then added to the report as columns.

```
Sub Main()
   Dim strCubePath As String
   Dim objPPRep As Object
   Dim objFind As Object
   Dim objAdvanced As Object
   strCubePath = "C:\Cubes and Reports\Great Outdoors.mdc"
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.New strCubePath, 1
   objPPRep.ExplorerMode = False
   objPPRep.Visible = True
   Set objFind = objPPRep.ReportQueries.Add(1)
   With objFind
      .Name = "Find Star"
      .Dimension = "Products"
      .SearchShortName = False
      .SearchText = "Star"
      .Pattern = 2
   End With
   Set objAdvanced = objPPRep.ReportQueries.Add(3)
   With objAdvanced
      .Name = "Star Products"
      .Dimension = "Products"
      .Level "Product Id"
      .Find objFind.Name
      .Execute
      .AddToReport 1,1,3
   End With
   Set objAdvanced = Nothing
   Set objFind = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "AdvancedQuery Object" on page 8

FindNext Method

Finds the next matching category label in a report.

Syntax

Report.**FindNext** (SearchText [, Pattern [, AllLayers [, Dimension]]])

Applies To

Report Object

Discussion

Use this method to locate categories within reports. Use the Execute method to locate categories within a cube.

FindNext looks for the next category in the report that matches the value prescribed by the parameters. If a match is found, True is returned; otherwise, False is returned.

The method's four parameters set the search options. Once the required parameter SearchText is given, the three optional parameters let you determine:

- if the search string is to be found anywhere in a word, at the start of a word only, or the end of a word only
- if pattern matching is set on or off
- if the search text is to match the whole word or any part of the word
- if the search is to be case sensitive
- if the current layer or all layers are to be searched
- if labels in rows, columns, layers or a combination of these are to be searched

If you search all three report dimensions (rows, columns, layers), the search progresses downward one layer at a time in the following order:

- rows in the current layer
- columns in the current layer
- the current layer label

The search ignores any hidden categories inside the report.

When a search returns True (that is, a match was found), the cursor is positioned on the matching label.

In nested crosstabs, FindNext searches for labels in each dimension, beginning at the current cursor position, and moves downward through the hierarchy to the lowest level.

When you set one of the three optional parameters, you must also set any optional parameters to its left, even if just the default is given. For example, to search just the rows dimension (value = 1) for the next label that contains "GO" in the current layer, your code looks like this:

```
FindNext("GO", 1, FALSE, 1)
```

Even though the second and third parameters use the defaults, they are needed as placeholders when changing the fourth. However, you can omit parameters if you need just the left-hand parameters and those to the right remain at their default settings.

To turn on more than one search option for the Pattern or Dimension parameters, add up the search option values. For example, to search just columns in the current layer for the next label beginning with "GO" with case sensitivity turned on, the code looks like this:

```
FindNext("GO", 34)
```

If you also want to search all rows, columns, and layers (Dimension = 7), the code looks like this:

```
FindNext("GO", 34, FALSE, 7)
```

Unless you specifically include a value for a parameter, the search option is not in effect. For example, to make a search case sensitive, you must include the value 32.

Adding values that cannot be used together causes an exception error. For example, don't combine the Contains, Begins With, or Ends With values since they are mutually exclusive. Do not use any of these with MatchWhole. Pattern Matching (Pattern = 8) cannot be used with MatchCase (Pattern = 16) or MatchWhole (Pattern = 32).

Examples of valid combinations include:

- MatchCase + Pattern Matching = (40)
- MatchCase + Contains = (33)
- MatchCase + Begins With = (34)
- MatchCase + Ends With = (36)
- MatchCase + MatchWhole = (48)

When the Pattern Matching value is used (Pattern = 8), the Find operation recognizes some characters in the text string as wildcards and some metacharacters are treated as reserved characters. If the metacharacters are included, an error message appears. If pattern matching is not used, wildcards and metacharacters are treated as normal characters.

Parameter	**Description**
SearchText	Required. Specifies the text to search for. Can contain wildcards if pattern matching (8) is turned on. An empty string is invalid. Type: String

Parameter	Description
Pattern	Optional. Specifies the search options. (There are certain reserved characters noted below.) Values are added to set option variations. 1 = Contains: the search text can be found anywhere in a word 2 = Begins With: the search text must be found at the start of a word 4 = Ends With: the search text must be found at the end of a word 8 = Pattern Matching: certain characters are treated as wildcards (these are listed in the table below) 16 = MatchWhole: the search text must match the whole word 32 = MatchCase: the search is case sensitive Default: 1 Type: Integer
AllLayers	Optional. Specifies whether or not all layers or just the current layer are searched. Valid options are True = all layers False = current layer Default: False Type: Boolean
Dimension	Optional. Specifies the scope of the search. 1 = row labels only 2 = column labels only 3 = row and column labels only 4 = layer labels only 5 = layer and row labels only 6 = layer and column labels only 7 = all labels Default: 3 Type: Integer

Return Type

Boolean

Example

This example searches for the next row beginning with "Star", if no row is found it tries to find the previous row containing "Star". A message is returned confirming whether a matching row was found.

```
Sub Main()
   Dim objPPRep As Object
   Dim intFound As Integer
   Const begins_with = 2
   Const current_layer = False
   Const rows = 1
```

```
   Set objPPRep = GetObject(, "CognosPowerPlay.Report")
   intFound = objPPRep.FindNext("Star", begins_with,
_
      current_layer, rows)
   If intFound = 0 Then
      intFound = objPPRep.FindPrevious("Star", _
         begins_with, current_layer, rows)
   End If
   If intFound <> 0 Then
      objPPRep.Rows.Active.Select
      MsgBox "A row was found and has been selected."
   Else
      MsgBox "No rows found matching criteria."
   End If
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "Execute Method" on page 143
- "Find Method" on page 146
- "FindPrevious Method"

FindPrevious Method

Finds the previous matching category label in a report.

Syntax

Report.**FindPrevious** (SearchText [, Pattern [, AllLayers [, Dimension]]])

Applies To

Report Object

Discussion

Use this method to locate categories within reports. Use the Execute method to locate categories within a cube.

FindPrevious performs a find previous or find up operation and looks for the previous category in the report that matches the value prescribed by the parameters. If a match is found, True is returned; otherwise, False is returned.

The method's four parameters set the search options. Once the required parameter SearchText is given, the three optional parameters let you determine:

- if the search string is to be found anywhere in a word, at the start of a word only, or the end of a word only
- if pattern matching is set on or off
- if the search text is to match the whole word or any part of the word
- if the search is to be case sensitive
- if the current layer or all layers are to be searched
- if labels in rows, columns, layers or a combination of these are to be searched

If you search all three report dimensions (rows, columns, layers), the search progresses upward one layer at a time in the following order:

- rows in the current layer
- columns in the current layer
- the current layer label

When a search returns True (that is, a match was found), the cursor is positioned on the matching label.

The search ignores any hidden categories inside the report.

In nested crosstabs, FindPrevious searches for labels in each dimension, beginning at the current cursor position, and moves upward through the hierarchy to the highest level.

When you set one of the three optional parameters, you must also set any optional parameters to its left, even if just the default is given. For example, to search just the rows dimension (value = 1) for the previous label which contains "GO" in the current layer, your code looks like this:

```
FindPrevious("GO", 1, FALSE, 1)
```

Even though the second and third parameters use the defaults, they are needed as placeholders when changing the fourth. However, you can omit parameters if you need just the left-hand parameters and those to the right remain at their default settings.

To turn on more than one search option for the Pattern or Dimension parameters, add up the search option values. For example, to search just columns in the current layer for the next label beginning with "GO" with case sensitivity turned on, the code looks like this:

```
FindPrevious("GO", 34)
```

If you also want to search all rows, columns, and layers (Dimension = 7), the code looks like this:

```
FindPrevious("GO", 34, FALSE, 7)
```

Unless you specifically include a value for a parameter, the search option is not in effect. For example, to make a search case sensitive, you must include the value 32.

Adding values that cannot be used together causes an exception error. For example, do not combine the Contains, Begins With, or Ends With values since they are mutually exclusive, and don't use any of these with MatchWhole. Pattern Matching (Pattern = 8) cannot be used with MatchCase (Pattern = 16) or MatchWhole (Pattern = 32).

Examples of valid combinations include:

- MatchCase + Pattern Matching = (40)
- MatchCase + Contains = (33)
- MatchCase + Begins With = (34)
- MatchCase + Ends With = (36)
- MatchCase + MatchWhole = (48)

When the Pattern Matching value is used (Pattern = 8), the Find operation recognizes some characters in the text string as wildcards and some metacharacters are treated as reserved characters when Pattern Matching is used. If the metacharacters are included, an error message appears. If pattern matching is not used, wildcards and metacharacters are treated as normal characters.

Parameter	Description
SearchText	Required. Specifies the text to search for. Can contain wildcards if pattern matching (8) is turned on. An empty string is invalid. Type: String
Pattern	Optional. Specifies the search options. (There are certain reserved characters noted below.) Values are added to set option variations. 1 = Contains: the search text can be found anywhere in a word. 2 = Begins With: the search text must be found at the start of a word 4 = Ends With: the search text must be found at the end of a word 8 = Pattern Matching: certain characters are treated as wildcards (these are listed in the table below) 16 = MatchWhole: the search text must match the whole word 32 = MatchCase: the search is case sensitive Default: 1 Type: Integer
AllLayers	Optional. Specifies whether or not all layers or just the current layer are searched. Valid options are True = all layers False = current layer Default: False Type: Boolean
Dimension	Optional. Specifies the scope of the search. 1 = row labels only 2 = column labels only 3 = row and column labels only 4 = layer labels only 5 = layer and row labels only 6 = layer and column labels only 7 = all labels Default: 3 Type: Integer

Return Type

Boolean

Example

This example searches for the next row beginning with "Star", if no row is found it tries to find the previous row containing "Star". A message is returned confirming whether a matching row was found.

```
Sub Main()
   Dim objPPRep As Object
   Dim intFound As Integer
   Const begins_with = 2
   Const current_layer = False
   Const rows = 1
   Set objPPRep = GetObject(, "CognosPowerPlay.Report")
   intFound = objPPRep.FindNext("Star", begins_with,
_
      current_layer, rows)
   If intFound = 0 Then
      intFound = objPPRep.FindPrevious("Star", _
         begins_with, current_layer, rows)
   End If
   If intFound <> 0 Then
      objPPRep.Rows.Active.Select
      MsgBox "A row was found and has been selected."
   Else
      MsgBox "No rows found matching criteria."
   End If
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "Execute Method" on page 143
- "Find Method" on page 146
- "FindNext Method" on page 148

Forecast Method (Explorer)

Creates a specified number of forecast categories based on the existing time dimensions.

Syntax

Report.**Forecast**

Applies To

Report Object

TERMS OF USE

The forecasting methods utilized in the Forecasting Function are based on the statistical analysis of historical information drawn from underlying data sources. The accuracy of the forecasted values is subject to many variables, including the accuracy of the underlying historical data and external events which could affect

the validity of that underlying historical data for forecasting purposes. The Forecasting Function is to be used only as a guide of the future values for the measures being forecasted and is not intended to be used as the basis for complex financial or business decisions.

IBM makes no representations as to the accuracy of the actual future values and does not guarantee any specific results. You use the Forecasting Function and the data it generates at your own risk. The Forecasting Function may contain errors or produce inaccurate calculations. You accept the Forecasting Function and the documentation "AS IS". IN NO EVENT SHALL IBM BE LIABLE FOR DAMAGES OF ANY KIND INCLUDING, WITHOUT LIMITATION, DIRECT, INDIRECT, INCIDENTAL, PUNITIVE OR CONSEQUENTIAL DAMAGES, RESULTING FROM THE USE OF THE FORECASTING FUNCTION OR THE INTERPRETATION OF THE DATA RESULTING THEREFROM.

Discussion

This method is available only if the Report object is in Explorer mode (the ReporterMode property is False).

You can use one of the following time series forecasting methods:

- The Trend forecasting method is based on the linear regression technique of time series forecasting. Trend forecasting gives the best forecasting reliability when the driving factors of your business affect your measures in a linear fashion. Use the Trend forecasting method when you have only two data values representing two time periods in your historic data.
- The Growth forecasting method is based on the exponential regression technique of time series forecasting. Growth forecasting gives you the best forecasting reliability when the driving factors of your business affect your measures exponentially.
- The Autoregression forecasting method is based on the auto-correlational approach to time series forecasting. Autoregression forecasting detects the linear, non-linear, and seasonal fluctuations in historical data and projects these trends into the future. Autoregression provides the best forecasting reliability when the driving factors underlying your business are affected by seasonal fluctuations. Use the Autoregression method when you have historic data representing a large number of time periods (for example, more than 24 monthly periods).

The Rows or Columns must contain time categories.

The forecast horizon is limited to the number of time categories in the report. For example, if there are two year categories in the report (2008, 1997), the maximum horizon value is two.

If the horizon value is zero, existing forecasts are removed from the report.

Parameter	Description
MethodValue	Required. Specifies the forecast method to be used. IBM Cognos PowerPlay provides the Trend, Growth, and Autoregression methods of forecasting. The forecast method is 1 for Trend, 2 for Growth, and 3 for Autoregression. **Type:** Short
HorizonValue	Required. Specifies the number of forecasts to be returned. The forecast horizon is limited to the number of time categories in the report. **Type:** Short

Return Type

Nothing

Example

This example returns an open report, confirms that the report is in Explorer mode, verifies that either the Column objects or the Row objects contain time categories, verifies that the horizon value is valid, and enters a forecast horizon value of two.

```
Sub Main()
   Dim objPPRep As Object
   Set objPPRep = CreateObject ("CognosPowerPlay.Report")
   objPPRep.Open ("C:\Great Outdoors.mdc")
   objPPRep.ExplorerMode = True
   objPPRep.Visible = True
   objPPRep.Forecast 1,2
   Set objPPRep = Nothing
End Sub
```

Related Topics

"Report Object" on page 37

GetDataNow Method

Updates the data in the Report object.

Syntax

Report.**GetDataNow**

Applies To

Report Object

Discussion

In a situation where data is constantly fluctuating, the data can be immediately updated using the GetDataNow Method. By using the Scheduler with a macro that uses GetDataNow to update reports, you can do so at 2:00AM when the network traffic is low.

Return Type

Nothing

Example

This example opens a report, and if the GetDataAutomatically property is set to False, the GetDataNow method updates the data in the report.

```
Sub Main()
   Dim objPPRep as Object
   Set objPPRep = CreateObject ("CognosPowerPlay.Report")
   objPPRep.Open "C:\Cubes and Reports\Sample1.ppx"
   objPPRep.Visible = True
   If objPPRep.GetDataAutomatically = 0 Then
      objPPRep.GetDataNow
   End if
   objPPRep.Close
   Set objPPrep = Nothing
End Sub
```

Related Topics

- "GetDataNow Method" on page 156
- "ActiveReport Method" on page 82
- "Report Object" on page 37

Graphs Method

Returns one Graph object or the entire collection.

Syntax

Report.**Graphs**

Applies To

Report Object

Discussion

If no index is specified, a Graphs collection object is returned, otherwise the method returns the requested Graph object.

Return Type

Object

Example

This example opens a report, and shows the display type for the first display in the report.

```
Sub Main()
   Dim objPPRep as Object
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.Open "C:\Cubes and Reports\Graph.ppx"
   MsgBox "The type of graph is" & _
      objPPRep.Graphs.Item(1).Type & "."
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "Graph Object" on page 25
- "Graphs" on page 56
- "ActiveReport Method" on page 82
- "Report Object" on page 37

HasParent Method

Returns whether the current category has a parent.

Syntax

Dimension.**HasParent**

Applies To

Dimension Object

Discussion

Use this method to determine if the current category has a parent category. Top level, alternate, and calculated categories do not have a parent. Use the Parent method to return the name of the parent category.

True indicates that there is a parent, and False indicates no parent.

Return Type

Boolean

Example

For the category that the first dimension is filtered on, this example determines if it has a parent category, whether it is visible, and displays the parent name.

```
Sub Main()
   Dim objPPRep As Object
   Dim objDimension As Object
   Dim strName As String
   Dim strVisible As String
   Set objPPRep = GetObject(,"CognosPowerPlay.Report")
```

```
   Set objDimension = objPPRep.DimensionLine.Item(1)
   objDimension.ChangeToTop
   If objDimension.HasParent = 0 Then
      Msgbox "The " & objDimension.Name & " dimension
has " & _
         "no parent."
      strName = objDimension.Children.Item(1).Name
      objDimension.Change strName
      If objDimension.Visible = -1 Then
         strVisible = "visible"
      Else
         strVisible = "not visible"
      End If
      MsgBox "The " & objDimension.Name & " dimension
is " & _
         strVisible & "." & chr$(10) & chr$(10)
& _
         "Its parent is " & objDimension.Parent &
"."
   End If
   Set objDimension = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "Dimension Object" on page 19
- "ChangeToParent Method" on page 116

Hide Method

Hides the category.

Syntax

object.**Hide**

Applies To

Column Object

Columns

Row Object

Rows

Discussion

Use this method to hide a single category or a collection of categories in a Report object. To reveal them, use the UnhideAllCategories method on the Report object. References to the position of an object in the collection are not valid after you use this method.

You cannot hide summary categories.

Return Type

Nothing

Example

This example opens a report and hides the column "Tents".

```
Sub Main()
   Dim objPPRep as Object
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.Open "C:\Cubes and Reports\Sample1.ppx"
   objPPRep.Columns.Item("Tents").Hide
   objPPRep.Save
   objPPRep.Close
End Sub
```

Related Topics

- "Column Object" on page 15
- "Columns" on page 52
- "Report Object" on page 37
- "Row Object" on page 41
- "Rows" on page 66
- "UnhideAllCategories Method" on page 250

HideSelected Method

Hides selected objects in a collection.

Syntax

Report.**HideSelected**

Applies To

Report Object

Discussion

Use this method in conjunction with the Select method. Once objects are selected in a collection using Select, you can hide them using HideSelected. To reveal all hidden objects, use the Unhide method. It returns True if it is successful; otherwise, it returns False if an error occurs.

Return Type

Boolean

Example

This example searches for and hides all rows that begin with "Star."

```
Sub Main()
```

```
   Dim objPPRep As Object
   Dim intFound As Integer
   Const begins_with = 2
   Const current_layer = False
   Const rows = 1
   Set objPPRep = GetObject(, "CognosPowerPlay.Report")
   objPPRep.Rows.Item(1).Activate
   intFound = 1
   Do While intFound <> 0
      intFound = objPPRep.FindNext("Star", begins_with,
_
         current_layer, rows)
      If intFound = -1 Then
         objPPRep.Rows.Active.Select
         objPPRep.HideSelected
      End If
   Loop
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "HideUnselected Method"
- "Select Method" on page 220

HideUnselected Method

Hides any object in a collection that is not selected.

Syntax

Report.**HideUnSelected**

Applies To

Report Object

Discussion

Use this method in conjunction with the Select method. Once objects are selected in a collection using Select, you can use HideUnselected to hide all objects not currently selected. It returns True if it is successful; otherwise, it returns False if an error occurs.

If you use HideUnselected without first using Select, you will hide everything.

Return Type

Boolean

Example

This example selects the first row which starts with 'Star' and hides all other rows.

```
Sub Main()
```

```
   Dim objPPRep As Object
   Dim intFound As Integer
   Const begins_with = 2
   Const current_layer = False
   Const rows = 1
   Set objPPRep = GetObject(, "CognosPowerPlay.Report")
   objPPRep.Rows.Item(1).Activate
   intFound = 1
   intFound = objPPRep.FindNext("Star", begins_with,
_
         current_layer, rows)
   If intFound = -1 Then
      objPPRep.Rows.Active.Select
      objPPRep.HideUnselected
   Else
      MsgBox "No rows found."
   End If
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "HideSelected Method" on page 160
- "Select Method" on page 220

Include Method

Sets the categories to include in the query.

Syntax

AdvancedQuery.**Include** CategoryName [,ParentName]

Applies To

AdvancedQuery Object

Discussion

Use this method to identify the categories to include in the subset definition for the AdvancedQuery. To include more than one category in the query, specify multiple Include statements in the subset definition for an AdvancedQuery.

If the resulting query finds more than one category that matches the specified label, specify the optional parameter, which can be the parent category or drill-down path of the desired category. The functionality is added to identify the category. For example

```
Include("CategoryName", "ParentName or drill-down path
name")
```

Note: The order of the components in the subset definition is important. First, specify the Dimension property, followed by the Level method. The Include, Exclude, and Find methods are optional and can appear in any order following the

Dimension and Level properties. The Name property is required and can be set anywhere within the subset definition. Specify Execute, followed by the AddToReport method last.

Parameter	Description
CategoryName	Required. Specifies the name of the category to include in the subset. Type: String
ParentName	Optional. Specifies the name of the ancestor category of the category to include in the subset. For example, two countries or regions may have the same city name. Use the country or region name (the ancestor) to differentiate between the two cities. Type: String

Return Type

Nothing

Example

This example creates an AdvancedQuery (type 3) subset definition that retrieves all categories belonging to Europe. The resulting subset is then added to the report as rows.

```
Sub Main()
   Dim strCubePath As String
   Dim objPPRep As Object
   Dim objAdvanced As Object
   strCubePath = "C:\Cubes and Reports\Great Outdoors.mdc"
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.New strCubePath, 1
   objPPRep.ExplorerMode = False
   objPPRep.Visible = True
   Set objAdvanced = objPPRep.ReportQueries.Add(3)
   With objAdvanced
      .Name = "European Countries"
      .Dimension = "Locations"
      .Level "Country or Region"
      .Include "Europe"
      .Execute
      .AddToReport 0,1,3
   End With
   Msgbox "Name:  " & objAdvanced.Name & chr$(10)
& _
      "Dimension:  " & objAdvanced.Dimension &
chr$(10) & _
      "Level List:  " & objAdvanced.LevelList &
chr$(10) & _
```

```
      "Query Type Code:  " & objAdvanced.Type &
chr$(10) & _
      "Number of Categories:  " & objAdvanced.Count
& _
      chr$(10) & _
      "First Category:  " & objAdvanced.Item(1).Name,
_
      ,"Subset"
   Set objAdvanced = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "AdvancedQuery Object" on page 8
- "Exclude Method" on page 141

Item Method

Returns an object from the collection or the object that maintains a list of other objects.

Syntax

object.**Item**(Index)

Applies To

AdvancedQuery Object

Children

Columns

DimensionLine Object

Exceptions

FindQuery Object

Graphs

Layers

ParentageQuery Object

Ranges

Reports

ReportQueries

Rows

Discussion

Since the Item method is the default method for all collections except Exceptions and Graphs, only the index parameter is required and not the method name. If Item is the default method, the following are equivalent:

```
Collection.Item(3)
Collection(3)
```

For Exceptions and Graphs collections and DimensionLine object, you must use a syntax similar to the following to select an object from them:

```
Set objPPRange = objPPRep.Exceptions.Item(1).Range
```

Parameters	Description
Index	Required. The category label or index of the object to return. For the Graphs collection, only the category index is available. Type: String or Long

Return Type

Object

Example

This example adds column one and column four. In this case, the Item method is used to select the specific rows and columns.

```
Sub Main()
   Dim objPPRep As Object
   Dim objNewCol As Object
   Dim objNewRow As Object
   Set objPPRep = GetObject( , "CognosPowerPlay.Report")
   Set objNewRow = objPPRep.Rows.Subset(1, 3).Addition
   MsgBox "The sum of the first three rows is " _
      &objPPRep.CellValue(objNewRow(1).Index,1)
   Set objNewCol = objPPRep.Columns.Item(4).Addition
_
      (objPPRep.Columns.Item(1))
   MsgBox " The sum of column one and column four is
" _
      &objPPRep.CellValue(1,objNewCol.Index)
   objPPRep.Save
   Set objNewRow = Nothing
   Set objNewCol = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- Chapter 4, “Methods,” on page 71
- Chapter 5, “Properties,” on page 259

ItemAtLevel Method

Returns either a row or column object from a nested report.

Syntax

collection.**ItemAtLevel**(Label or Index, Level)

Applies To

Columns

Rows

Discussion

Use this method to retrieve a row or column from a Rows or Columns collection in a nested crosstab report. The Item method can be used for non-nested reports.

The item to search for is set using the Label parameter. The level where the item resides is set using the Level parameter, where 0 indicates the level closest to the actual data --that is, the lowest level row or column--and 1, means the next level up, and so on. For example, where a crosstab has levels for Years and Months and then data, level 0 is Months and level 1 is Years.

The row or column object returned could span multiple indexes when the level is > 0.

ItemAtLevel is equivalent to Item when level = 0.

Parameters	Description
Label	Required. Specifies the category label or index of the object to return. Type: Variant
Level	Required. Specifies the value indicating the nesting level in which to add the categories. Type: Long

Return Type

Object

Example

This example selects the last column in the report, applies a style to it and changes the cell size.

```
Sub Main()
   Dim objPPRep As Object
   Dim intCount As Integer
   Set objPPRep = GetObject(,"CognosPowerPlay.Report")
```

```
   intCount = objPPRep.Columns.Count
   objPPRep.Columns.ItemAtLevel(intCount,0).Select
   objPPRep.StyleSelected "Good News"
   objPPRep.SizeSelected 100
   Set objPPRep = Nothing
End Sub
```

Related Topics

- Chapter 4, "Methods," on page 71
- Chapter 5, "Properties," on page 259

Layers Method

Returns one Layer object or the entire collection.

Syntax

Report.**Layers**

Applies To

Report Object

Discussion

A layer is a set of dimension categories that provide details for another dimension and a new perspective on the report results. A report can contain several layers.

If no index is specified, a Layers collection is returned. Otherwise the method returns the requested Layer object.

Return Type

Object

Example

This example removes the first two layers in a report, shows the sum of the two new first layers in a new layer and shows the maximum of all the layers in a new layer.

```
Sub Main()
   Dim objPPApp as Object
   Dim objPPRep as Object
   Set objPPRep = GetObject ("C:\Cubes and Reports\Sample1.ppx")
   Set objPPApp = objPPRep.Application
   objPPApp.Visible = True
   objPPApp.Reports.Item(1).ExplorerMode = False
   objPPApp.Reports.Item(1).Layers.Subset(1,2).Remove
   objPPApp.Reports.Item(1).Layers.Subset(1,2).Addition
   objPPApp.Reports.Item(1).Layers.Maximum
   Set objPPRep = Nothing
   Set objPPApp = Nothing
```

```
End Sub
```

Related Topics

- "Layer Object" on page 28
- "Layers" on page 59
- "ActiveReport Method" on page 82
- "Report Object" on page 37

Level Method

Sets the level used by the AdvancedQuery object to retrieve categories for the query.

Syntax

AdvancedQuery.**Level LevelName** [,TopCategory]

Applies To

AdvancedQuery Object

Discussion

Use this method to specify one or more levels to identify the categories to be retrieved by the query. The query retrieves all categories for the levels specified in the subset definition. To include more than one Level in the query, specify multiple Level statements in the subset definition for AdvancedQuery. Specify at least one Level to include in the subset. When specifying more than one level, you must use Levels from the same dimension.

Use the LevelList property to return the list of included levels.

Note: The order of the components in the subset definition is important. First, specify the Dimension property, followed by the Level method. The Include, Exclude, and Find methods are optional and can appear in any order following the Dimension and Level properties. The Name property is required and can be set anywhere within the subset definition. Specify Execute, followed by the AddToReport method last.

Parameter	Description
LevelName	Required. Specifies the name of the level that contains the category used for the search. Type: String
TopCategory	Optional. Specifies the name of the top-level category for the specified level. Used for selecting levels from alternate drill-down paths. If this parameter is not specified, the level is assumed to be the primary drill-down path. Type: Variant

Return Type

Nothing

Example

This example creates an AdvancedQuery (type 3) subset definition that retrieves all categories belonging to Europe. The resulting subset is then added to the report as rows.

```
Sub Main()
   Dim strCubePath As String
   Dim objPPRep As Object
   Dim objFind As Object
   Dim objAdvanced As Object
   strCubePath = "C:\Cubes and Reports\Great Outdoors.mdc"
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.New strCubePath, 1
   objPPRep.ExplorerMode = False
   objPPRep.Visible = True
   Set objFind = objPPRep.ReportQueries.Add(1)
   With objFind
      .Name = "Find Star"
      .Dimension = "Products"
      .SearchShortName = False
      .SearchText = "Star"
      .Pattern = 2
   End With
   Set objAdvanced = objPPRep.ReportQueries.Add(3)
   With objAdvanced
      .Name = "Star Products"
      .Dimension = "Products"
      .Level "Product Id"
      .Find objFind.Name
      .Execute
      .AddToReport 1,1,3
   End With
   Set objAdvanced = Nothing
   Set objFind = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "AdvancedQuery Object" on page 8
- "LevelList Property" on page 333

Levels Method

Returns all levels available in the dimension for a category.

Syntax

Dimension.**Levels**([AlternateDrillDown])

Applies To

Dimension Object

Discussion

Use this method to return a collection of Level objects for a dimension. To return the collection of Level objects in the primary drill-down path, call this method with no parameter. To return the collection of Level objects in the alternate drill-down path, call this method and specify the AlternateDrillDown parameter with the name of an alternate drill-down path. The objects this method returns represent all the levels available in that dimension, starting from the top-level category. For an alternate drill-down path, this method returns all levels starting at this path.

This method fails when

- the specified drill-down path is not in the dimension or is misspelled
- no levels are found for the dimension or specified drill-down path
- the drill-down path occurred more than once in the dimension
- the drill-down path specified was not an alternate drill-down path

Parameter	Description
AlternateDrillDown	Optional. Specifies the name of an alternate drill-down path. All levels starting at the alternate drill-down path are returned. This parameter is not case-sensitive. Type: String

Return Type

Object

Example

This example returns a list of Level objects for the dimension. The category is being filtered for the dimension line index.

```
Sub Main()
   Dim objPPRep As Object
   Dim objDimension As Object
   Dim objLevel As Object
   Dim  strLevelList As String
   Dim intx As Integer
   Set objPPRep = GetObject(,"CognosPowerPlay.Report")
   Set objDimension = objPPRep.DimensionLine.Item(1)
   For intx = 1 to objDimension.Levels.Count
      Set objLevel = objDimension.Levels.Item(intx)
```

```
      strLevelList = strLevelList & chr$(10) &
objLevel.Name
      Set objLevel = Nothing
   Next intx
   MsgBox "The levels in the " & objDimension.Name
& _
         " dimension are:" & chr$(10) & strLevelList
   Set objDimension = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- Chapter 4, "Methods," on page 71
- Chapter 5, "Properties," on page 259

Logon Method

Logs on to IBM Cognos Analytics as an authenticated user to provide the application object access to secured IBM Cognos Analytics resources such as packages.

Syntax

*object***Logon** Namespace, UserID, Password

Applies To

Application Object

Discussion

You can repeat the logon method to log on to multiple namespaces. When you use this technique access permissions are a union of the permissions granted for each log on.

When you use the Logon method on an Application object, the method closes all reports using a remote package. If the application is visible and the reports that are using a remote package have been modified, you are prompted to save before closing. If the application is invisible, you are not prompted to save before closing.

If there is only one IBM Cognos Analytics namespace, you can use an empty string for Namespace. For example, Logon "", "username", "password"

When single sign-on is configured in an IBM Cognos Analytics namespace, only the namespace parameter is required. For example, Logon "namespace"

Parameter	Description
Namespace	Required. Specifies the IBM Cognos Analytics namespace to log on to. Type: String

Parameter	Description
UserID	Optional. Specifies the user ID credential for the IBM Cognos Analytics namespace. Type: String
Password	Optional. Specifies the password credential for the user ID. Type: String

Return Type

Nothing

Example

This example logs into an IBM Cognos Analytics namespace, opens a report using a remote secured package, and then logs off.

```
Sub Main()
   Dim objPPRep As Object
   Dim objPPApp As Object
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   Set objPPApp = objPPRep.Application
   objPPApp.Logon "Cognos", "UserA", "PwdA"
   objPPRep.Open "D:\PPlay1.ppx"
   objPPApp.Logoff
   Set objPPRep = Nothing
   Set objPPApp = Nothing
End Sub
```

Related Topics

- Chapter 4, “Methods,” on page 71

Logoff Method

Revokes authentication to all IBM Cognos Analytics namespaces for the application object. Even if you used multiple namespaces in the session, you logoff only once.

Syntax

object.**Logoff**

Applies To

Application Object

Discussion

When you use the Logoff method on an application object, the method closes all reports using a remote package. If the application is visible and the reports that are using a remote package have been modified, you are prompted to save the reports.

If the application is invisible, you are not prompted to save any modified reports.

Return Type

Nothing

Example

This example logs into an IBM Cognos Analytics namespace, opens a report using a remote secured package, and then logs off.

```
Sub Main()
   Dim objPPRep As Object
   Dim objPPApp As Object
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   Set objPPApp = objPPRep.Application
   objPPApp.Logon "Cognos", "UserA", "PwdA"
   objPPRep.Open "D:\PPlay1.ppx"
   objPPApp.Logoff
   Set objPPRep = Nothing
   Set objPPApp = Nothing
End Sub
```

Related Topics

- Chapter 4, “Methods,” on page 71

Maximize Method

Maximizes the object window.

Syntax

object.**Maximize**

Applies To

Application Object

Report Object

Discussion

This method cannot reveal any hidden windows. To do so, use the Visible attribute of either the Application or Report object.

Return Type

Nothing

Example

This example simply finds a visible instance of IBM Cognos PowerPlay and maximizes it.

```
Sub Main()
```

```
   Dim objPPApp as Object
   Set objPPApp = GetObject( ,"CognosPowerPlay.Report")
   objPPApp.Visible = True
   objPPAPP.Activate
   objPPApp.Maximize
   Set objPPApp = Nothing
End Sub
```

Related Topics

- "Application Object" on page 11
- "ActiveReport Method" on page 82
- "Report Object" on page 37

Maximum Method (Collections) (Reporter)

Determines the maximum between either a constant value or a category, and one or more categories.

Syntax

collection.**Maximum**[(Operand)]

Applies To

Columns

Layers

Rows

Discussion

This method can also determine the maximum of multiple categories.

This method is only available if the Report object is in Reporter Mode (ExplorerMode property set to False).

For Column, Layer, and Row objects, the method determines a maximum for each category and operand pair, and creates a new category for each result.

Depending on whether the method was applied to an object or a collection, the results are returned respectively as an object or a collection. The new calculation is inserted directly after the active row or column.

References to the position of an object in the collection are not valid after you use this method.

Parameters	Description
Operand	Optional. Specifies either a constant value or a category object. If you do not specify this parameter, the method creates a category to display the maximum of all the categories in the collection. If you specify this parameter, this method determines a maximum from each category and creates a new category for each result. Type: Variant

Return Type

Object

Example

This example determines the maximum of the first three rows and of column one and four.

```
Sub Main()
   Dim objPPRep As Object
   Dim objNewCol As Object
   Dim objNewRow As Object
   Set objPPRep = GetObject( , "CognosPowerPlay.Report")
   objPPRep.ExplorerMode = False
   Set objNewRow = objPPRep.Rows.Subset(1, 3).Maximum
   MsgBox "The maximum of the first three rows is " _
      &objPPRep.CellValue(objNewRow(1).Index,1)
   Set objNewCol = objPPRep.Columns.Item(4).Maximum _
      (objPPRep.Columns.Item(1))
   MsgBox " The maximum of column one and column four
is " _
      &objPPRep.CellValue(1,objNewCol.Index)
   objPPRep.Save
   Set objNewRow = Nothing
   Set objNewCol = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "Column Object" on page 15
- "Columns" on page 52
- "Layer Object" on page 28
- "Layers" on page 59
- "Row Object" on page 41
- "Rows" on page 66

Maximum Method (Objects) (Reporter)

Determines the maximum between either a constant value or a category, and one or more categories.

Syntax

*object***Maximum**(Operand)

Applies To

Column Object

Layer Object

Row Object

Discussion

For Column, Layer, and Row objects, the method determines a maximum for each category and operand pair, and creates a new category for each result. This method can also determine the maximum of multiple categories.

Depending on whether the method was applied to an object or a collection, the results are returned respectively as an object or a collection. The new calculation is inserted directly after the active row or column.

This method is only available if the Report object is in Reporter Mode (the ExplorerMode property is False).

References to the position of an object in the collection are not valid after you use this method.

Parameters	**Description**
Operand	Required. Specifies either a constant value or a category object. If you do not specify this parameter, this method creates a category to display the maximum of all the categories in the collection. If you specify this parameter, this method determines a maximum from each category and creates a new category for each result. Type: Variant

Return Type

Object

Example

This example compares a constant value with the values in a column and returns the maximum of the two values in a new column.

```
Sub Main()
```

```
   Dim objPPRep as Object
   Dim objPPCol as Object
   Set objPPRep = CreateObject ("CognosPowerPlay.Report")
   objPPRep.Open "C:\Cubes and Reports\Sample1.ppx"
   objPPRep.ExplorerMode = False
   Set objPPCol = objPPRep.Columns.Item("Tents")
   objPPCol.Maximum(80000)
   objPPRep.Save
   Set objPPCol = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "Column Object" on page 15
- "Columns" on page 52
- "Layer Object" on page 28
- "Layers" on page 59
- "Row Object" on page 41
- "Rows" on page 66

Minimize Method

Minimizes the object window.

Syntax

object.**Minimize**

Applies To

Application Object

Report Object

Discussion

This method does not reveal any hidden windows. To do so, use the Visible attribute of either the Application or Report object.

Return Type

Nothing

Example

This example finds a visible instance of IBM Cognos PowerPlay and minimizes it.

```
Sub Main()
   Dim objPPApp as Object
   Set objPPApp = GetObject( ,"CognosPowerPlay.Report")
   objPPApp.Visible = True
   objPPAPP.Activate
   objPPApp.Minimize
```

```
   Set objPPApp = Nothing
End Sub
```

Related Topics

- "Application Object" on page 11
- "ActiveReport Method" on page 82
- "Report Object" on page 37

Minimum Method (Collections) (Reporter)

Determines the minimum between either a constant value or a category, and one or more categories.

Syntax

collection.**Minimum**[(Operand)]

Applies To

Columns

Layers

Rows

Discussion

This method can also determine the minimum of multiple categories.

This method is only available if the Report object is in Reporter Mode (the ExplorerMode property is False).

Depending on whether the method was applied to an object or a collection, the results are returned respectively as an object or a collection. The new calculation is inserted directly after the active row or column.

References to the position of an object in the collection are not valid after you use this method.

Parameters	Description
Operand	Optional. Specifies either a constant value or a category object. If you do not specify this parameter, this method creates a category to display the minimum of all the categories in the collection. If you specify this parameter, the method determines a minimum from each category and creates a new category for each result. Type: Variant

Return Type

Object

Example

This example determines the maximum of the first three rows and of column one and four.

```
Sub Main()
   Dim objPPRep As Object
   Dim objNewCol As Object
   Dim objNewRow As Object
   Set objPPRep = GetObject( , "CognosPowerPlay.Report")
   objPPRep.ExplorerMode = False
   Set objNewRow = objPPRep.Rows.SubSet(1,3).Minimum
   MsgBox "The minimum of the first three rows is " _
      &objPPRep.CellValue(objNewRow(1).Index,1)
   Set objNewCol = objPPRep.Columns.Item(4).Minimum _
      (objPPRep.Columns.Item(1))
   MsgBox "The minimum of columns 1 and 4 is " _
      &objPPRep.CellValue(1,objNewCol.Index)
   Set objNewRow = Nothing
   Set objNewCol = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "Column Object" on page 15
- "Layer Object" on page 28
- "Report Object" on page 37
- "Row Object" on page 41

Minimum Method (Objects) (Reporter)

Determines the minimum between either a constant value or a category, and one or more categories.

Syntax

object.**Minimum**(Operand)

Applies To

Column Object

Layer Object

Discussion

For Column, Layer, and Row objects, the method determines a minimum for each category and operand pair and creates a new category for each result.

This method can also determine the minimum of multiple categories.

This method is only available if the Report object is in Reporter Mode (the ExplorerMode property is False).

Depending on whether the method was applied to an object or a collection, the results are returned respectively as an object or a collection. The new calculation is inserted directly after the active row or column.

References to the position of an object in the collection are not valid after you use this method.

Parameters	Description
Operand	Required. Specifies either a constant value or a category object. Type: Variant

Return Type

Object

Example

This example compares a constant value with the values in a column and returns the minimum of the two values in a new column.

```
Sub Main()
   Dim objPPRep as Object
   Dim objPPCol as Object
   Set objPPRep = CreateObject ("CognosPowerPlay.Report")
   objPPRep.Open "C:\Cubes and Reports\Sample1.ppx"
   objPPRep.ExplorerMode = False
   Set objPPCol = objPPRep.Columns.Item("Tents")
   objPPCol.Minimum(60000)
   objPPRep.Save
   Set objPPCol = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- “Column Object” on page 15
- “Layer Object” on page 28
- “Report Object” on page 37
- “Row Object” on page 41

Multiplication Method (Collections)

Multiplies a constant value or a category by one or more categories.

Syntax

collection.**Multiplication**[(Operand)]

Applies To

Columns

Layers

Rows

Discussion

This method can also multiply multiple categories together.

Depending on whether the method was applied to an object or a collection, the results are returned respectively as an object or a collection.

In Explorer mode, the new calculation is inserted directly after the last operand. In Reporter mode, the new calculation is inserted directly after the active row or column.

References to the position of an object in the collection are not valid after you use this method.

In Explorer mode, if you change a report by removing a level, drilling, filtering or nesting, then all calculations that can not be created in the changed report disappear.

Parameters	Description
Operand	Optional. Specifies either a constant value or a category object. If you do not specify this parameter, this method creates a category to display the results of the multiplication of all the categories in the collection. If you specify this parameter, this method determines a result from each category and a new category is created for each result. Type: Variant

Return Type

Object

Example

This example multiples the first five rows by a constant value and multiplies column one by column four.

```
Sub Main()
   Dim objPPRep As Object
   Dim objRslt As Object
   Set objPPRep = GetObject( , "CognosPowerPlay.Report")
```

```
   objPPRep.Rows.SubSet(1,5).Multiplication 6
   Set objRslt = objPPRep.Rows.Item(1).Multiplication
_
      (objPPRep.Rows.Item(4))
   MsgBox "The result of multiplying column 1 by column
4 is " _
      &objPPRep.CellValue(objRslt,1)
   Set objRslt = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "Column Object" on page 15
- "Layer Object" on page 28
- "Report Object" on page 37
- "Row Object" on page 41

Multiplication Method (Objects)

Multiplies a constant value or a category by an object.

Syntax

object **.Multiplication**(Operand)

Applies To

Column Object

Layer Object

Row Object

Discussion

For Column, Layer, and Row objects, the method determines a result for each category and operand pair, and creates a new category for each result.

Depending on whether the method was applied to an object or a collection, the results are returned respectively as an object or a collection.

In Explorer mode, the new calculation is inserted directly after the last operand. In Reporter mode, the new calculation is inserted directly after the active row or column.

References to the position of an object in the collection are not valid after you use this method.

In Explorer mode, if you change a report by removing a level, drilling, filtering or nesting, then all calculations that can not be created in the changed report disappear.

Parameters	Description
Operand	Required. Specifies either a constant value or a category object. Type: Variant

Return Type

Object

Example

This example multiplies the values in a column by 6 and returns the product in a new column.

```
Sub Main()
   Dim objPPRep as Object
   Dim objPPCol as Object
   Set objPPRep = CreateObject ("CognosPowerPlay.Report")
   objPPRep.Open "DC:\Cubes and Reports\Sample1.ppx"
   Set objPPCol = objPPRep.Columns.Item("Tents")
   objPPCol.Multiplication(6)
   objPPRep.Save
   Set objPPCol = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "Column Object" on page 15
- "Layer Object" on page 28
- "Report Object" on page 37
- "Row Object" on page 41

New Method

Creates a new Report object.

Syntax

Report.**New** ConnectionType, Location [, Timeout [, ExplorerMode]]

Applies To

Report Object

Discussion

Use this method to fully initialize a new Report object that was created using CreateObject("CognosPowerPlay.Report").

When using the CreateObject method to create a new report, use the New method to complete the initialization process.

In order to avoid this two-step approach, use the GetObject method to pass the MDC file name as the only parameter. For example, GetObject("c:\cognos\samples\outdoors.mdc").

Use the ConnectionType and Location parameters to create a report based on a local cube or remote package. For remote connections the IBM Cognos Analytics server must be running.

Parameters	Description
ConnectionType	Required. Identifies whether the connection type is for a local cube or remote package. This parameter can be either "local" or "remote". Type: String
Location	Required. If the connection type is local, then a fully qualified local cube is expected. Example, "C:\Cubes\Great Outdoors.mdc" If the connection type is remote, then a package search path in native encoding or a store ID is expected. Search path example, "/content/package[@name=Great Outdoors]" Store ID example, "storeID('iAA1ECBF2EA9B46F78651D4787F219509')" Type: String
Timeout	Optional. Is the amount of time to allow users to attempt to connect to the specified IBM Cognos Analytics server, for example 45. The default is 60 seconds. Type: Integer
ExplorerMode	Optional. Specifies whether Explorer or Reporter mode is use for the document. False (0) = Reporter True (-1) = Explorer Type: Boolean

Return Type

Nothing

Example

This example adds a new level of categories as layers to new report.

```
Sub Main()
   Dim objCubeCategories As Object
   Dim objPPRep As Object
   Const level_0 = 0
   Const level_1 = 1
   Const add_to_current = 0
   Const add_to_all = 1
   Const as_parent = 0
   Const as_child = 1
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.New "local", "C:\Cubes and Reports\Great
Outdoors.mdc"
   objPPRep.ExplorerMode = False
   objPPRep.Visible = True
   Set objCubeCategories = objPPRep.CategoryList
   objCubeCategories.Add level_1, "Locations"
   objPPRep.Layers.Add objCubeCategories
   Set objCubeCategories = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "ActiveReport Method" on page 82
- "Report Object" on page 37

Open Method (Reports)

Opens a collection of Report objects.

Syntax

Reports.**Open**(Report)

Applies To

Reports

Discussion

When using CreateObject to open an existing report, use the Open method to complete the initialization process.

To avoid this two-step approach, you can use GetObject and include the report file name and extension as the only parameter. For example, GetObject("C:\Cognos\Sample.ppx").

You cannot open a report generated in Extensible Markup Language (.xml) that is not a IBM Cognos PowerPlay portable report (.ppx).

When you use the Open method for the Reports collection, you must capture the new Report object or it will terminate. For example, you must use the following to capture the report opened using the object, PPRep1:

```
Set objPPRep1 = objPPApp.Reports.Open("c:\Cognos\Outdoors.ppx")
```

Using

```
objPPApp.Reports.Open("c:\Cognos\Outdoors.ppx")
```

will terminate the report.

Parameters	Description
Report	Required. Specifies the PowerPlay report to open. Type: String

Return Type

Object

Example

This example opens all PowerPlay reports in the specified directory and converts them to use a remote cube on the connection "Great Outdoors".

```
Sub Main()
   Dim objPPRep As Object
   Dim strPath As String
   Dim strExtension As String
   Dim strFolder As String
   strPath = "C:\Reports\"
   strExtension = "*.ppx"
   strFolder=Dir$(strPath & strExtension,16)
   If strFolder <> " Then
     Do Until strFolder = "
        Set objPPRep = CreateObject("CognosPowerPlay.Report")
        objPPRep.Open strPath & strFolder, "Great
Outdoors.mdc"
        objPPRep.Save
        objPPRep.Close
        Set objPPRep = Nothing
        strFolder = Dir
     Loop
   End If
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "ActiveReport Method" on page 82
- "Reports" on page 65

Open Method (Report)

Opens an existing Report object.

Syntax

Report.**Open**(Report [,ConnectionType, Location[,Timeout]])

Applies To

Report Object

Discussion

When using CreateObject to open an existing report, use the Open method to complete the initialization process.

To avoid this two-step approach, you can use GetObject and include the report file name and extension as the only parameter. For example, GetObject("C:\Cognos\Sample.ppx").

You cannot open a report generated in Extensible Markup Language (.xml) that is not a IBM Cognos PowerPlay portable report (.ppx).

When you use the Open method for the Reports collection, you must capture the new Report object or it will terminate. For example, you must use the following to capture the report opened using the object, PPRep1:

```
Set objPPRep1 = objPPApp.Reports.Open("c:\Cognos\Outdoors.ppx")
```

Using

```
objPPApp.Reports.Open("c:\Cognos\Outdoors.ppx")
```

will terminate the report.

The ConnectionType, Location, and Timeout parameters are commonly used to override the reference to the local cube which was used when the report was created. When using this method without specifying the optional parameters, PowerPlay will look for the report in the following places (in order)

- user defined path
- current directory
- report path
- preference path (if it exists)
- directory of the executable
- Windows system directory
- Windows directory
- paths listed in the path environment variable
- relative path across the local drives and then the connected drives

Parameters	Description
Report	Required. Specifies the PowerPlay report to open. Type: String

Parameters	Description
ConnectionType	Required. Identifies whether the connection type is for a local cube or remote package. This parameter can be either "local" or "remote". Type: String
Location	Required. If the connection type is local, then a fully qualified local cube is expected. Example, "C:\Cubes\Great Outdoors.mdc" If the connection type is remote, then a package search path in native encoding or a store ID is expected. Search path example, "/content/package[@name=Great Outdoors]" Store ID example, "storeID('iAA1ECBF2EA9B46F78651D4787F219509')" Type: String
Timeout	Optional. Is the amount of time to allow users to attempt to connect to the specified server, for example 45. The default is 60 seconds. Type: Integer

Return Type

Nothing

Example

This example opens a report, makes it visible and adds a stacked bar graph.

```
Sub Main()
   Dim objPPRep as Object
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.Open "c:\Cognos\Outdoors.ppx", "remote",
       "storeID('iAA1ECBF2EA9B46F78651D4787F219509')"
   objPPRep.Visible
   objPPRep.Graphs.Add 5
   MsgBox "The active graph type is " _
      &objPPRep.Graphs.Active.Type
   objPPRep.Save
   objPPRep.Close
   Set objPPRep = Nothing
End Sub
```

Related Topics

- Chapter 4, “Methods,” on page 71
- Chapter 5, “Properties,” on page 259

OpenRemoteReport Method

Opens an existing remote report object.

Syntax

Report.**OpenRemoteReport**(Report [,ConnectionType, Location[,Timeout]])

Applies To

Report Object

Discussion

When using CreateObject to open an existing report, use the OpenRemoteReport method to complete the initialization process.

The ConnectionType, Location, and Timeout parameters are commonly used to override the reference to the remote package which was used when the report was created.

Parameters	Description
Report	Required. Specifies the remote PowerPlay report to open. A package search path in native encoding or a store ID is expected. Search path example, "/content/package[@name=Great Outdoors]" Store ID example, "storeID('iAA1ECBF2EA9B46F78651D4787F219509')" Type: String
ConnectionType	Required. Identifies whether the connection type is for a local cube or remote package. This parameter can be either "local" or "remote". Type: String
Location	Required. If the connection type is local, then a fully qualified local cube is expected. Example, "C:\Cubes\Great Outdoors.mdc" If the connection type is remote, then a package search path in native encoding or a store ID is expected. Search path example, "/content/package[@name=Great Outdoors]" Store ID example, "storeID('iAA1ECBF2EA9B46F78651D4787F219509')" Type: String

Parameters	Description
Timeout	Optional. Is the amount of time to allow users to attempt to connect to the specified server, for example 45. The default is 60 seconds. Type: Integer

Return Type

Nothing

Example

This example opens a report, makes it visible and adds a stacked bar graph.

```
Sub Main()
   Dim objPPRep as Object
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.OpenRemoteReport "storeID('iAA1ECBF2EA9B46F78651D4787F219509')",
      "remote", "/content/package[@name=Great Outdoors]"
   objPPRep.Visible
End Sub
```

Related Topics

- Chapter 4, “Methods,” on page 71
- Chapter 5, “Properties,” on page 259

Parent Method

Returns the name of the parent category.

Syntax

Dimension.**Parent**

Applies To

Dimension Object

Discussion

Use this method to identify the name of the parent category in the dimension. Top level, alternate, and calculated categories do not have a parent.

Use the Has Parent method to determine if the current category has a parent category. If the Parent method is used when there is no parent, an error occurs.

Return Type

String

Example

For the category that the first dimension is filtered on, this example determines if it has a parent category, whether it is visible, and displays the parent name.

```
Sub Main()
   Dim objPPRep As Object
   Dim objDimension As Object
   Dim strName As String
   Dim strVisible As String
   Set objPPRep = GetObject(,"CognosPowerPlay.Report")
   Set objDimension = objPPRep.DimensionLine.Item(1)
   objDimension.ChangeToTop
   If objDimension.HasParent = 0 Then
      Msgbox "The " & objDimension.Name & " dimension
has " & _
         "no parent."
      strName = objDimension.Children.Item(1).Name
      objDimension.Change strName
      If objDimension.Visible = -1 Then
         strVisible = "visible"
      Else
         strVisible = "not visible"
      End If
      MsgBox "The " & objDimension.Name & " dimension
is " & _
         strVisible & "." & chr$(10) & chr$(10)
& _
         "Its parent is " & objDimension.Parent &
"."
   End If
   Set objDimension = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "Dimension Object" on page 19
- "HasParent Method" on page 158

Paste Method (Reporter)

Pastes the contents of the Clipboard into the selected categories of the Report object.

Syntax

Report.**Paste**

Applies To

Report Object

Discussion

It is possible to paste between reports, using the same multidimensional cube.

This method is only available if the Report object is in Reporter Mode (ExplorerMode property set to False).

Return Type

Nothing

Example

This example opens two reports, copies columns from one report, pastes them into the second report, and saves the second report.

```
Sub Main()
   Dim objPPRep1 as Object
   Dim objPPRep2 as Object
   Set objPPRep1 = CreateObject("CognosPowerPlay.Report")
   Set objPPRep2 = CreateObject("CognosPowerPlay.Report")
   objPPRep1.Open "C:\Cubes and Reports\Sample1.ppx"
   objPPRep2.Open "C:\Cubes and Reports\Sample2.ppx"
   objPPRep1.ExplorerMode = False
   objPPRep2.ExplorerMode = False
   objPPRep1.Columns.Item("Back Packs").Select
   objPPRep1.Copy
   objPPRep2.Paste
   objPPRep2.Save
   objPPRep1.Close
   objPPRep2.Close
   Set objPPRep1 = Nothing
   Set objPPRep2 = Nothing
End Sub
```

Related Topics

- "ActiveReport Method" on page 82
- "Report Object" on page 37

PDFFile Method

Sets the file name of the PDF object when it is saved.

Syntax

Report.**PDFFile**(FileName, Overwrite)

Applies To

Report Object

Discussion

Use this method to identify the fully-qualified name for saving a report as a PDF. You must use the Save method to save the report as a PDF.

You can specify the Overwrite parameter to determine whether to overwrite an existing PDF.

Parameter	Description
FileName	Required. Specifies the name of the PDF to save. The fully-qualified name must be enclosed in quotation marks with the .pdf file extension, such as "c:\Cognos\pdf\PDFSample.pdf" Type: String
Overwrite	Required. Specifies whether you can overwrite saved reports as PDFs. If True, saving the report as a PDF will overwrite another file with the same name in the same location. If False, the file will save as a PDF only if the name is unique for the specified location. Default: False Type: Boolean

Return Type

Nothing

Example

This example opens a report, sets a name to save the report as a PDF file, sets options for saving the report, and then saves the report in portable document format (.pdf).

```
Sub Main()
   Dim objPDF as Object
   Dim objPPRep as Object
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.Open( "c:\Cognos\sample.ppx" )
   objPPRep.visible( TRUE )
   Set objPDF = objPPRep.PDFFile( "c:\Cognos\PDFSample"
, True )
   With objPDF
      .SaveEntireReport = False
      .SaveAllCharts = True
      .AxisOnAllPages = True
      .ChartTitleOnAllPages = False
      .IncludeLegend = True
```

```
      .SetListOfLayersToSave objPPRep.Layers
      .SetListOfRowsToSave objPPRep.Rows
   End With
   objPDF.Save
   Set objPPRep = Nothing
   Set objPDF = Nothing
End Sub
```

Related Topics

- "Application Object" on page 11

Percent Method

Adds one or more percent categories based on either a category or a constant value.

Syntax

object.**Percent(Operand [, Reverse])**

Applies To

Column Object

Columns

Layer Object

Layers

Row Object

Rows

Discussion

This method always requires an Operand since the method calculates the percent on a pair of values. It creates a category for the result of the percent. The percent is reversible.

Depending on whether the method was applied to an object or a collection, the results are returned respectively as an object or a collection.

In Explorer mode, the new calculation is inserted directly after the last operand. In Reporter mode, the new calculation is inserted directly after the active row or column.

References to the position of an object in the collection are not valid after you use this method.

In Explorer mode, if you change a report by removing a level, drilling, filtering or nesting, then all calculations that can not be created in the changed report disappear.

Parameters	Description
Operand	Required. Specifies either a constant value or a category object. Type: Variant
Reverse	Optional. Specifies whether the results show the value of the category as a percentage of the Operand or the opposite. False = the results show the value of the category as a percentage of the operand True = the results show opposite. Type: Boolean

Return Type

Object

Example

This example takes the percent of individual rows based on the value of a Summary row

```
Sub Main()
   Dim objPPRep as Object
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.New "C:\Cubes and Reports\Great Outdoors.mdc",True
   objPPRep.Visible = True
   objPPRep.Rows.SubSet _
      ("2008","1997").Percent objPPRep.Rows("Years")
   objPPRep.SaveAs "Percent Report.ppx"
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "Column Object" on page 15
- "Layer Object" on page 28
- "Report Object" on page 37
- "Row Object" on page 41

PercentGrowth Method

Calculates the percentage change between two categories or measures.

Syntax

object.**PercentGrowth**(Operand [, Reverse])

Applies To

Column Object

Row Object

Discussion

Use this method to calculate the magnitude and direction of the percentage growth between values in selected rows or columns over time or across categories.

This method requires an operand because the calculation is performed on a pair of values. In Explorer mode, the operands of the calculation must be from the same dimension, from the same axis, and from the same level.

In Explorer mode, the new calculation category is inserted directly after the last operand category. In Reporter mode, the new calculation category is inserted directly after the active row or column

References to the position of an object in the collection are not valid after you use this method.

In Explorer mode, if you change a report by removing a level, drilling, filtering or nesting, all calculations that can not be created in the changed report disappear.

Parameters	**Description**
Operand	Required. Specifies a category object. Type: Object
Reverse	Optional. Specifies whether to reverse the order of the operands. Default: False Type: Boolean

Return Type

Object

Example

This example calculates the percentage growth between row 1 and row 2 and inserts the result as a new row.

```
Sub Main()
   Dim objPPRep As Object
   Dim objRslt As Object
   Set objPPRep = GetObject( , "CognosPowerPlay.Report")
   Set objRslt = objPPRep.Rows.Item(1).PercentGrowth
_
      (objPPRep.Rows.Item(2))
   Set objRslt = Nothing
```

```
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "Addition Method (Collections)" on page 95
- "Addition Method (Objects)" on page 96
- "CumPercentOfBase Method" on page 122
- "Division Method" on page 136
- "Multiplication Method (Collections)" on page 180
- "Multiplication Method (Objects)" on page 182
- "Percent Method" on page 194
- "Subtraction Method (Collections)" on page 244
- "Subtraction Method (Objects)" on page 246

PercentOfBase Method

Adds one or more percent of base categories using a category from a different dimension as the base.

Syntax

object.**PercentOfBase**(BaseCategory)

Applies To

Column Object

Columns

Row Object

Rows

Discussion

To determine the percent of base for a row, use a column as the base category. To determine the percent of base for a column, use a row as the base category.

Depending on whether the method was applied to an object or a collection, the results are returned respectively as an object or a collection. The new calculation is inserted directly after the last operand.

References to the position of an object in the collection are not valid after you use this method.

In Explorer mode, if you change a report by removing a level, drilling, filtering or nesting, then all calculations that can not be created in the changed report disappear.

Parameters	Description
BaseCategory	Required. Specifies the category on which the calculation is based. Type: Object

Return Type

Object

Example

This example calculated the percent of base for the Columns collection, using 2008 as the base category.

```
Sub Main()
   Dim objPPRep as Object
   Dim objPPRes as Object
   Set objPPRep = GetObject(, "CognosPowerPlay.Report")
   objPPRep.ExplorerMode = False
   objPPRep.Visible = True
   Set objPPRes = objPPRep.Columns.PercentOfBase _
      (objPPRep.Rows.Item("2008"))
   objPPRep.SaveAs "New Report.ppx"
   Set objPPRep = Nothing
   Set objPPRes = Nothing
End Sub
```

Related Topics

- "Column Object" on page 15
- "Columns" on page 52
- "Row Object" on page 41
- "Rows" on page 66

Print Method

Returns a Print object.

Syntax

Report.**Print**

Applies To

Report Object

Discussion

You can set the properties of the Print object to determine

- the number of copies
- whether colors will print as patterns

- which rows and layers to print
- where to print the display's titles, summary categories, legends, axis and labels
- whether the copies are collated
- whether to print all displays or only the selected display

Return Type

Object

Example

This example opens a report and prints one copy of all the data for the second graphical display only.

```
Sub Main()
   Dim objPPRep as Object
   Dim objRepPrt as Object
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.Open "C:\Cubes and Reports\Sample2.ppx"
   Set objRepPrt = objPPRep.Print
   objRepPrt.PrintAllCharts = False
   objRepPrt.SetListOfRowsToPrint objPPrep.Rows
   objRepPrt.SetListOfLayersToPrint objPPRep.Layers
   objRepPrt.SetChartToPrint objPPRep.Graphs.Item(2)
   objRepPrt.IncludeLegend = True
   objRepPrt.ChartTitleOnAllPages = True
   objRepPrt.SummariesOnAllPages = True
   objRepPrt.AxisOnAllPages = True
   objRepPrt.Collate = True
   objRepPrt.Copies =1
   objRepPrt.PrintOut
   Set objRepPrt = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "Print Object" on page 33
- "ActiveReport Method" on page 82

PrintOut Method

Prints the Report object.

Syntax

Print.**PrintOut**

Applies To

Print Object

Discussion

If you use the SetListOfRowsToPrint and SetListOfLayersToPrint methods, you must set them before using the PrintOut method. If you do not set these methods you receive "Invalid method or property error" for the PrintOut method.

After calling the PrintOut method, the Print object is re-initialized. To reuse this object, if you do not want to apply the default settings, you must reset the properties.

Return Type

Nothing

Example

This example opens a report and prints one copy, and then resets the properties of the Print object to use the default settings and prints two copies.

```
Sub Main()
   Dim objPPRep as Object
   Dim objRepPrt as Object
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.Open "C:\Cubes and Reports\Sample2.ppx"
   Set objRepPrt = objPPRep.Print
   objRepPrt.PrintAllCharts = False
   objRepPrt.SetListOfRowsToPrint objPPrep.Rows
   objRepPrt.SetListOfLayersToPrint objPPRep.Layers
   objRepPrt.SetChartToPrint objPPRep.Graphs.Item(2)
   objRepPrt.IncludeLegend = True
   objRepPrt.ChartTitleOnAllPages = True
   objRepPrt.SummariesOnAllPages = True
   objRepPrt.AxisOnAllPages = True
   objRepPrt.Collate = True
   objRepPrt.Copies = 1
   objRepPrt.PrintOut
   objRepPrt.Copies = 2
   objRepPrt.PrintOut
   Set objRepPrt = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- “Print Method” on page 198
- “Print Object” on page 33
- “PrintOut Method” on page 199
- “SetListOfLayersToPrint Method” on page 229
- “SetListOfRowsToPrint Method” on page 232

PublishToPortal Method

Creates a report in the IBM Cognos Analytics content store. Users access the report using the Cognos Analytics portal.

Syntax

Report.**PublishToPortal** ReportName, TargetFolder [, ReportDescription]

Applies To

Report Object

Discussion

Use this method to publish a report to the IBM Cognos Analytics content store. After you publish the report, web-based users can access a PDF version of the report using the Cognos Analytics portal.

To use this method you must

- specify a report that uses remote package as the data source
- be a user with authorized access to the remote package
- have write-access to a folder on the IBM Cognos Analytics server

If the report is successfully published, the method returns True.

If the target folder already contains an object with the same name as the report, you will receive an error indicating the publish failed.

NOTE: To allow updates to a published report, you must save the report after you publish. If you do not save the report after publishing, you can not republish using the same report name.

Parameters	**Description**
ReportName	Required. Specifies the name displayed in the Cognos Analytics portal. Type: String
TargetFolder	Required. Specifies the search path or store ID of the folder that will contain the report. Search path example, "/content/folder[@name='My Folders']/folder[@name='Sales']" Store ID example, "storeID('iAA1ECBF2EA9B46F78651D4787F219509')" Type: String
ReportDescription	Optional. Specifies a description of the report to display in the Cognos Analytics portal. Type: String

Return Type

Boolean

Example

This example publishes a report to the IBM Cognos Analytics content store for display in the Cognos Analytics portal.

```
Sub Main()
     Dim objPPRepRemote As Object
     Set objPPRepRemote = CreateObject("CognosPowerPlay.Report")
     Dim objPPApp As Object
     Set objPPApp = objPPRepRemote.Application
     objPPRepRemote.Open "C:\PPlay.ppx"
     objPPRepRemote.PublishToPortal "Tents","/content/folder[@name='MyFolder']
          /package[@name='GO8']", "Q1 sales"
     objPPRepRemote.Save
     objPPRepRemote.Close
     set objPPRepRemote = Nothing
End sub
```

Quit Method

Exits IBM Cognos PowerPlay.

Syntax

Application.**Quit**

Applies To

Application Object

Discussion

This method is the same as using the Exit command (File menu) in the application. The method prompts the user to save the changes for each modified visible report. After closing all visible reports, the method hides the application and, if no documents are using the application, the application will terminate.

Return Type

Nothing

Example

This example creates an instance of the PowerPlay Application object and shows some its properties to the user before the application closes.

```
Sub Main()
   Dim objPPlayApp as Object
   Set objPPlayApp = CreateObject("CognosPowerPlay.Application")
   objPPlayApp.Visible = 1
   MsgBox "The name of the Application is " &objPPLayApp.Name
```

```
   MsgBox "The location of the Application is " _
      &objPPLayApp.Path
   MsgBox "The Application version is " &objPPLayApp.Version
   objPPlayApp.Quit
   Set objPPlayApp = Nothing
End Sub
```

Ranges Method

Returns one Range object or the entire collection.

Syntax

Exception.**Ranges**

Applies To

Exception Object

Discussion

If no index is specified, a Ranges collection is returned, otherwise the Ranges method returns the requested Range object.

Return Type

Object

Example

This example creates an exception with one range and then applies it to a new report. The style that is applied to the exception must be predefined.

```
Sub Main()
   Dim objPPRep As Object
   Dim strExceptionName As String
   strExceptionName = "At Least 400 Thousand"
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.New "C:\Cubes and Reports\Great Outdoors.mdc",True
   objPPRep.ExplorerMode = False
   objPPRep.Visible = True
   objPPRep.Exceptions.Add strExceptionName
   objPPRep.Exceptions.Item(strExceptionName).Ranges.Add
_
       "Minimum",399999,"Good News"
   objPPRep.Columns.Exception =strExceptionName
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "Exception Object" on page 22
- "Range Object" on page 35

Rank2 Method

Ranks and sorts Row or Column objects.

Syntax

object.**Rank2**([ShowCount] [,[PortionToShow] [,[SortOrder] [,[RankSequence] [,[AutoRank]]]]])

Applies To

Column Object

Row Object

Discussion

Rank2 sorts and ranks a Column object based on a Row object, or a Row object based on a Column object.

If one of its parameters is not included, a comma is required as a placeholder. When a parameter is not specified, its corresponding entry in your IBM Cognos PowerPlay preferences settings is used.

You can set the number of categories to rank using the ShowCount parameter. If this parameter is not included, all categories appear.

In automatic rank mode (AutoRank = 1), changing data values or adding new values causes a re-rank action.

Rank2 behaves differently from the Sort method in that Rank2 adds a rank sequence column to the report. By setting the RankSequence parameter, you can have the highest value or the lowest value ranked first.

References to the position of an object in the collection are not valid after you use this method.

Although it is not included in the PowerPlay methods list, the Rank method continues to provide backwards compatibility for existing PowerPlay 5.2x macros. The Rank function maps its parameters to the new ranking functionality and then calls Rank2. The Rank method has 4 parameters: ShowCount, PortionToShow, SortOrder, and ShowRankCategory, whereas the Rank2 method has 5 parameters: ShowCount, PortionToShow, SortOrder, RankSequence, and AutoReRank. (ShowRankCategory from the Rank method is an optional parameter where you specified whether the new rank category was visible).

Existing macros that use the Rank method attempt to map the settings to PowerPlay versions 6.0 and later functionality for ranking automation. We strongly recommend that you update existing macros that use the Rank method to use the Rank2 method. The Rank function continues to exist to support macros that have not been changed to use the new ranking OLE method, Rank2.

Parameters	Description
ShowCount	Optional. Specifies the number of categories to show. If not used, the Preferences setting is used. Type: Variant
PortionToShow	Optional. Specifies what will be visible. If not used, the Preferences setting is used. 0 = All 1 = Top n, where n is the number of rows or columns. 2 = Bottom n, where n is the number of rows or columns. Type: Variant
SortOrder	Optional. Specifies how categories will be sorted. If not used, the Preferences setting is used. 0 = none 1 = descending 2 = ascending Type: Variant
RankSequence	Optional. Specifies whether the highest value or lowest value will be the highest ranked ordinal (1): 0= lowest value is ordinal 1 1= highest value is ordinal 1 Type: Variant
AutoRank	Optional. Specifies a value that states if rows are re-ranked automatically when the report changes. If True, the rows are automatically re-ranked. Type: Boolean

Return Type

Nothing

Example

This example ranks columns based on the last row. The rank shows the top five columns in ascending order. The highest value will have ordinal one and auto re-rank will be activated.

```
Sub Main()
   Dim objPPRep As Object
   Dim intLastRow As Integer
   Const show_five = 5
   Const top_five = 1
   Const rank_high = 1
```

```
   Const re_rank = 1
   Const ascending = 2
   Set objPPRep = GetObject(, "CognosPowerPlay.Report")
   If objPPRep.Graphs.Active.Type <> 0 Then
      objPPRep.Graphs.Active.SetType 0
   End If
   intLastRow = objPPRep.Rows.Count
   objPPRep.Rows.Item(intLastRow).Rank2 _
         show_five, top_five, ascending, rank_high, re_rank
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "Column Object" on page 15
- "Row Object" on page 41

Remove Method

Removes a single object or all objects from a collection.

Syntax

object.**Remove**

Applies To

AdvancedQuery Object

CategoryList Object

Column Object

Columns

Exception Object

Exceptions

FindQuery Object

Graph Object

Graphs

Layer Object

Layers

ParentageQuery Object

Range Object

Ranges

ReportQueries

Row Object

Rows

Discussion

Removing a collection object deletes all the objects contained by the object. The Remove method associated with the ReportQueries collection, removes a specific object in the collection.

A blank row or column belongs to the category immediately before a row or column. When a Row or Column object is removed from a report, the blank row or column after it is also removed.

References to the position of an object in the collection are not valid after you use this method.

When you remove an exception using the Remove method, it will not remove the exception from the PPEXCEPT.INI file. It will only remove the exception from that instance of the report. When you reopen the report, the removed exception, if it is shared, will reappear. To remove a shared exception, use the Highlight Exceptions dialog box.

Return Type

Nothing

Example

This example removes one column and one row from the open report.

```
Sub Main()
   Dim objPPRep As Object
   SET objPPRep = GetObject(, "CognosPowerPlay.Report")
   objPPRep.ExplorerMode = False
   objPPRep.Columns.Item("2008 Q 1").Remove
   objPPRep.Rows.Item("Far East").Remove
   objPPRep.Save
   objPPRep.Close
   Set objPPRep = Nothing
End Sub
```

Related Topics

- Chapter 4, “Methods,” on page 71
- Chapter 5, “Properties,” on page 259

Remove Method (ReportQueries)

Removes all query objects from the ReportQueries collection.

Syntax

ReportQueries.**Remove** (RemoveCategoriesFromReport)

Applies To

ReportQueries

Discussion

Use this method to remove all query objects from the ReportQueries collection. To remove only a single item, get the specific item from the collection using the Item method, and then use the Remove method on the individual report query.

The RemoveCategoriesFromReport parameter determines what action to perform. If True, the syntax

```
objPPrep.ReportQueries.Remove(True)
```

removes all queries and categories from the report. If False, the syntax

```
Set objQuery = objPPrep.ReportQueries.Item(1)
objQuery.Remove(False)
```

removes a single query from the report at the specified index, and also removes the categories.

Any object references made prior to using this method are not valid afterward.

Parameter	Description
RemoveCategoriesFromReport	Required. Specifies whether to remove subset categories from the report. True = the categories are removed from the report False = leaves the categories in the report and breaks the subset. Type: Boolean

Return Type

Nothing

Example

This example counts the number of subset definitions in the active report and deletes them, if they exist.

```
Sub Main()
   Dim objPPRep As Object
   Dim intCount As Integer
   Dim intx As Integer
   Set objPPRep = GetObject(,"CognosPowerPlay.Report")
   intCount = objPPRep.ReportQueries.Count
   If intCount = 0 Then
      Msgbox "There are no subset definitions in this"
& _
         " report.", , "Subsets"
   Else
      objPPRep.ReportQueries.Item(1).Remove True
```

```
      Msgbox "1 subset definition has been removed.",
' _
         "Subsets"
      intCount = objPPRep.ReportQueries.Count
      If intCount > 0 Then
         objPPRep.ReportQueries.Remove False
         Msgbox intCount & " subset definition(s)
" & _
            "have been removed.", , "Subsets"
      End If
   End If
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "Add Method (ReportQueries)" on page 90
- "ReportQueries" on page 63
- "ReportQueries Method" on page 210

RemoveLevel Method

Removes a column, layer, or row level from a nested crosstab in a report.

Syntax

collection.**RemoveLevel**(NestingLevel)

Applies To

Columns

Layers

Rows

Discussion

Use this method to remove unwanted data from a report. Removing a level only removes the data for that level. In a nested crosstab report, you can remove any level at any time without deleting the children of the deleted level. For example, you can have a crosstab with three levels of nesting based on years (nesting level 2), quarters (nesting level 1), and months (nesting level 0). You can remove the quarters level, and the report will show two levels of nesting: years and months. Removing the middle level (quarters) does not remove the lowest level (months).

In Explorer mode, this method applies only to nested crosstabs. If the graph shown is not a nested crosstab, the method generates an error.

In Reporter mode, you can remove a level, even if it is not in a nested crosstab.

An error occurs if the nesting level is invalid.

Parameter	Description
NestingLevel	Required. Specifies the nested level to remove from the display. A nesting level of 0 specifies the inner-most level, 1 for the next level up, and continues to increment by 1 for each level in the report. Type: Integer

Return Type

Nothing

Example

This example displays the name of the first column and row nesting level, and then removes levels from the report.

```
Sub Main()
   Dim objPPRep as Object
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.New "C:\Cognos\sample.mdc", True
   objPPRep.Visible = True
   objPPRep.ExplorerMode = False
   Msgbox "Nested name of first column:  " & _
      objPPRep.Columns.Item(1).NestedName
   Msgbox "Nested name of first row:  " & _
      objPPRep.Rows.Item(1).NestedName
   objPPRep.Rows.RemoveLevel(1)
   objPPRep.Columns.RemoveLevel(1)
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "AddLevel Method" on page 98
- "Levels Method" on page 169

ReportQueries Method

Returns a ReportQueries collection.

Syntax

Report.**ReportQueries**

Applies To

Report Object

Discussion

Use this method to access the ReportQueries collection when creating or defining a subset definition using the AdvancedQuery, FindQuery, ParentageQuery, or ValueRestriction objects.

The ReportQueries collection can be blank or already contain existing subset definitions. For example, an existing report may already have subset definitions defined.

Return Type

Object

Example

This example creates an AdvancedQuery (type 3) subset definition that retrieves all categories except those belonging to Europe. This subset is then added to the report as layers.

```
Sub Main()
   Dim strCubePath As String
   Dim objPPRep As Object
   Dim objCatList As Object
   Dim objAdvanced As Object
   strCubePath = "C:\Cubes and Reports\Great Outdoors.mdc"
      Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.New strCubePath, 1
   objPPRep.ExplorerMode = False
   objPPRep.Visible = True
   Set objCatList = objPPRep.CategoryList()
   objCatList.Add 0,"Locations"
   objPPRep.Layers.Add objCatList
   Set objAdvanced = objPPRep.ReportQueries.Add(3)
   With objAdvanced
      .Name = "Americas & Far East"
      .Dimension = "Locations"
      .Level "Country"
      .Exclude "Europe"
      .Execute
      .AddToReport 2,1,4
   End With
   Msgbox "Name: " & objAdvanced.Name & chr$(10)
& _
      "Dimension: " & objAdvanced.Dimension &
chr$(10) & _
      "Level List: " & objAdvanced.LevelList &
chr$(10) & _
      "Query Type Code: " & objAdvanced.Type &
chr$(10) & _
      "Number of Categories: " & objAdvanced.Count
& _
      chr$(10) & _
```

```
      "First Category: " & objAdvanced.Item(1).Name,
_
      ,"Subset"
   Set objAdvanced = Nothing
   Set objCatList = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "Add Method (ReportQueries)" on page 90
- "AddToReport Method" on page 101
- "AdvancedQuery Object" on page 8
- "FindQuery Object" on page 23
- "ParentageQuery Object" on page 31
- "ReportQueries" on page 63

Reports Method

Returns one Report object or the entire collection.

Syntax

Application.**Reports**

Applies To

Application Object

Discussion

If no index is specified, a Reports collection is returned. Otherwise the method returns the requested Report object.

Return Type

Object

Example

This example manipulates the first object in the Reports collection.

```
Sub Main()
   Dim objPPApp as Object
   Dim objPPRep as Object
   Set objPPRep = GetObject ("C:\Cubes and Reports\Layer2.ppx")
   Set objPPApp = objPPRep.Application
   objPPApp.Visible = True
   objPPApp.Reports.Item(1).ExplorerMode = False
   objPPApp.Reports.Item(1).Layers.Subset(1,2).Remove
   objPPApp.Reports.Item(1).Layers.Subset(1,2).Addition
   objPPApp.Reports.Item(1).Layers.Maximum
   Set objPPRep = Nothing
   Set objPPApp = Nothing
```

```
End Sub
```

Related Topics

- "Application Object" on page 11
- "Report Object" on page 37
- "Reports" on page 65

ResetPrintOptionsToDefault Method

Resets print options back to the default settings.

Syntax

objPrint.**ResetPrintOptionsToDefault**

Applies To

Print Object

Discussion

Print options are initialized to the options saved with the report. Unless a print macro overrides a property, its state will correspond to the way in which the author saved the report.

This method applies all default print options as opposed to the settings saved with the report.

Return Type

Nothing

Example

This example opens a report and prints it using the default settings as opposed to the print settings saved with the report.

```
Sub Main()
   Dim objPPRep as Object
   Dim objRepPrt as Object
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.Open "C:\Cubes and Reports\Sample2.ppx"
   Set objRepPrt = objPPRep.Print
   objRepPrt.ResetPrintOptionsToDefault.
   objRepPrt.PrintOut
   Set objRepPrt = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- Chapter 4, "Methods," on page 71
- Chapter 5, "Properties," on page 259
- "Print Object" on page 33

Restore Method

Restores the object window to its original size and position.

Syntax

object.**Restore**

Applies To

Application Object

Report Object

Discussion

This method does not reveal any hidden windows. To do so, use the Visible attribute of either the application or a Report object.

Return Type

Nothing

Example

This example finds a visible instance of IBM Cognos PowerPlay and restores it.

```
Sub Main()
   Dim objPPApp as Object
   Set objPPApp = GetObject( ,"CognosPowerPlay.Application"
)
   objPPApp.Restore
   Set objPPApp = Nothing
End Sub
```

Related Topics

- "Application Object" on page 11
- "ActiveReport Method" on page 82
- "Report Object" on page 37

Rollup Method

Groups categories containing calculated values to create a new, dynamic calculation.

Syntax

object.**Rollup** (Operand)

Applies To

Column Object

Layer Object

Row Object

Discussion

Rollup lets you select an arbitrary group of numeric categories from within a report and perform a rollup calculation on them. The nature of the calculation depends on the measure of the categories used in the rollup. Only categories that reside in the same Dimension object can be included in the rollup.

For objects, the Operand parameter sets the starting point for the rollup, whereas the object specification sets the ending point. For example, here the Operand parameter sets the starting point at the second column in the report, while the end point is the fourth column.

```
objPRep.Columns.Item(4).Rollup (PPRep.Columns.Item(2))
```

Parameters	Description
Operand	Required. Specifies a Category object. Type: Variant

Return Type

Object

Example

This example finds the first row in the report beginning with the word Star, and performs a dynamic rollup using the first row and the found row.

```
Sub Main()
   Dim objPPRep As Object
   Dim intFound As Integer
   Dim intIndex As Integer
   Dim objRollup As Object
   Set objPPRep = GetObject(,"CognosPowerPlay.Report")
   objPPRep.ExplorerMode False
   objPPRep.Rows.Item(1).Activate
   intFound = objPPRep.FindNext("Star", 2, False, 1)
   If intFound <> 0 Then
      intIndex = objPPRep.Rows.Active.Index
      Set objRollup = objPPRep.Rows.Item _
         (intIndex).Rollup(objPPRep.Rows.Item(1))
   Else
      Msgbox "No matching rows were found.",,"Not Found"
   End If
   Set objPPRep = Nothing
End Sub
```

Related Topics

- Chapter 4, “Methods,” on page 71

Rows Method

Returns one Row object or the entire collection.

Syntax

Report.**Rows**

Applies To

Report Object

Discussion

If no index is specified, the method returns a Rows collection. Otherwise, it returns the requested Row object.

Return Type

Object

Example

This example uses the Rows and AddBlanks methods to add a blank row before the last row.

```
Sub Main()
   Dim objPPRep As Object
   Dim intRow As Integer
   Dim intColumn As Integer
   Set objPPRep = GetObject(,"CognosPowerPlay.Report")
   objPPRep.ExplorerMode = False
   intRow = objPPRep.Rows.Count - 1
   intColumn = objPPRep.Columns.Count - 1
   objPPRep.Rows.Item(intRow).Select
   objPPRep.AddBlanks
   objPPRep.Rows.Unselect
   objPPRep.Columns.Item(intColumn).Select
   objPPRep.AddBlanks
   objPPRep.Columns.Unselect
   objPPRep.Rows.ItemAtLevel(intRow,0).SelectBlank(1)
   objPPRep.Columns.ItemAtLevel(intColumn,0).SelectBlank(1)
   Msgbox " A blank row and column have been added "
& _
      "and selected.",64,"Blanks"
   objPPRep.Rows.ItemAtLevel(intRow,0).UnselectBlank(1)
   objPPRep.Columns.ItemAtLevel(intColumn, _
      0).UnselectBlank(1)
   Msgbox " The blank row and column have now been "
& _
      "unselected.",64,"Blanks"
   objPPRep.Save
   Set objPPRep = Nothing
```

```
End Sub
```

Related Topics

- "Report Object" on page 37
- "Row Object" on page 41
- "Rows" on page 66

Save Method

Saves one or all Report objects.

Syntax

object.**Save**

Applies To

Report Object

"SaveAsPDF Object" on page 44

Discussion

Use this method to save changes to an existing report that you have already named and saved by using the SaveAs method. If you use the Save method to save a new report for the first time, you will get an error because the macro cannot determine the name of the file to save. This method will overwrite the existing report. Use only when you are certain you want to save the changes.

You can also use this method to save a report as a PDF. Use this method when you want to capture the original report (including all fonts, images, graphics, and formatting), regardless of which client the report was created in. A PDF is useful for distributing standard reports using Adobe Reader and provides quality multipage output. PDF files are compact, portable, and platform-independent.

Rows and layers that are suppressed or hidden do not appear in the Rows and Layers when saving the report as a PDF.

Return Type

Nothing

Example

This example multiplies a column in a report by 10 percent. The report is then saved and closed.

```
Sub Main()
   Dim objPPRep As Object
   Set objPPRep = GetObject( , "CognosPowerPlay.Report")
   objPPRep.ExplorerMode = False
   objPPRep.Columns("2008 Q 1").Multiplication(1.10)
   objPPRep.Save
   objPPRep.Close
   Set objPPRep = Nothing
```

```
End Sub
```

This example opens a report, sets options for saving the report, and then saves the report as a PDF.

```
Sub Main()
   Dim objPDF as Object
   Dim objPPRep as Object
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.Open( "c:\Cognos\sample.ppx" )
   objPPRep.visible( TRUE )
   Set objPDF = objPPRep.PDFFile( "c:\Cognos\PDFSample"
, True )
   With objPDF
      .SaveEntireReport = False
      .SaveAllCharts = True
      .AxisOnAllPages = True
      .ChartTitleOnAllPages = False
      .IncludeLegend = True
       SetChartToSave objPPRep.Graphs.Item( 1 )
      .SetListOfLayersToSave objPPRep.Layers
      .SetListOfRowsToSave objPPRep.Rows
   End With
   objPDF.Save
   Set objPPRep = Nothing
   Set objPDF = Nothing
End Sub
```

Related Topics

- "Report Object" on page 37
- "Reports" on page 65

SaveAs Method

Saves the Report object with a different name and, if desired, a different format.

Syntax

Report.**SaveAs** FileName [, Format [, Overwrite[, Password]]]

Applies To

Report Object

Discussion

Use the SaveAs method to save a new report for the first time, to save an existing report with a different file name than the original report, or to save an existing report in a different file format than the original report. SaveAs is the only method that can be used to write the data in formats other than IBM Cognos PowerPlay report. The following formats are available:

- PowerPlay portable report (.ppx)
- MDC (multidimensional cube)

- ASCII file, which looks like a screen capture of the report and which the PowerPlay application cannot open
- Excel Spreadsheet (.xls), which the PowerPlay application cannot open

In Windows, if the extension of the FileName is different than the selected filter, the proper extension is appended to FileName.

PowerPlay can save reports generated in the Extensible Markup Language (.xml) as PowerPlay portable reports (.ppx). Users can read these reports on both a Windows and UNIX platform. Most web browsers can read this standard report format.

Use the SavePDF method to save the report in portable document format (.pdf).

Default: PowerPlay report (.ppx)

Parameters	**Description**
FileName	Required. Specifies the new name of the report. Type: String
Format	Optional. Specifies the format in which the data will be saved. The format values are 1 = not supported 2 = MDC (multidimensional cube) 3 = ASCII or CSV (comma separated values) 4 = Excel Spreadsheet 5 = PowerPlay Portable Report Default: 5 Type: Variant
Overwrite	Optional. Specifies whether to overwrite an existing file. If True, this save action will overwrite another file if it has the same name; if False, the file will be written only if the name is unique for that folder/directory. Default: True Type: Boolean
Password	Optional. Specifies a password used to secure the output. Currently, only used for the MDC format. Type: String

Return Type

Nothing

Example

This example saves an existing report with a different file name than the original report.

```
Sub Main()
   Dim objPPRep as Object
   Dim objPPRes as Object
   Set objPPRep = GetObject(, "CognosPowerPlay.Report")
   Set objPPRes = objPPRep.Columns.CumPercentOfBase _
      (objPPRep.Rows.Item("2008"))
   objPPRep.SaveAs "MyNewReport"
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "ActiveReport Method" on page 82
- "Report Object" on page 37

Select Method

Selects categories.

Syntax

object.**Select**

Applies To

Column Object

Columns

Layer Object

Layers

Report Object

Row Object

Rows

Discussion

Use this method to select categories in an object. To clear selected categories, use the Unselect method.

Return Type

Nothing

Example

This example opens a report, selects the column "Outdoor Products", and copies it to the Clipboard.

```
Sub Main()
   Dim objPPRep as Object
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.Open "C:\Cubes and Reports\Sample1.ppx"
   objPPRep.Columns.Item("Outdoor Products").Select
   objPPRep.Copy
   objPPRep.Close
   Set objPPRep = Nothing
End Sub
```

Related Topics

- Chapter 4, "Methods," on page 71
- Chapter 5, "Properties," on page 259
- "Column Object" on page 15
- "Layer Object" on page 28
- "Row Object" on page 41
- "Unselect Method" on page 251

SelectAllDimensions Method

Selects all the dimension objects in the dimension line that can be filtered when a report is opened in the IBM Cognos portal.

Syntax

DeploymentOptions.**SelectAllDimensions**

Discussion

Use this property when a report author specifies that a report consumer can filter a report is opened in the IBM Cognos portal. The user can filter unnecessary information from any dimension so that only the required information appears in the report.

This method is useful when the number of dimension objects is large. To select individual dimension objects, one at a time, use the PromptForDimension property.

When you use this method, you can filter all Dimension objects, including those that are hidden in the dimension line.

Return Type

Nothing

Example

This example specifies the prompts that the report consumer sees when the report is opened in the IBM Cognos portal. This example also specifies that the report consumer can filter all the dimensions in the dimension line list.

```
Sub Main()
   Dim objPPRep as Object
   Dim objDeploymentOptions as Object
   Set objPPRep = GetObject(,"CognosPowerPlay.Report")
   Set objDeploymentOptions = objPPRep.DeploymentOptions
   objDeploymentOptions.PromptForCurrency = True
   objDeploymentOptions.PromptForLongShortNames = True
   objDeploymentOptions.PromptForZeroSuppression = True
   objDeploymentOptions.PromptForSwapRowsAndColumns =
True
   objDeploymentOptions.SelectAllDimensions
   objPPRep.Save
   Set objDeploymentOptions = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "UnselectAllDimensions Method" on page 251

SelectBlank Method

Selects a specific blank row or column.

Syntax

object.**SelectBlank** BlankNumber

Applies To

Column Object

Row Object

Discussion

Use SelectBlank in conjunction with the Item and ItemAtLevel methods in crosstab reports. When a crosstab has blank rows or columns, SelectBlank selects the nth blank after the specified non-blank row or column item in the collection as determined by BlankNumber parameter.

Parameter	Description
BlankNumber	Required. A value indicating a blank item. BlankNumber acts as an index, where the first blank is 1, the second is 2, and so on. Type: Long

Return Type

Nothing

Example

This example uses the AddBlanks method to add a blank row before the last row and a blank column before the last column in the active report. It applies to crosstab reports in Reporter mode only.

```
Sub Main()
   Dim objPPRep As Object
   Dim intRow As Integer
   Dim intColumn As Integer
   Set objPPRep = GetObject(,"CognosPowerPlay.Report")
   objPPRep.ExplorerMode = False
   intRow = objPPRep.Rows.Count - 1
   intColumn = objPPRep.Columns.Count - 1
   objPPRep.Rows.Item(intRow).Select
   objPPRep.AddBlanks
   objPPRep.Rows.Unselect
   objPPRep.Columns.Item(intColumn).Select
   objPPRep.AddBlanks
   objPPRep.Columns.Unselect
   objPPRep.Rows.ItemAtLevel(intRow,0).SelectBlank(1)
   objPPRep.Columns.ItemAtLevel(intColumn,0).SelectBlank(1)
   Msgbox " A blank row and column have been added "
& _
      "and selected.",64,"Blanks"
   objPPRep.Rows.ItemAtLevel(intRow,0).UnselectBlank(1)
   objPPRep.Columns.ItemAtLevel(intColumn, _
      0).UnselectBlank(1)
   Msgbox " The blank row and column have now been "
& _
      "unselected.",64,"Blanks"
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "SelectBlank Method" on page 222
- "UnselectBlank Method" on page 252

SetChartToPrint Method

Specifies which Graph object of a report to print.

Syntax

Print.**SetChartToPrint** (GraphObject)

Applies To

Print Object

Discussion

Use this method to print all data of a specified Graph object. If this method is not invoked, the default is to print all data of the currently active Graph object in the report.

This method is only meaningful when the PrintAllCharts property set to False.

Parameter	Description
GraphObject	Required. Specifies the Graph object to print. Type: Object

Return Type

Nothing

Example

This example opens a report and prints one copy of all the data for the second graphical display only. This example includes the display title, summary category, and axis on all pages; and excludes the legend.

```
Sub Main()
   Dim objPPRep as Object
   Dim objRepPrt as Object
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.Open "C:\Cubes and Reports\Trend.ppx"
   Set objRepPrt = objPPRep.Print
   objRepPrt.PrintAllCharts = False
   objRepPrt.SetListOfRowsToPrint objPPrep.Rows
   objRepPrt.SetListOfLayersToPrint objPPRep.Layers
   objRepPrt.SetChartToPrint objPPRep.Graphs.Item(2)
   objRepPrt.IncludeLegend = False
   objRepPrt.ChartTitleOnAllPages = True
   objRepPrt.SummariesOnAllPages = True
   objRepPrt.AxisOnAllPages = True
   objRepPrt.Copies =1
   objRepPrt.PrintOut
   Set objRepPrt = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- Chapter 4, “Methods,” on page 71
- Chapter 5, “Properties,” on page 259
- “Graph Object” on page 25
- “Print Method” on page 198
- “Print Object” on page 33
- “PrintAllCharts Property” on page 364

SetChartToSave Method

Specifies which Graph object to save in a PDF.

Syntax

SaveAsPDF.**SetChartToSave** (GraphObject)

Applies To

SaveAsPDF Object

Discussion

Use this method to capture all data associated with a specified Graph object (including all fonts, images, graphics, and formatting) in a PDF.

When saving a report as a PDF, you can only use this method if the SaveAllCharts property is set to False.

Default: If this method is not invoked, the default is to save all the data of the currently active Graph object in the report.

Parameter	Description
GraphObject	Required. Specifies which Graph object in the report to save in a PDF. Type: Object

Return Type

Nothing

Example

This example opens a report, sets options for saving the report, and then saves the report as a PDF.

```
Sub Main()
   Dim objPDF as Object
   Dim objPPRep as Object
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.Open( "c:\Cognos\sample.ppx" )
   objPPRep.visible( TRUE )
   Set objPDF = objPPRep.PDFFile( "c:\Cognos\PDFSample"
, True )
   With objPDF
      .SaveEntireReport = False
      .AxisOnAllPages = True
      .ChartTitleOnAllPages = False
      .IncludeLegend = True
      .SetChartToSave objPPRep.Graphs.Item( 1 )
      .SetListOfLayersToSave objPPRep.Layers
```

```
      .SetListOfRowsToSave objPPRep.Rows
   End With
   objPDF.Save
   Set objPPRep = Nothing
   Set objPDF = Nothing
End Sub
```

Related Topics

- "SetListOfLayersToSave Method" on page 231
- "SetListOfRowsToSave Method" on page 233

SetDataSourceInfo Method

Stores security information for a data source in memory.

Syntax

*Application***SetDataSourceInfo** Connection Type, Location[, Password, DataSourceConnectionName]

Applies To

Application Object

Discussion

Use this method to provide security information for local PowerCube or remote package using automation. This information must be provided before opening the specified report and data source. For remote packages, if a data source uses more than one data source connection, specify the data source connection to use.

The security parameters are stored in memory for subsequent opening of reports based on the data source. To prevent the opening of subsequent reports, use DeleteDataSourceInfo or DeleteAllDataSourceInfo after all desired reports are open.

Parameter	Description
Connection Type	Required. Identifies whether the connection type is for a local cube or a remote package. This parameter can be either "local" or "remote". Type: String

Parameter	Description
Location	Required. If the connection type is local, then a fully qualified local cube is expected. For example, "C:\Cubes\Great Outdoors.mdc" If the connection type is remote, then a package search path in native encoding or a store ID is expected. Search path example, "/content/package[@name=Great Outdoors]" Store ID example, "storeID('iAA1ECBF2EA9B46F78651D4787F219509')" Type: String
Password	Optional. Specifies password required to access the secured cube. Type: String
Data Source Connection Name	Optional. When a data source includes more than one data source connection, specifics the data source connection name used in IBM Cognos Administration.

Return Type

Nothing

Example

This example sets the data source security access information record and then opens a report based on a password protected cube. Next, the data source security access record is deleted from memory.

```
Sub Main()
     Dim objPPApp As Object
     Dim objPPRep As Object
     Dim strMDCName As String
     Set objPPRep = CreateObject("CognosPowerPlay.Report")
     Set objPPApp = objPPRep.Application
     strMDCName = "C:\Cubes and Reports\Sample1.mdc"
     objPPApp.SetDataSourceInfo "local", strMDCName,
"cube_password"
     objPPRep.Open "C:\Cubes and Reports\Sample1.ppx"
     objPPApp.DeleteDataSourceInfo "local", strMDCName
     Set objPPRep = Nothing
     Set objPPApp = Nothing
End Sub
```

This example sets the data source security access information record and then opens a report based on a remote package containing ambiguous data source

connections. Datasource1 refers to the name of the data source connection to use from the package. Next, the data source security access record is deleted from memory.

```
Sub Main()
     Dim objPPApp As Object
     Dim objPPRep As Object
     Dim strPackageSearchPath As String
     Set objPPRep = CreateObject("CognosPowerPlay.Report")
     Set objPPApp = objPPRep.Application
     strPackageSearchPath = "/content/package[@name=Great Outdoors]"
     objPPApp.SetDataSourceInfo "remote", strPackageSearchPath,, "datasource1"
     objPPRep.Open "C:\Cubes and Reports\Sample1.ppx"
     objPPApp.DeleteDataSourceInfo "remote", strPackageSearchPath
     Set objPPRep = Nothing
     Set objPPApp = Nothing
End Sub
```

Related Topics

- Chapter 4, "Methods," on page 71
- Chapter 5, "Properties," on page 259

SetDrivingCategory Method

Sets the driving category for the Exception object.

Syntax

Exception.**SetDrivingCategory** (DrivingCategory, DrivingDimension)

Applies To

Exception Object

Discussion

The DrivingDimension and DrivingCategory pair identifies a unique category to base the exception definition. The selected category is the driving category for each range of values. IBM Cognos PowerPlay compares only the data in the driving category with the range of values specified, and uses the driving category to determine the parts of the report to highlight. When no driving category is specified, PowerPlay searches all categories for which the exception is defined to find values that match the specified range.

The SetDrivingCategory method is case-sensitive. The return values of the SetDrivingCategory method are

0 = Failure 1 = Success

To clear the parameter, DrivingCategory, use "<None>". For example, to clear the Exception object, Except1, of the driving category, use the following:

```
Except1.SetDrivingCategory ("<None>", 0)
```

Parameters	Description
DrivingCategory	Required. Specifies the identifier for the driving category. This parameter refers to the category label of the row or column desired. Type: Variant
DrivingDimension	Required. Specifies the category type for the driving category. Possible values are 0 = Rows 1 = Columns Type: Integer

Return Type

Boolean

Example

This example opens a report, sets the driving category for an existing exception, and saves the changes.

```
Sub Main()
  Dim objPPRep as Object
  Dim objExcept1 as Object
  Set objPPRep = GetObject("C:\Cubes and Reports\test.ppx")
  Set objExcept1 = objPPRep.Exceptions.Item("Overdue")
  objExcept1.SetDrivingCategory "Environmental Line",1
  objPPRep.Columns.Exception = "Overdue"
  objPPRep.Save
  Set objExcept1 = Nothing
  Set objPPRep = Nothing
End Sub
```

Related Topics

- Chapter 4, “Methods,” on page 71
- Chapter 5, “Properties,” on page 259
- “DrivingCategory Property” on page 300
- “Exception Object” on page 22
- “SetDrivingCategory Method” on page 228

SetListOfLayersToPrint Method

Specifies the range of layers of the report to print.

Syntax

Print.**SetListOfLayersToPrint** (ListOfLayers)

Applies To

Print Object

Discussion

Use this method to specify which report layers to print. This method must be used in conjunction with SetListOfRowsToPrint to have any effect on the printed range. If only one or neither of these methods is invoked, only the current layer and row of the report are printed.

If you use this method, set it before using the PrintOut method. If you do not set this method before the PrintOut method, you receive an "Invalid method or property error" on the PrintOut method.

After calling the PrintOut method, the Print object is destroyed. To reuse this object, you must recreate it in the same macro.

Parameters	Description
ListOfLayers	Required. Specifies the collection of Layer objects to be printed. Type: Object

Return Type

Nothing

Example

This example opens a report and prints one copy of all the data for the second graphical display only.

```
Sub Main()
   Dim objPPRep as Object
   Dim objRepPrt as Object
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.Open "C:\Cubes and Reports\Sample2.ppx"
   Set objRepPrt = objPPRep.Print
   objRepPrt.PrintAllCharts = False
   objRepPrt.SetListOfRowsToPrint objPPrep.Rows
   objRepPrt.SetListOfLayersToPrint objPPRep.Layers
   objRepPrt.SetChartToPrint objPPRep.Graphs.Item(2)
   objRepPrt.IncludeLegend = True
   objRepPrt.ChartTitleOnAllPages = True
   objRepPrt.SummariesOnAllPages = True
   objRepPrt.AxisOnAllPages = True
   objRepPrt.Collate = True
   objRepPrt.Copies =1
   objRepPrt.PrintOut
   Set objRepPrt = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- Chapter 4, “Methods,” on page 71
- Chapter 5, “Properties,” on page 259
- “Print Method” on page 198
- “Print Object” on page 33
- “PrintOut Method” on page 199
- “SetListOfRowsToPrint Method” on page 232

SetListOfLayersToSave Method

Specifies the range of layers to save in a PDF.

Syntax

SaveAsPDF.**SetListOfLayersToSave** (ListOfLayers)

Applies To

SaveAsPDF Object

Discussion

Use this method in conjunction with the SetListOfRowsToSave method to specify the range (layers and rows) to save. If neither of these methods is invoked, only the current layer and row in the report are saved.

When saving this report as a PDF, this method is ignored if the SaveEntireReport property is set to True.

Parameters	Description
ListOfLayers	Required. Specifies the collection of Layer objects to save in a PDF. Type: Object

Return Type

Nothing

Example

This example opens a report, sets options for saving the report, and then saves the report as a PDF.

```
Sub Main()
   Dim objPDF as Object
   Dim objPPRep as Object
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.Open( "c:\Cognos\sample.ppx" )
   objPPRep.visible( TRUE )
   Set objPDF = objPPRep.PDFFile( "c:\Cognos\PDFSample"
, True )
```

```
   With objPDF
      .SaveEntireReport = False
      .SaveAllCharts = True
      .AxisOnAllPages = True
      .ChartTitleOnAllPages = False
      .IncludeLegend = True
      .SetListOfLayersToSave objPPRep.Layers
      .SetListOfRowsToSave objPPRep.Rows
   End With
   objPDF.Save
   Set objPPRep = Nothing
   Set objPDF = Nothing
End Sub
```

Related Topics

- "Application Object" on page 11
- "SaveEntireReport Property" on page 379
- "SetListOfRowsToSave Method" on page 233

SetListOfRowsToPrint Method

Specifies the range of rows of the report to print.

Syntax

Print.**SetListOfRowsToPrint** (ListOfRows)

Applies To

Print Object

Discussion

Use this method to set the range of rows to print in a report. This method must be used in conjunction with SetListOfLayersToPrint to affect the printed range. If only one or neither of these methods is invoked, the current layer and row of the report are printed.

If you use this method, set it before using the PrintOut method. If you do not set this method before the PrintOut method, you receive an "Invalid method or property error" on the PrintOut method.

After calling the PrintOut method, the Print object is destroyed. To reuse this object, you must recreate it in the same macro.

Parameter	Description
ListOfRows	Required. Specifies the collection of Row objects to be printed. Type: Object

Return Type

Nothing

Example

This example opens a report and prints one copy of all the data for the second graphical display only.

```
Sub Main()
   Dim objPPRep as Object
   Dim objRepPrt as Object
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.Open "C:\Cubes and Reports\Sample2.ppx"
   Set objRepPrt = objPPRep.Print
   objRepPrt.PrintAllCharts = False
   objRepPrt.SetListOfRowsToPrint objPPrep.Rows
   objRepPrt.SetListOfLayersToPrint objPPRep.Layers
   objRepPrt.SetChartToPrint objPPRep.Graphs.Item(2)
   objRepPrt.IncludeLegend = True
   objRepPrt.ChartTitleOnAllPages = True
   objRepPrt.SummariesOnAllPages = True
   objRepPrt.AxisOnAllPages = True
   objRepPrt.Collate = True
   objRepPrt.Copies =1
   objRepPrt.PrintOut
   Set objRepPrt = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- Chapter 4, “Methods,” on page 71
- Chapter 5, “Properties,” on page 259
- “Print Method” on page 198
- “Print Object” on page 33
- “PrintOut Method” on page 199
- “SetListOfRowsToPrint Method” on page 232

SetListOfRowsToSave Method

Specifies the range of rows to save in a PDF.

Syntax

SaveAsPDF.**SetListOfRowsToSave** (ListOfRows)

Applies To

SaveAsPDF Object

Discussion

Use this method in conjunction with the SetListOfLayersToSave method to specify the range (layers and rows) to save. If neither of these methods is invoked, only the current layer and row in the report are saved.

When saving this report as a PDF, this method is ignored if the SaveEntireReport property is set to True.

Parameter	Description
ListOfRows	Required. Specifies the collection of Row objects to save in a PDF. Type: Object

Return Type

Nothing

Example

This example opens a report, sets options for saving the report, and then saves the report as a PDF.

```
Sub Main()
   Dim objPDF as Object
   Dim objPPRep as Object
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.Open( "c:\Cognos\sample.ppx" )
   objPPRep.visible( TRUE )
   Set objPDF = objPPRep.PDFFile( "c:\Cognos\PDFSample"
, True )
   With objPDF
      .SaveEntireReport = False
      .SaveAllCharts = True
      .AxisOnAllPages = True
      .ChartTitleOnAllPages = False
      .IncludeLegend = True
      .SetListOfLayersToSave objPPRep.Layers
      .SetListOfRowsToSave objPPRep.Rows
   End With
   objPDF.Save
   Set objPPRep = Nothing
   Set objPDF = Nothing
End Sub
```

Related Topics

- "Application Object" on page 11
- "SaveEntireReport Property" on page 379
- "SetListOfLayersToSave Method" on page 231

SetMacro Method

Sets the name and style for the macro used by the Exception object.

Syntax

Exception.**SetMacro** (Name, Style)

Applies To

Exception Object

Discussion

Use this method to set the name and style for a macro before using the MacroName and MacroStyle properties.

The SetMacro method searches for the specified macro only in the default macro directory. The SetMacro method will not accept a path. You can set the default macro directory using the Preferences command (File menu) or the Application method DefaultMacroDirectory.

The macro called by this method must follow the specified ExceptionMacro format in the Format for Macros Used in the Highlighting Exceptions Dialog Box topic.

Parameters	Description
Name	Required. Specifies the name of the macro. Type: String
Style	Required. Specifies the style to associate to this macro. Type: String

Return Type

Boolean

Example

This example sets the macro name and style for the first exception in the report.

```
Sub Main()
   Dim objPPRep as Object
   Dim objPPExpt as Object
   Set objPPRep = GetObject (, "CognosPowerPlay.Report")
   objPPRep.Application.DefaultMacroDirectory = _
       "C:\Samples\PowerPlay\Macros\"
   Set objPPExpt = objPPRep.Exceptions.Item ("GreatRevenue")
   objPPExpt.SetMacro "Report Sorting.mac", "Good News"
   objPPRep.SaveAs "New Exception Report.ppx"
   Set objPPRep = Nothing
End Sub
```

Related Topics

- Chapter 4, “Methods,” on page 71
- Chapter 5, “Properties,” on page 259
- “DefaultMacroDirectory Property” on page 294
- “Exception Object” on page 22
- “Highlight Exceptions Macro” on page 439
- “MacroName Property” on page 339
- “MacroStyle Property” on page 340
- “SetMacro Method” on page 235

SetMDCAccessInfo Method

Stores security access information for a local PowerCube in memory.

Syntax

Application.**SetMDCAccessInfo** MDCName[, cube_password]

Applies To

Application Object

Discussion

Use this method to provide security information for a secured PowerCube using automation. This information must be provided before opening the specified report and MDC file.

The security parameters are stored in memory for subsequent opening of reports based on the PowerCube. To prevent the opening of subsequent reports, use DeleteMDCAccessInfo or DeleteAllMDCAccessInfo after all desired reports are open.

Parameter	Description
MDCName	Required. Specifies the name of a local PowerCube. A "*" can be used as a wildcard so that the programmer can specify the default set of security parameters to use if the MDC name is not found. Type: String
cube_password	Optional. Specifies the password for a password protected PowerCube. Type: Variant

Return Type

Nothing

Example

This example sets the cube security access information record and then opens a report based on the password protected cube. Next, the mdc security access record is deleted from memory.

```
Sub Main()
     Dim objPPApp As Object
     Dim objPPRep As Object
     Dim strMDCName As String
     Set objPPRep = CreateObject("CognosPowerPlay.Report")
     Set objPPApp = objPPRep.Application
     strMDCName = "C:\Cubes and Reports\Sample1.mdc"
     objPPApp.SetMDCAccessInfo strMDCName, ", "cube_password"
     objPPRep.Open "C:\Cubes and Reports\Sample1.ppx"
     objPPApp.DeleteMDCAccessInfo(strMDCName)
     Set objPPRep = Nothing
     Set objPPApp
End Sub
```

Related Topics

- "Logon Method" on page 171
- "Logoff Method" on page 172
- Chapter 5, "Properties," on page 259
- "Application Object" on page 11

SetType Method

Sets the Graph object type.

Syntax

Graph.**SetType**(Type[, Depth] [, Vertical])

Applies To

Graph Object

Discussion

Use the following list to set the Graph object type

- 0 (Crosstab)
- 1 (Pie)
- 2 (3-D)
- 3 (Bar)
- 4 (Cluster)
- 5 (Stack)
- 6 (Line)
- 7 (Multi-Line)
- 8 (Correlated)
- 9 (Scatter)

Parameters	Description
Type	Required. Specifies the Graph object type. Type: Integer
Depth	Optional. Specifies whether the Graph object is three-dimensional (3D). Applies specifically to Type 1, 3, 4, and 5. False = types 0, 6, 7, 8 True = type 2 Default: True (for Graph object types for which this property has meaning) Type: Boolean
Vertical	Optional. Specifies whether the Graph object is a vertical display. If the property is False, it applies only to Type 3. Default: True Type: Boolean

Return Type

Nothing

Example

This example changes the display type for the first Graph object to a three-dimensional cluster bar, and displays the settings for the Graph object for an open report.

```
Sub Main()
   Dim objPPRep as Object
   Dim objPPGph as Object
   Set objPPRep = GetObject( , "CognosPowerPlay.Report")
   Set objPPGph = objPPRep.Graphs.Item(1)
   objPPGph.SetType 4, 1, 1
   MsgBox "The Graph object type is " & objPPGph.Type
& "."
   If objPPGph.Depth = -1 Then
      MsgBox "The graph is not 3D."
   Else
       MsgBox "The graph is  3D."
   End If
   Set objPPGph = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- Chapter 4, "Methods," on page 71
- Chapter 5, "Properties," on page 259
- "Graph Object" on page 25

SizeSelected Method

Applies a size to selected objects.

Syntax

Report.**SizeSelected** Size

Applies To

Report Object

Discussion

Use this method in conjunction with the Select method. After objects are selected in a report using the Select method, you can use SizeSelected to set a size for all items selected.

Parameter	Description
Size	Required. Specifies a size in pixels. Type: Long

Return Type

Boolean

Example

This example selects the last column in the report, applies a style to it, and changes the cell size.

```
Sub Main()
   Dim objPPRep As Object
   Dim intCount As Integer
   Set objPPRep = GetObject(,"CognosPowerPlay.Report")
   intCount = objPPRep.Columns.Count
   objPPRep.Columns.ItemAtLevel(intCount,0).Select
   objPPRep.StyleSelected "Good News"
   objPPRep.SizeSelected 100
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "Select Method" on page 220

Sort Method

Sorts columns, layers, or rows in ascending or descending order.

Syntax

collection.**Sort SortBy**, OppDimIndex [, Order[, AutoSort]]

Applies To

Columns

Layers

Rows

Discussion

Sort arranges data according to criteria set in the parameters. You can sort by values or labels of a column or row, or by the labels of layers. Sorting can be automatic.

In automatic sort mode (AutoSort = True), drilling up or down causes an automatic re-sort of the data using the current parameter settings. If automatic sort mode is not on (AutoSort = False), drilling up or down does not cause a re-sort of the data. Items that are already sorted and any new or changed values remain in the order in which they appear.

There is only one active ordering action per Dimension object; that is, the user can sort by labels or values, but not both, in the same report dimension.

Sort behaves differently from the Rank2 method in that it doesn't actually add anything to the Report object; it simply re-orders what the user sees on the screen. Rank2 adds a rank sequence column or row to the report.

The sort method always sorts categories that have been suppressed or hidden, since they still exist as part of the report.

Do not use the *OppDimIndex* parameter when sorting on layers.

When any of these parameters are not included, the equivalent Preferences setting is used.

Parameters	**Description**
SortBy	Required. Specifies what to sort by. 0 = no sort 1 = sort by label 2 = sort by value (not valid for layers) If 0 is set, the other three parameters have no effect. Type: Variant

Parameters	Description
OppDimIndex	Required. Specifies the column or row on which to sort the data. It is always the opposite dimension; that is, rows use a column index and columns use a row index. The index starts at 1. This parameter is not valid for layers. Type: Variant
Order	Optional. Specifies how to sort the categories. If not used, the Preferences setting is used. 1 = descending, 2 = ascending Default: descending Type: Variant
AutoSort	Optional. Specifies if rows are sorted automatically when the report changes. If not used, the Preferences setting is used. False (does not auto-sort) True (auto-sorts) Type: Boolean

Return Type

Nothing

Example

This example sorts columns by the values of the last row.

```
Sub Main()
   Dim objPPRep As Object
   Dim intRowIndex As Integer
   Const SortByValue = 2
   Const Descending = 1
   Const NoAutoSort = 0
   Set objPPRep = GetObject(, "CognosPowerPlay.Report")
   intRowIndex = objPPRep.Rows.Count
   objPPRep.Columns.Sort _
      SortByValue, intRowIndex, Descending, NoAutoSort
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "Rank2 Method" on page 204

StyleSelected Method

Applies a style to selected objects.

Syntax

Report.**StyleSelected** Style

Applies To

Report Object

Discussion

Use StyleSelected in conjunction with the Select method. Once objects are selected in a collection using Select, you can use StyleSelected to apply a format style to all items selected.

Parameter	Description
Style	Required. Specifies the value indicating a predefined style. True = successful False = error Type: String

Return Type

Boolean

Example

This example selects the last column in the report, applies a style to it, and changes the cell size.

```
Sub Main()
   Dim objPPRep As Object
   Dim intCount As Integer
   Set objPPRep = GetObject(,"CognosPowerPlay.Report")
   intCount = objPPRep.Columns.Count
   objPPRep.Columns.ItemAtLevel(intCount,0).Select
   objPPRep.StyleSelected "Good News"
   objPPRep.SizeSelected 100
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "Select Method" on page 220

Subset Method

Returns a subset of objects from the current collection.

Syntax

collection.**Subset**(IndexOfFirstCategory, IndexOfLastCategory)

Applies To

Columns

Layers

Rows

Discussion

A subset is a group of categories used to isolate information that shares some common criteria. The subset can include all the objects in the collection.

Parameters	Description
IndexOfFirstCategory	Required. Specifies the index or category name of the first category of the range. Type: Integer
IndexOfLastCategory	Required. Specifies the index or category name of the last category of the range. Type: Integer

Return Type

Object

Example

This example uses the Subset method to add the first three rows in the open report.

```
Sub Main()
   Dim objPPRep As Object
   Dim objNewCol As Object
   Dim objNewRow As Object
   Set objPPRep = GetObject( , "CognosPowerPlay.Report")
   Set objNewRow = objPPRep.Rows.Subset(1, 3).Addition
   MsgBox "The sum of the first three rows is " _
      &objPPRep.CellValue(objNewRow(1).Index,1)
   Set objNewCol = objPPRep.Columns.Item(4).Addition
_
      (objPPRep.Columns.Item(1))
   MsgBox " The sum of column one and column four is
" _
      &objPPRep.CellValue(1,objNewCol.Index)
   objPPRep.Save
   Set objNewRow = Nothing
```

```
    Set objNewCol = Nothing
    Set objPPRep = Nothing
End Sub
```

Related Topics

- Chapter 4, “Methods,” on page 71
- Chapter 5, “Properties,” on page 259
- “Columns” on page 52
- “Layers” on page 59
- “Rows” on page 66

Subtraction Method (Collections)

Subtracts a constant value or a category from one or more categories in the collection, or subtracts all categories from a constant value or another category.

Syntax

collection.**Subtraction**[(Operand [, Reverse])]

Applies To

Columns

Layers

Rows

Discussion

The method can subtract all categories from a constant value or another category. The subtraction is reversible.

Depending on whether the method was applied to an object or a collection, the results are returned respectively as an object or a collection.

In Explorer mode, the new calculation is inserted directly after the last operand. In Reporter mode, the new calculation is inserted directly after the active row or column.

References to the position of an object in the collection are not valid after you use this method.

In Explorer mode, if you change a report by removing a level, drilling, filtering or nesting, then all calculations that can not be created in the changed report disappear.

Parameters	Description
Operand	Optional. Specifies either a constant value or a category object. If you do not specify this parameter, this method creates a category to display the subtraction of all the categories in the collection. If you specify this parameter, this method subtracts the category from the operand, and creates a new category for each result. Type: Variant
Reverse	Optional. Specifies whether the category is subtracted from the Operand or the opposite. True = the category is subtracted from the Operand False = the Operand is subtracted from the category Type: Boolean

Return Type

Object

Example

This example returns the difference between two columns in a new column.

```
Sub Main()
   Dim objPPRep As Object
   Set objPPRep = GetObject( , "CognosPowerPlay.Report")
   objPPRep.ExplorerMode = False
   objPPRep.Columns.Item("Environmental Line").Subtraction
_
      objPPRep.Columns.Item("Products")
   objPPRep.Save
   objPPRep.Close
   Set objPPREp = Nothing
End Sub
```

Related Topics

- Chapter 4, “Methods,” on page 71
- Chapter 5, “Properties,” on page 259
- “Column Object” on page 15
- “Layer Object” on page 28
- “Report Object” on page 37
- “Row Object” on page 41

Subtraction Method (Objects)

Subtracts a constant value or another category from an object, or subtracts an object from the category or constant value.

Syntax

object.**Subtraction**(Operand [, Reverse])

Applies To

Column Object

Layer Object

Row Object

Discussion

For Column, Layer, and Row objects, the method subtracts the category from the operand for each category and operand pair, and creates a new category for each result.

Depending on whether the method was applied to an object or a collection, the results are returned respectively as an object or a collection.

In Explorer mode, the new calculation is inserted directly after the last operand. In Reporter mode, the new calculation is inserted directly after the active row or column.

References to the position of an object in the collection are not valid after you use this method.

In Explorer mode, if you change a report by removing a level, drilling, filtering or nesting, then all calculations that can not be created in the changed report disappear.

Parameters	**Description**
Operand	Required. Specifies either a constant value or a category object. Type: Variant
Reverse	Optional. Specifies whether the category is subtracted from the Operand or the opposite. True = the category is subtracted from the Operand False = the Operand is subtracted from the category Type: Boolean

Return Type

Object

Example

This example subtracts a constant value from each value in a column and returns the difference in a new column.

```
Sub Main()
   Dim objPPRep as Object
   Dim objPPCol as Object
   Set objPPRep = CreateObject ("CognosPowerPlay.Report")
   objPPRep.Open "C:\Cubes and Reports\Sample1.ppx"
   Set objPPCol = objPPRep.Columns.Item("Tents")
   objPPCol.Subtraction(1000)
   objPPRep.Save
   Set objPPCol = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- Chapter 4, "Methods," on page 71
- Chapter 5, "Properties," on page 259
- "Column Object" on page 15
- "Layer Object" on page 28
- "Report Object" on page 37
- "Row Object" on page 41

SwapColumnsAndLayers Method

Exchanges the positions of the Column objects and Layer objects.

Syntax

Report.**SwapColumnsAndLayers**

Applies To

Report Object

Discussion

Use this method to exchange the position of categories in a report. You can use other swapping methods to exchange the positions of rows and columns, and rows and layers.

References to the position of an object in the collection are not valid after you use this method.

Return Type

Nothing

Example

This example exchanges the positions of the columns and layers in an open report.

```
Sub Main()
   Dim objPPRep as Object
   Set objPPRep = GetObject(, "CognosPowerPlay.Report")
   objPPRep.SwapColumnsAndLayers
   objPPRep.SaveAs "New Report.ppx"
   Set objPPRep = Nothing
End Sub
```

Related Topics

- Chapter 4, "Methods," on page 71
- Chapter 5, "Properties," on page 259
- "Column Object" on page 15
- "Layer Object" on page 28
- "Report Object" on page 37
- "SwapRowsAndColumns Method"
- "SwapRowsAndLayers Method" on page 249

SwapRowsAndColumns Method

Exchanges the positions of the Row objects and Column objects.

Syntax

Report.**SwapRowsAndColumns**

Applies To

Report Object

Discussion

You can use other swapping methods exchange the positions of columns and layers, and rows and layers.

You can swap rows and columns to analyze information differently in a report, or to change the data in a report so that it will fit the current page size. For example, if the rows contain quarters of the fiscal year and the columns contain products, you can swap them so rows contain products, and columns contain quarters.

References to the position of an object in the collection are not valid after you use this method.

Return Type

Nothing

Example

This example exchanges the positions of the rows and columns in an open report.

```
Sub Main()
```

```
   Dim objPPRep as Object
   Set objPPRep = GetObject(, "CognosPowerPlay.Report")
   objPPRep.SwapRowsAndColumns
   objPPRep.SaveAs "New Report.ppx"
   Set objPPRep = Nothing
End Sub
```

Related Topics

- Chapter 4, "Methods," on page 71
- Chapter 5, "Properties," on page 259
- "Column Object" on page 15
- "Report Object" on page 37
- "Row Object" on page 41
- "SwapRowsAndColumns Method" on page 248
- "SwapRowsAndLayers Method"

SwapRowsAndLayers Method

Exchanges the positions of the Row objects and Layer objects.

Syntax

Report.**SwapRowsAndLayers**

Applies To

Report Object

Discussion

Other methods exchange columns and layers, and rows and columns.

If you use this method on a report that does not have any layers, the rows will become layers and there will be no rows in the report.

References to the position of an object in the collection are not valid after you use this method.

Return Type

Nothing

Example

This example exchanges the positions of the rows and layers in an open report.

```
Sub Main()
   Dim objPPRep as Object
   Set objPPRep = GetObject(, "CognosPowerPlay.Report")
   objPPRep.SwapRowsAndLayers
   objPPRep.SaveAs "New Report.ppx"
   Set objPPRep = Nothing
End Sub
```

Related Topics

- Chapter 4, “Methods,” on page 71
- Chapter 5, “Properties,” on page 259
- “Layer Object” on page 28
- “Report Object” on page 37
- “Row Object” on page 41
- “SwapRowsAndColumns Method” on page 248
- “SwapRowsAndLayers Method” on page 249

UnhideAllCategories Method

Makes all hidden categories visible.

Syntax

Report.**UnhideAllCategories**

Applies To

Report Object

Discussion

When categories are hidden using the Hide method, they lose all connection to the macro. This is the only method of revealing them.

References to the position of an object in the collection are not valid after you use this method.

Return Type

Nothing

Example

This example reveals all the hidden categories in an open report.

```
Sub Main()
   Dim objPPRep as Object
   Set objPPRep = GetObject(, "CognosPowerPlay.Report")
   objPPRep.UnhideAllCategories
   objPPRep.Save
   Set objPPRep = Nothing
End Sub
```

Related Topics

- Chapter 4, “Methods,” on page 71
- Chapter 5, “Properties,” on page 259
- “Hide Method” on page 159
- “ActiveReport Method” on page 82
- “Report Object” on page 37

Unselect Method

De-selects categories.

Syntax

object.**Unselect**

Applies To

Column Object

Columns

Layer Object

Layers

Row Object

Rows

Discussion

Use this method to remove all selections in an object or collection.

Return Type

Nothing

Example

This example clears the selected column from an open report.

```
Sub Main()
   Dim objPPRep as Object
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.Open "C:\Cubes and Reports\Sample1.ppx"
   objPPRep.Columns.Item("Outdoor Products").Select
   objPPRep.Copy
   ObjPPRep.Columns.Unselect
   objPPRep.Close
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "Select Method" on page 220
- "SelectBlank Method" on page 222
- "UnselectBlank Method" on page 252

UnselectAllDimensions Method

Clears all selected Dimension objects in the dimension line that can be filtered when a report is opened in the IBM Cognos portal.

Syntax

DeploymentOptions.**UnselectAllDimensions**

Discussion

Use this property when a report author specifies that a report consumer can filter a report opened in the IBM Cognos portal. The user can filter unnecessary information so that only the required information appears in the report.

This method clears all selected Dimension objects in the dimension line for the DeploymentOptions object so that the user can use the PromptForDimension property to select specific dimensions or to remove filtering on all dimensions.

Return Type

Nothing

Example

This example clears all selected dimensions from the dimension line filter and then specifies the dimensions that a user can filter in a report.

```
Sub Main()
   Dim objPPRep as Object
   Dim objDeploymentOptions as Object
   Set objPPRep = GetObject(,"CognosPowerPlay.Report")
   Set objDeploymentOptions = objPPRep.DeploymentOptions
   objDeploymentOptions.PromptForCurrency = True
   objDeploymentOptions.PromptForLongShortNames = True
   objDeploymentOptions.PromptForZeroSuppression = True
   objDeploymentOptions.PromptForSwapRowsAndColumns =
True
   objDeploymentOptions.UnselectAllDimensions
   objDeploymentOptions.Dimension(1) = True
   objDeploymentOptions.Dimension(2) = True
   objPPRep.Save
   Set objDeploymentOptions = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "SelectAllDimensions Method" on page 221

UnselectBlank Method

Unselects a specific blank row or column.

Syntax

object.**UnselectBlank BlankNumber**

Applies To

Column Object

Row Object

Discussion

Use UnselectBlank in conjunction with the Item and ItemAtLevel methods in crosstab reports. When a crosstab has blank rows or columns that have been selected by the SelectBlank method, UnselectBlank turns off the selection of the blank item.

Parameter	Description
BlankNumber	Required. Specifies the value indicating a blank item. BlankNumber acts as an index, where the first blank row or column is 1, the second is 2, and so on. Type: Long

Return Type

Nothing

Example

This example uses the AddBlanks method to add a blank row before the last row and a blank column before the last column in the active report.

```
Sub Main()
   Dim objPPRep As Object
   Dim intRow As Integer
   Dim intColumn As Integer
   Set objPPRep = GetObject(,"CognosPowerPlay.Report")
   objPPRep.ExplorerMode = False
   intRow = objPPRep.Rows.Count - 1
   intColumn = objPPRep.Columns.Count - 1
   objPPRep.Rows.Item(intRow).Select
   objPPRep.AddBlanks
   objPPRep.Rows.Unselect
   objPPRep.Columns.Item(intColumn).Select
   objPPRep.AddBlanks
   objPPRep.Columns.Unselect
   objPPRep.Rows.ItemAtLevel(intRow,0).SelectBlank(1)
   objPPRep.Columns.ItemAtLevel(intColumn,0).SelectBlank(1)
   Msgbox " A blank row and column have been added "
& _
      "and selected.",64,"Blanks"
   objPPRep.Rows.ItemAtLevel(intRow,0).UnselectBlank(1)
   objPPRep.Columns.ItemAtLevel(intColumn, _
      0).UnselectBlank(1)
```

```
   Msgbox " The blank row and column have now been "
& _
      "unselected.",64,"Blanks"
   Set objPPRep = Nothing
End Sub
```

UpdatePublishedReport Method

Updates a report previously published to the IBM Cognos Analytics content store.

Syntax

Report.**UpdatePublishedReport** [ReportDescription]

Applies To

Report Object

Discussion

Use this method to update a report that was previously published to the IBM Cognos Analytics content store using the PublishToPortal method.

Parameters	Description
ReportDescription	Optional. Specifies a description of the saved report. Type: String

Return Type

Boolean

Example

```
Sub Main()
     Dim objPPApp As Object
     Dim objPPRepRemote As Object
     Set objPPRepRemote = CreateObject("CognosPowerPlay.Report"
     Set objPPApp = objPPRepRemote.Application
     objPPApp.Visible = True
     objPPRepRemote.Open "C:\PPlay.ppx"
' Modify report.....
     objPPRepRemote.UpdatePublishedReport
     objPPRepRemote.Close
     set objPPRepRemote = Nothing
End Sub
```

Related Topics

- "PublishToPortal Method" on page 201

ValueRestriction Method

Returns the value restriction for an AdvancedQuery.

Syntax

AdvancedQuery.**ValueRestriction**(ValueName)

Applies To

AdvancedQuery Object

Discussion

Use this method to specify the name of the value restriction to filter the results of an AdvancedQuery. The ValueRestriction may include only the categories whose values are:

- the n largest or smallest values (specify the Largest or Smallest Operator and the Count property)
- greater than, less than, greater than or equal to, less than or equal to, or equal to a specified value (specify the greater than, less than, greater than or equal to, less than or equal to, or equal to Operator and Operand1.
- within a specified range (specify the Between Operator and both Operand1 and Operand2).

Specify only one value restriction at a time. For example, "Revenue> 10,000 and Units Sold < 30" is not valid.

Do not include functions applied to measures for value restrictions. For example, "Average(Revenue)>500" and "Revenue - Cost in Top 10" are not valid.

Value restrictions apply only to the lowest level categories of the AdvancedQuery results. For example, the result of an AdvancedQuery contains numerous Years and Quarters. When applying a value restriction to select only the top 10 values, the resulting AdvancedQuery contains the top 10 Quarters (the lowest level categories) and their corresponding years. The resulting report has 10 rows (for the top 10 values), but the query may return more than 10 categories.

Parameter	Description
ValueName	Required. Specifies the name assigned to the ValueRestriction object for the value restriction query. Type: String

Return Type

Nothing

Example

This example creates an advanced subset that selects countries or regions from the Locations dimension. The value restriction (type 4) limits the results to return only those countries or regions whose Revenue values for Environmental Line are between 25,000 and 100,000.

```
Sub Main()
   Dim strCubePath As String
   Dim objPPRep As Object
   Dim objValue As Object
   Dim objAdvanced As Object
   strCubePath = "C:\Cubes and Reports\Great Outdoors.mdc"
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.New strCubePath, 1
   objPPRep.ExplorerMode = False
   objPPRep.Visible = True
   Set objValue = objPPRep.ReportQueries.Add(4)
   With objValue
      .Name = "25000-100000"
      .Dimension = "Locations"
      .Measure = "Revenue"
      .Operator = "between"
      .Operand1 = 25000
      .Operand2 = 100000
      .DimensionFilter 2, "Environmental Line"
   End With
   Set objAdvanced = objPPRep.ReportQueries.Add(3)
   With objAdvanced
      .Name = "Locations"
      .Dimension = "Locations"
      .Level "Country or Region"
      .ValueRestriction objValue.Name
      .Execute
      .AddToReport 0,1,3
   End With
   Set objAdvanced = Nothing
   Set objValue = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- Chapter 4, “Methods,” on page 71
- Chapter 5, “Properties,” on page 259

Vertical Method

Returns whether the Graph object is a vertical display.

Syntax

Graph.**Vertical**

Applies To

Graph Object

Discussion

To set this method, use the Add method for Graph collections, or the SetType method for Graph objects.

Graph type 3 (bar chart) can be either vertical or horizontal in a 2D or 3D graph. This means that graph type 3 can return True (when it is vertical) or False (when it is horizontal).

A crosstab will always return False.

Default: True

Return Type

Boolean

Example

This example changes the display type for the first Graph object to a three-dimensional cluster bar, and displays the settings for the Graph object for an open report.

```
Sub Main()
   Dim objPPRep as Object
   Dim objPPGph as Object
   Set objPPRep = GetObject( , "CognosPowerPlay.Report")
   Set objPPGph = objPPRep.Graphs.Item(1)
   objPPGph.SetType 4, 1, 1
   MsgBox "The Graph object type is " & objPPGph.Type
& "."
   If objPPGph.Depth = -1 Then
      MsgBox "The graph is not 3D."
   Else
       MsgBox "The graph is  3D."
   End If
   If objPPGph.Vertical = -1 Then
      MsgBox "The graph is oriented vertically."
   Else
      MsgBox "The graph is oriented horizontally."
   End If
   Set objPPGph = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- Chapter 4, “Methods,” on page 71
- Chapter 5, “Properties,” on page 259
- “Graph Object” on page 25

Chapter 5. Properties

You work with the following properties for IBM Cognos PowerPlay OLE automation.

Name	Description
Application Property	Returns the Application object.
AutomaticExceptions Property	Sets or returns whether automatic highlighting of exceptions is on or off.
AutomaticExceptionSensitivity Property	Sets or returns the automatic exception sensitivity.
Average Property	Sets or returns whether to calculate the average of the selected categories in the CategoryList object.
AxisOnAllPages Property	Sets or returns whether the axis and labels appear on every page of the printed report or PDF.
BlankWhenDividedByZero Property	Sets or returns whether a numeric value divided by zero appears as zero or blanks.
BlankWhenMissing Property	Sets or returns whether missing numeric values appear as zero or blanks.
BlankWhenZero Property	Sets or returns whether zero numeric values appear as zeros or blanks.
CalculatedCategories Property	Sets or returns whether calculated categories are on or off, or whether a PowerCube contains calculated categories.
Caption Property	Returns the title of the Application object window.
CellText Property	Returns the text in a cell.
CellValueAlignment Property	Returns the alignment applied to a cell value in a report.
CellValueFontColor Property	Returns the font color applied to a cell value in a report.
CellValueFontName Property	Returns the name of the font applied to a cell value in a report.
CellValueFontSize Property	Returns the size of the font applied to a cell value in a report.

Name	Description
ChartTitleOnAllPages Property	Sets or returns whether titles appear on every page of the printed report or PDF.
Collate Property	Sets or returns whether the Report object collates during printing.
Copies Property	Sets or returns the number of copies to print.
Count Property	Returns the number of categories one level below the current category in the Dimension object, or the number objects in a collection.
CubeName Property	Returns the file name of the cube for the active report.
DataGridlines Property	Sets or returns whether the gridlines settings are on or off for a crosstab.
DefaultAlternateDirectory Property	Sets or returns the directory to save updates to a read-only report.
DefaultCubeDirectory Property	Sets or returns the default path for multi-dimensional cube files (.mdc).
DefaultMacroDirectory Property	Sets or returns the default path for macro files.
DefaultReportDirectory Property	Sets or returns the default path for PowerPlay report files.
Dimension Property	Sets or returns the dimension from which categories are returned.
DimensionLineIndex Property	Returns the position of a dimension line item to maintain a list of Layer, Row, and Column objects.
DimensionSettings Property	Returns a comma-separated string of all the dimension line settings for the ValueRestriction.
DrivingCategory Property	Returns the driving category for the Exception object.
DrivingDimension Property	Returns the driving dimension for the Exception object.
Each Property	Sets or returns whether all selected and new categories or just the new categories appear in the report.

Name	Description
EnableUserColumnSummaryLabel Property	Sets or returns whether a user-defined label is used for the innermost summary column in a nested crosstab.
EnableUserRowSummaryLabel Property	Sets or returns whether a user-defined label is used for the innermost summary row in a nested crosstab.
Exception Property	Sets or returns the exception for one or more categories.
ExplorerMode Property	Sets or returns whether the Report object is an Explorer or Reporter report.
FitToPage Property	Sets or returns whether the report is scaled to fit on one page.
FooterText Property	Sets or returns the text in the footer of a report.
FullName Property	Returns the full name, including the location, of either the Application object or the Report object.
GetDataAutomatically Property	Sets or returns whether the Report object retrieves data automatically each time it is modified.
HeaderText Property	Sets or returns the text in the header of a report.
HideRankCategory Property	Sets or returns whether the rank category is hidden.
IncludeLegend Property	Sets or returns whether the legend appears in a printed report or PDF.
IndentTotalsLevel Property	Sets or returns the current indent level for summary cells in a nested crosstab.
Index Property	Returns the position of an object in a collection.
Intersect Property	Sets or returns whether to determine the values at the intersection of selected categories from different dimensions.
IsAlternate Property	Returns whether the drill-down path is primary or alternate.
IsCalculatedCategory Property	Returns whether the category is a calculated category.

Name	Description
KeepSummaryVisible Property	Sets or returns whether the summary category will remain visible on all scrolled pages.
LabelAlignment Property	Returns the alignment applied to a cell label in a report.
LabelFontColor Property	Returns the font color applied to a category label in a report.
LabelFontName Property	Returns the name of the font applied to a category label in a report.
LabelFontSize Property	Returns the size of the font applied to a category label in a report.
LabelGridlines Property	Sets or returns whether the gridlines are on or off for category labels in a nested crosstab.
Layout Property	Sets or returns the current layout style in a nested crosstab.
Level Property	Returns the level of the category in a dimension.
LevelList Property	Returns the list of levels for a specified drill-down path.
LevelsDown Property	Sets the number of levels down the hierarchy for specifying the next level ParentageQuery subsets.
LogonPrompt Property	Sets or returns whether the application prompts for logon or security information.
LowerBoundary Property	Sets or returns the value defined for the lower boundary of the Range object.
LowestLevel Property	Sets whether the query uses the next lower level or lowest level of the parent category.
MacroName Property	Sets or returns the name of the macro associated with an Exception object.
MacroStyle Property	Sets or returns the name of the style associated with the macro used by an Exception object.
MaximumNumberOfRanges Property	Returns the maximum number of Range objects definable for an Exception object.

Name	Description
MaxPrintedBars Property	Sets or returns the maximum number of bars on a single printed page.
MaxVisibleBars Property	Sets or returns the maximum number of bars visible on a single page of scrolled data.
Measure Property	Sets or returns the name of measure whose values are used for a value restriction.
MeasureCurrency Property	Sets or returns the value and symbol for a specified currency.
Name Property	Sets or returns name of the object.
NamesShown Property	Sets or returns whether category names appear beside pie chart slices.
NestedCharts Property (Explorer)	Sets or returns whether multiple charts that represent summarized data appear in one display.
NestedName Property	Returns the nested name for a category.
Operand1 Property	Sets or returns the value used to compare report cell values based on a specified operator.
Operand2 Property	Sets or returns the second value when the Between operator is used to specify a range.
Operator Property	Sets or returns the type of operator used for a value restriction.
ParentCategory Property	Returns the name of the parent category for the object.
Path Property	Returns the path of the Report object or the Application object.
Pattern Property	Sets search criteria for a subset definition.
Precedence Property	Sets or returns the precedence used in complex calculations.
PrintAllCharts Property	Sets or returns whether all displays print on the same page.
PrintColorsAsPatterns Property	Sets or returns whether colors print as patterns or as colors.
PrintEntireReport Property	Sets or returns whether to print the entire report, including all displays, layers, and rows.

Name	Description
PrintPageLayout Property	Sets or returns whether to print all displays visible in the page layout or page width view on the same page.
PrintSelectedDisplay Property	Sets or returns whether to print the selected or currently active Graph object.
PromptForCurrency Property	Sets or returns whether the report consumer can change the currency in a report published to the IBM Cognos portal.
PromptForDimension Property	Sets or returns whether a report consumer can filter the specified Dimension object when a report is opened in the IBM Cognos portal.
PromptForLongShortNames Property	Sets or returns whether the report consumer can change between long and short category names in a report published to the IBM Cognos portal.
PromptForSwapRowsAndColumns Property	Sets or returns whether the report consumer can swap rows and columns in a report published to the IBM Cognos portal.
PromptForZeroSuppression Property	Sets or returns whether the report consumer can apply or turn off zero suppression in a report published to the IBM Cognos portal.
RefreshSubCube Property	Sets or returns whether the sub-cube is refreshed automatically.
SaveAllCharts Property	Sets or returns whether all Graph objects are saved in a PDF.
Saved Property	Returns whether the Report object has been saved.
SaveEntireReport Property	Sets or returns whether to save the entire report as a PDF.
SearchDescription Property	Sets or returns whether the FindQuery object searches the category descriptions in a cube.
SearchShortName Property	Sets or returns whether the FindQuery object searches short or long category names.
SearchText Property	Sets or returns the search string used in the subset definition of a FindQuery query.
ShareDimensionLine Property	Sets or returns whether open reports share a dimension line.

Name	Description
ShareOf Property	Sets or returns whether to show the values in selected categories as a percentage of the higher-level category.
ShowSummaryBreakdown Property (Explorer)	Sets or returns whether to show the breakdown of summary rows and columns in a crosstab.
ShowSummaryColumn Property (Explorer)	Sets or returns whether to show the summary column.
ShowSummaryRow Property (Explorer)	Sets or returns whether to show the summary row.
ShowTies Property	Sets or returns whether to show label ties.
ShowValuesAs Property (Explorer)	Sets or returns how to show values in a report.
StatsLineCaption Property	Sets or returns the caption for a given statistical line on a graph.
StatsLineColor Property	Sets or returns the color for a given statistical line on a graph.
StatsLineOn Property	Sets or returns a statistical line on a graph.
StatsLineStyle Property	Sets or returns the line style of a given statistical line on a graph.
StatsLineUserValue Property	Sets a custom value for a statistical line on a graph.
Style Property	Sets or returns the style used for a category, an exception range or set of categories.
Sum Property	Sets or returns whether to calculate the sum of selected categories.
SummariesOnAllPages Property	Sets or returns whether existing summaries appear on every page of a printed report.
SummaryColumnOnAllPages Property	Sets or returns whether to show the summary column category on every page of a report or PDF.
SummaryRowOnAllPages Property	Sets or returns whether to show the summary row category on every page of a report or PDF.
Suppress8020 Property (Explorer)	Sets or returns the 80/20 suppress mode for report dimensions.

Name	Description
SuppressZeros Property	Sets or returns the suppress mode for the Report object.
Threshold Property	Sets or returns the maximum printing page limit for the Print object.
TitleText Property	Sets or returns the text in the title of a report.
TopLevelCategory Property (Explorer)	Returns the name of the dimension for the object.
TopLevelParentCategory Property	Returns the name of the dimension for the object.
Type Property	Returns the object type.
UpperBoundary Property	Sets or returns the value defined for the upper boundary of the Range object.
UseFontSubstitution Property	Sets or returns whether to save full font information in a report published to the IBM Cognos portal.
UserControl Property	Sets or returns whether the Application object is under user control.
UserColumnSummaryLabel Property	Sets or returns the user-defined label for the innermost summary column in a nested crosstab.
UserRowSummaryLabel Property	Sets or returns the user-defined label for the innermost summary row in a nested crosstab.
UseScrolling Property	Sets or returns whether scrolling is enabled.
ValuesAutoFit Property	Sets or returns whether value labels fit within graph bars and pie segments.
ValuesFontColor Property	Sets or returns the font color used for the graph value labels.
ValuesFontName Property	Sets or returns the font name used for the value labels.
ValuesFontSize Property	Sets or returns the font size used for the graph value labels.
ValuesFontStyle Property	Sets or returns the font style used for the graph value labels.

Name	Description
ValuesPosition Property	Sets or returns the position of value labels on some graph types.
ValuesShown Property	Sets or returns whether value labels appear next to pie chart slices.
Version Property	Returns the version number of PowerPlay.
Visible Property	Sets or returns whether the object is visible to the user.

Application Property

Returns the Application object.

Syntax

object.**Application**

Applies To

AdvancedQuery Object

Application Object

CategoryList Object

Column Object

Columns

Dimension Object

DimensionLine Object

Exception Object

Exceptions

FindQuery Object

Graph Object

Graphs

Layer Object

Layers

"Level Object" on page 30

Levels

ParentageQuery Object

Print Object

Range Object

Ranges

Report Object

ReportQueries

Reports

Row Object

Rows

SaveAsPDF Object

ValueRestriction Object

Discussion

Use this property to gain access to application level settings such as default directories or the current version of the product. It can also be used to place the application on top of all the others, that is make it the active one.

Type

Object

Access

Read

Example

This example returns the name and location of the open report. Then, using the Application property, it brings IBM Cognos PowerPlay to the front and makes it active.

```
Sub Main()
   Dim objRep As Object
   Set objRep = GetObject( ,"CognosPowerPlay.Report")
   MsgBox "The name of the current report is " &objRep.Name
   MsgBox "The location of the current report is " &objRep.Path
   If objRep.Saved = False Then
      objRep.Save
      MsgBox "Changes to the report have been saved."
   Else
      MsgBox "No changes have been made to the report."
   End If
   objRep.Application.Activate
   Set objRep = Nothing
```

```
End Sub
```

Related Topics

- Chapter 4, “Methods,” on page 71
- Chapter 5, “Properties,” on page 259
- “Application Object” on page 11

AutomaticExceptions Property

Sets or returns whether automatic highlighting of exceptions is on or off.

Syntax

Report.**AutomaticExceptions**

Applies To

Report Object

Discussion

Use this property to set whether exceptions are automatically highlighted. Automatic exceptions are data driven exceptions (otherwise known as anomalies or outliers). Whenever the crosstab cell values change in a report, the expected value of each cell, based on the row, column, and crosstab totals, is computed and compared to the observed value (the value in the cell). If the difference is significant, the cell value is highlighted as a high (green) or low (red) exception.

This property is only valid when IBM Cognos PowerPlay is in Explorer mode. Otherwise, a message displays indicating that this operation is not available.

Do not use automatic exception highlighting when the row or column is a time dimension.

Default: False

Type

Boolean

Access

Read/Write

Example

This example changes the current setting of the AutomaticExceptions property of the active report. For example, if it is True, it changes to False.

```
Sub Main()
   Dim objPPRep As Object
   Set objPPRep = GetObject(,"CognosPowerPlay.Report")
   If objPPRep.AutomaticExceptions = True Then
      objPPRep.AutomaticExceptions = False
```

```
      Msgbox "Automatic Exceptions highlighting " &
_
         "has been turned off." , , "Automatic Exceptions"
   Else
      objPPRep.AutomaticExceptions = True
      Msgbox "Automatic Exceptions highlighting " &
_
         "has been turned on." , , "Automatic Exceptions"
   End If
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "AutomaticExceptionSensitivity Property"

AutomaticExceptionSensitivity Property

Sets or returns the automatic exception sensitivity.

Syntax

Report.**AutomaticExceptionSensitivity**

Applies To

Report Object

Discussion

Use this property to identify the sensitivity of data driven exceptions (otherwise known as anomalies or outliers). Every cell value within a report that is within the valid sensitivity range for automatic exceptions (between 1 and 10) will be highlighted. For OLE automation, there are two default highlight styles, red and green. Exceptions that are above the mean of the deviation are automatically highlighted in green; those that are below are highlighted in red. If the user has changed the default automatic exception style, their modified style is used to highlight these exceptions.

Type

Integer

Access

Read/Write

Example

This example requires input from the user to change the AutomaticExceptionsSensitivity property of the active report. The user specifies a value between 1 and 10. Every cell value within the report that is within the this sensitivity range will be highlighted, based on the default automatic exception style.

```
Sub Main()
```

```
   Dim objPPRep As Object
   Dim strValue As String
   Set objPPRep = GetObject(,"CognosPowerPlay.Report")
   strValue = InputBox("Adjust the Automatic " &
_
      "Exception Sensitivity.  Enter a value between
" & _
      "1 and 10.", "Automatic Exception Sensitivity",
10)
   objPPRep.AutomaticExceptionSensitivity = strValue
   Msgbox "The Automatic Exception Sensitivity " &
_
      "has been reset to " & _
      objPPRep.AutomaticExceptionSensitivity & "."
_
      , , "Automatic Exceptions"
   Set objPPRep = Nothing
End Sub
```

Average Property

Sets or returns whether to calculate the average of the selected categories in the CategoryList object.

Syntax

CategoryList.**Average**

Applies To

CategoryList Object

Discussion

To help plan future or current events, you can use this property to calculate an average. If True, a new category is created in the report which shows the average of all the selected categories in the CategoryList object. For example, you can calculate the average revenue for the first quarter of the year or you can calculate the average profit margin for all products in order to determine if a product is above or below the average.

Default: False

Type

Boolean

Access

Read/Write

Example

This example adds categories from the Go Sports Line and the second quarter of 1996 to a new report, and shows the average for the products in Go Sports Line.

```
Sub Main
   Dim objPPRep as Object
   Dim objCatList as Object
   Const LARGE_NUMBER = 999
   Set objPPRep = CreateObject( "CognosPowerPlay.Report"
)
   objPPRep.New "C:\Cubes and Reports\Great Outdoors.mdc"
   objPPRep.Visible = True
   Set objCatList = objPPRep.CategoryList
   objCatList.Add LARGE_NUMBER, "Products", "Go Sport
Line"
   objCatList.Each = True
   objCatList.Average = True
   objPPRep.Columns.Add objCatList
   objCatList.Remove
   objCatList.Add 1, "Years", "1996", "1996 Q 2"
   objCatList.Each = True
   objCatList.Average = False
   objPPRep.Rows.Add objCatList
   Set objCatList = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- Chapter 4, "Methods," on page 71
- Chapter 5, "Properties," on page 259
- "CategoryList Object" on page 13

AxisOnAllPages Property

Sets or returns whether the axis and labels appear on every page of the printed report or PDF.

Syntax

object.**AxisOnAllPages**

Applies To

Print Object

SaveAsPDF Object

Discussion

Use this property when you want to save the axes and labels on every page of the PDF, or to print a report that shows the axes, labels, and legend on every printed page.

When printing a report, the PrintAllCharts property must be set to False to use this property.

When saving a report as a PDF file, to use this property, the SaveAllCharts property must be set to False, and the SaveEntireReport must be set to True.

If this property is True when you print the report or save the report as a PDF, the axes and labels appear on every page of the report. If False, the axes and labels appear only once.

Default: False

Type

Boolean

Access

Read/Write

Example

This example opens a report and prints one copy of all the data for the second graphical display only. This example includes the display title, summary category, and axis on all pages, turns on the collating option, and includes the legend.

```
Sub Main()
   Dim objPPRep as Object
   Dim objRepPrt as Object
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.Open "C:\Cubes and Reports\Sample2.ppx"
   Set objRepPrt = objPPRep.Print
   objRepPrt.PrintAllCharts = False
   objRepPrt.SetListOfRowsToPrint objPPrep.Rows
   objRepPrt.SetListOfLayersToPrint objPPRep.Layers
   objRepPrt.SetChartToPrint objPPRep.Graphs.Item(2)
   objRepPrt.IncludeLegend = True
   objRepPrt.ChartTitleOnAllPages = True
   objRepPrt.SummariesOnAllPages = True
   objRepPrt.AxisOnAllPages = True
   objRepPrt.Collate = True
   objRepPrt.Copies = 1
   objRepPrt.PrintOut
   Set objRepPrt = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "PrintAllCharts Property" on page 364
- "SaveAllCharts Property" on page 377
- "SaveEntireReport Property" on page 379

BlankWhenDividedByZero Property

Sets or returns whether a numeric value divided by zero appears as zero or blanks.

Syntax

Dimension.**BlankWhenDividedByZero**

Applies To

Dimension Object

Discussion

Set this property to True when you want the result of zero division to show in a Dimension object as blank spaces rather than a zero (0). Blank spaces are only meaningful in nested crosstabs; not in other graph types.

This property is only available for measures.

Default: False

Type

Boolean

Access

Read/Write

Example

This example changes the Measure to Product Cost and then sets the Financial Formatting properties.

```
Sub Main()
   Dim objPPRep As Object
   Dim objDimension as object
   Set objPPRep = GetObject(,"CognosPowerPlay.Report")
   Set objDimension = objPPRep.DimensionLine.Item("Measures")
   objDimension.ChangeToTop
   objDimension.Change "Product Cost"
   objDimension.BlankWhenZero = True
   objDimension.BlankWhenMissing = False
   objDimension.BlankWhenDividedByZero = False
   Set objDimension = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "BlankWhenMissing Property"
- "BlankWhenZero Property" on page 275

BlankWhenMissing Property

Sets or returns whether missing numeric values appear as zero or blanks.

Syntax

Dimension.**BlankWhenMissing**

Applies To

Dimension Object

Discussion

Set this property to True when you want missing values in a Dimension object to show as blank spaces rather than an NA or zero. Blank spaces are only meaningful in nested crosstabs; not in other graph types.

This property is only available for measures.

Default: False

Type

Boolean

Access

Read/Write

Example

This example changes the Measure to Product Cost and then sets the Financial Formatting properties.

```
Sub Main()
   Dim objPPRep As Object
   Dim objDimension as object
   Set objPPRep = GetObject(,"CognosPowerPlay.Report")
   Set objDimension = objPPRep.DimensionLine.Item("Measures")
   objDimension.ChangeToTop
   objDimension.Change "Product Cost"
   objDimension.BlankWhenZero = True
   objDimension.BlankWhenMissing = False
   objDimension.BlankWhenDividedByZero = False
   Set objDimension = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "BlankWhenDividedByZero Property" on page 273
- "BlankWhenZero Property"

BlankWhenZero Property

Sets or returns whether zero numeric values appear as zeros or blanks.

Syntax

Dimension.**BlankWhenZero**

Applies To

Dimension Object

Discussion

Set this property to True when you want zero values in a Dimension object to be shown as blank spaces rather than a zero. Blank spaces are only meaningful in nested crosstabs, not in other graph types.

This property is only available for measures.

Default: False

Type

Boolean

Access

Read/Write

Example

This example changes the Measure to Product Cost and then sets the Financial Formatting properties.

```
Sub Main()
   Dim objPPRep As Object
   Dim objDimension as object
   Set objPPRep = GetObject(,"CognosPowerPlay.Report")
   Set objDimension = objPPRep.DimensionLine.Item("Measures")
   objDimension.ChangeToTop
   objDimension.Change "Product Cost"
   objDimension.BlankWhenZero = True
   objDimension.BlankWhenMissing = False
   objDimension.BlankWhenDividedByZero = False
   Set objDimension = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "BlankWhenDividedByZero Property" on page 273
- "BlankWhenMissing Property" on page 274

CalculatedCategories Property

Sets or returns whether calculated categories are on or off, or whether a PowerCube contains calculated categories.

Syntax

Report.**CalculatedCategories**

Applies To

Report Object

Discussion

When a cube contains calculated categories, this property returns True; otherwise, it returns False. If the cube has calculated categories, they can be turned off by setting this property to False, or turned on by setting this property to True. If a cube contains no calculated categories and you attempt to set the property to True, the following error message is returned: "The cube has no calculated categories."

Default: True (for new reports).

Type

Boolean

Access

Read/Write when a cube has calculated categories.

Read when a cube doesn't have calculated categories.

Example

This example creates a new report, turns calculated categories off, adds categories to the report, then turns the calculated categories on again.

```
Sub Main()
   Dim objPPRep As Object
   Dim objCategoryList As Object
   Const CubePath = "C:\Cubes and Reports\Great Outdoors.mdc"
   Set objPPRep = GetObject(,"CognosPowerPlay.Report")
   If objPPRep.CalculatedCategories = True Then
      objPPRep.CalculatedCategories = False
   End If
   Set objCategoryList = objPPRep.CategoryList()
   objCategoryList.Add 2, "Products", "GO Sport Line"
   objPPRep.Rows.Add objCategoryList
   objCategoryList.Add 1, "Years"
   objPPRep.Columns.Add objCategoryList
   objPPRep.CalculatedCategories = True
   objPPRep.SaveAs "C:\Calculated Categories.ppx"
   Set objCategoryList = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "CategoryList Object" on page 13

Caption Property

Returns the title of the Application object window.

Syntax

Application.**Caption**

Applies To

Application Object

Discussion

Use this property to determine the title of the application in use.

Type

String

Access

Read

Example

This example creates an instance of the IBM Cognos PowerPlay Application object and returns the name, the location and the version of the application and the title of the application window.

```
Sub Main()
   Dim objPPlayApp as Object
   Set objPPlayApp = CreateObject("CognosPowerPlay.Application")
   objPPlayApp.Visible = 1
   MsgBox "The title of the application is " &objPPlayApp.Caption
   MsgBox "The name of the Application is " &objPPLayApp.Name
   MsgBox "The location of the Application is " _
      &objPPLayApp.Path
   MsgBox "The Application version is " &objPPLayApp.Version
   Set objPPlayApp = Nothing
End Sub
```

CellText Property

Returns the text in a cell.

Syntax

object.**CellText**(Index)

Applies To

Column Object

Row Object

Discussion

Use this property to return a calculated result, rank position, or a value in specific location of a report. You can use this property with

- CellValueAlignment
- CellValueFontColor
- CellValueFontName
- CellValueFontSize

to determine if there is special formatting applied to the cell.

This property returns text that is in rows and columns that are not hidden in the report.

Parameter	Description
Index	Required. Specifies the row or column number of the cell. Use the row number when the property applies to a Column object. Use the column number when the property applies to a Row object. The index starts at 1 and increments by 1, from left to right, for each cell in the chart. Type: Integer

Return Type

String

Example

This example opens a report, and returns the text within the specified cells.

```
Sub Main()
   Dim objPPRep As Object
   Set objPPRep = GetObject("C:\Cubes and Reports\Sample.ppx")
   objPPRep.Visible = True
   MsgBox "The text in the fourth cell of the second
row is " _
      &objPPRep.Rows.Item(2).CellText(4)
   MsgBox "The text in the third cell of the third column
is " _
      &objPPRep.Columns.Item(3).CellText(3)
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "ShowValuesAs Property (Explorer)" on page 391

CellValueAlignment Property

Returns the alignment applied to a cell value in a report.

Syntax

object.**CellValueAlignment**(Index)

Applies To

Column Object

Row Object

Discussion

Use this property to determine the alignment for a row or column cell value. A report author may also choose to use different cell alignments depending on the measure format, including currency type, commas, or percentage signs.

Because the alignment of cell values can vary throughout a report, you must specify an index to refer to a column or row cell.

This property applies to values only. Values can be measures or calculations.

An error occurs if the index reference for this property is out of bounds.

Valid alignment values are

0 = left aligned 1 = center aligned 2 = right aligned 3 = default

Default: 3 (report setting, if no alignment set for cell)

Parameters	Description
Index	Required. Specifies the row or column number of the cell. Use the row number when the property applies to a Column object. Use the column number when the property applies to a Row object. Type: Integer

Type

Long

Access

Read

Example

This example returns the alignment for the cell value in the second column, fourth row, and in the third column, third row.

```
Sub Main()
   Dim objPPRep as Object
   Set objPPRep = CreateObject ("CognosPowerPlay.Report")
```

```
   objPPRep.Open "C:\Cubes and Reports\Sample1.ppx"
   MsgBox "The alignment for this cell value is " _
      &objPPRep.Rows.Item(2).CellValueAlignment(4)
   MsgBox "The alignment for this cell value is " _
      &objPPRep.Columns.Item(3).CellValueAlignment(3)
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "CellValueFontColor Property"
- "CellValueFontName Property" on page 283
- "CellValueFontSize Property" on page 284

CellValueFontColor Property

Returns the font color applied to a cell value in a report.

Syntax

object.**CellValueFontColor**(Index)

Applies To

Column Object

Row Object

Discussion

Use this property to determine the font color for a row or column cell value, especially for a crosstab that uses multiple colors to highlight specific measures or categories. A report author may choose to apply different colors to each range in an exception definition.

Because the font color of cell values can vary throughout a report, you must specify an index to refer to a column or row cell.

This property applies to values only. Values can be measures or calculations.

An error occurs if the index reference for this property is out of bounds.

Valid font colors are

0 = Black

128 = Brown

32768 = Green

32896 = Olive

8388608 = Navy

8388736 = Purple

8421376 = Teal

8421504 = Gray

12632256 = Silver

255 = Red

65280 = Lime

65535 = Yellow

16711680 = Blue

16711935 = Fuschia

16776960= Aqua

16777215 = White

Default: 0 (black)

Parameters	Description
Index	Required. Specifies the row or column number of the cell. Use the row number when the property applies to a Column object. Use the column number when the property applies to a Row object. Type: Integer

Type

Long

Access

Read

Example

This example returns the font color for the cell value in the second column, fourth row, and in the third column, third row.

```
Sub Main()
   Dim objPPRep as Object
   Set objPPRep = CreateObject ("CognosPowerPlay.Report")
   objPPRep.Open "C:\Cubes and Reports\Sample1.ppx"
   MsgBox "The font color for this cell value is " _
      &objPPRep.Rows.Item(2).CellValueFontColor(4)
   MsgBox "The font color for this cell value is " _
      &objPPRep.Columns.Item(3).CellValueFontColor(3)
   Set objPPRep = Nothing
```

```
End Sub
```

Related Topics

- "CellValueAlignment Property" on page 279
- "CellValueFontName Property"
- "CellValueFontSize Property" on page 284

CellValueFontName Property

Returns the name of the font applied to a cell value in a report.

Syntax

object.**CellValueFontName(Index)**

Applies To

Column Object

Row Object

Discussion

Use this property to determine the name of a font used for a row or column cell value, especially for a crosstab that uses multiple fonts to highlight specific measures or categories. A report author may choose to apply different fonts to each range in an exception definition.

Because the font applied to cell values can vary throughout a report, you must specify an index to refer to a column or row cell.

This property applies to values only. Values can be measures or calculations.

An error occurs if the index reference for this property is out of bounds.

Default: Arial

Parameters	Description
Index	Required. Specifies the row or column number of the cell. Use the row number when the property applies to a Column object. Use the column number when the property applies to a Row object. Type: Integer

Type

String

Access

Read

Example

This example returns the name of the font applied to the cell value in the second column, fourth row, and in the third column, third row.

```
Sub Main()
   Dim objPPRep as Object
   Set objPPRep = CreateObject ("CognosPowerPlay.Report")
   objPPRep.Open "C:\Cubes and Reports\Sample1.ppx"
   MsgBox "The name of the font for this cell value is
" _
      &objPPRep.Rows.Item(2).CellValueFontName(4)
   MsgBox "The name of the font for this cell value is
" _
      &objPPRep.Columns.Item(3).CellValueFontName(3)
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "CellValueAlignment Property" on page 279
- "CellValueFontColor Property" on page 281
- "CellValueFontSize Property"

CellValueFontSize Property

Returns the size of the font applied to a cell value in a report.

Syntax

object.**CellValueFontSize**(Index)

Applies To

Column Object

Row Object

Discussion

Use this property to determine the size of a font used for a row or column cell value, especially for a crosstab that uses multiple sizes to highlight specific measures or categories. A report author may choose to apply different font sizes to each range in an exception definition.

Because the font sizes applied to cell values can vary throughout a report, you must specify an index to refer to a column or row cell.

This property applies to values only. Values can be measures or calculations.

An error occurs if the index reference for this property is out of bounds.

Default: 10

Parameters	Description
Index	Required. Specifies the row or column number of the cell. Use the row number when the property applies to a Column object. Use the column number when the property applies to a Row object. Type: Integer

Type

Long

Access

Read

Example

This example returns the name of the font applied to the cell value in the second column, fourth row and in the third column, third row.

```
Sub Main()
   Dim objPPRep as Object
   Set objPPRep = CreateObject ("CognosPowerPlay.Report")
   objPPRep.Open "C:\Cubes and Reports\Sample1.ppx"
   MsgBox "The size of the font for this cell value is
" _
      &objPPRep.Rows.Item(2).CellValueFontSize(4)
   MsgBox "The name of the font for this cell value is
" _
      &objPPRep.Columns.Item(3).CellValueFontSize(3)
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "CellValueAlignment Property" on page 279
- "CellValueFontColor Property" on page 281
- "CellValueFontName Property" on page 283

ChartTitleOnAllPages Property

Sets or returns whether titles appear on every page of the printed report or PDF.

Syntax

object.**ChartTitleOnAllPages**

Applies To

Print Object

SaveAsPDF Object

Discussion

Use this property when you want to save the title for the chart on every page of the PDF, or to print a report that shows the chart title on every printed page.

When printing a report, the PrintAllCharts property must be set to False to use this property.

When saving a report as a PDF file, the SaveAllCharts property must be set to False to use this property. This property is ignored if you set this property and the SaveEntireReport property is set to True.

If this property is True and the chart has a title, when you print the report or save the report as a PDF, the title appears on every page of the report. If False, the title appears only once.

This method does not apply if your report contains only one crosstab display.

Default: False

Type

Boolean

Access

Read/Write

Example

This example opens a report and prints one copy of all the data for the second graphical display only. This example includes the display title, summary category, and axis on all pages, turns on the collating option, and includes the legend.

```
Sub Main()
   Dim objPPRep as Object
   Dim objRepPrt as Object
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.Open "C:\Cubes and Reports\Sample2.ppx"
   Set objRepPrt = objPPRep.Print
   objRepPrt.PrintAllCharts = False
   objRepPrt.SetListOfRowsToPrint objPPrep.Rows
   objRepPrt.SetListOfLayersToPrint objPPRep.Layers
   objRepPrt.SetChartToPrint objPPRep.Graphs.Item(2)
   objRepPrt.IncludeLegend = True
   objRepPrt.ChartTitleOnAllPages = True
   objRepPrt.SummariesOnAllPages = True
   objRepPrt.AxisOnAllPages = True
```

```
   objRepPrt.Collate = True
   objRepPrt.Copies = 1
   objRepPrt.PrintOut
   Set objRepPrt = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "PrintAllCharts Property" on page 364
- "SaveAllCharts Property" on page 377
- "SaveEntireReport Property" on page 379

Collate Property

Sets or returns whether the Report object collates during printing.

Syntax

Print.**Collate**

Applies To

Print Object

Discussion

If True, the report prints collated, if your print driver supports collating. If False, the report prints in a group (that is all copies of the same page at once followed by all copies of the next page.)

Default: False

Type

Boolean

Access

Read/Write

Example

This example opens a report and prints one copy of all the data for the second graphical display only. This example includes the display title, summary category, and axis on all pages; turns on the collating option; and excludes the legend.

```
Sub Main()
   Dim objPPRep as Object
   Dim objRepPrt as Object
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.Open "C:\Cubes and Reports\Sample2.ppx"
   Set objRepPrt = objPPRep.Print
   objRepPrt.PrintAllCharts = False
   objRepPrt.SetListOfRowsToPrint objPPrep.Rows
```

```
    objRepPrt.SetListOfLayersToPrint objPPRep.Layers
    objRepPrt.SetChartToPrint objPPRep.Graphs.Item(2)
    objRepPrt.IncludeLegend = True
    objRepPrt.ChartTitleOnAllPages = True
    objRepPrt.SummariesOnAllPages = True
    objRepPrt.AxisOnAllPages = True
    objRepPrt.Collate = True
    objRepPrt.Copies = 1
    objRepPrt.PrintOut
    Set objRepPrt = Nothing
    Set objPPRep = Nothing
End Sub
```

Related Topics

- Chapter 4, "Methods," on page 71
- Chapter 5, "Properties," on page 259
- "Print Method" on page 198
- "Print Object" on page 33
- "Report Object" on page 37

Copies Property

Sets or returns the number of copies to print.

Syntax

Print.**Copies**

Applies To

Print Object

Discussion

This property sets the number of copies of a report to print. If not specified, the default is one copy. A value of zero returns an error. This property is not related to the Threshold property.

Default: 1

Type

Integer

Access

Read/Write

Example

This example opens a report and prints one copy of all the data for the second graphical display only. This example includes the display title, summary category, and axis on all pages; turns on the collating option; and includes the legend.

```
Sub Main()
   Dim objPPRep as Object
   Dim objRepPrt as Object
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.Open "C:\Cubes and Reports\Sample2.ppx"
   Set objRepPrt = objPPRep.Print
   objRepPrt.PrintAllCharts = False
   objRepPrt.SetListOfRowsToPrint objPPrep.Rows
   objRepPrt.SetListOfLayersToPrint objPPRep.Layers
   objRepPrt.SetChartToPrint objPPRep.Graphs.Item(2)
   objRepPrt.IncludeLegend = True
   objRepPrt.ChartTitleOnAllPages = True
   objRepPrt.SummariesOnAllPages = True
   objRepPrt.AxisOnAllPages = True
   objRepPrt.Collate = True
   objRepPrt.Copies = 1
   objRepPrt.PrintOut
   Set objRepPrt = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- Chapter 4, “Methods,” on page 71
- Chapter 5, “Properties,” on page 259
- “Print Object” on page 33

Count Property

Returns

- the number of categories one level below the current category in the Dimension object
- the number of objects in a collection or an object that maintains a list of objects

Syntax

object.**Count**

Applies To

AdvancedQuery Object

CategoryList Object

Children

Columns

Dimension Object

DimensionLine Object

Exceptions

FindQuery Object

Graphs

Layers

ParentageQuery Object

Ranges

Reports

ReportQueries

Rows

Discussion

Use this property to determine the number of times to run a For...Next Loop. It improves the flexibility of the macro.

The Count property does not apply to the Average category.

Type

Long

Access

Read

Example

This example counts the number of categories one level below the Years category.

```
Sub Main()
   Dim objPPRep as Object
   Dim objPPDim as Object
   Set objPPRep = GetObject( ,"CognosPowerPlay.Report")
   Set objPPDim = objPPRep.DimensionLine.Item("Years")
   MsgBox "Number of categories one level below: " _
      &objPPDim.Count
   Set objPPDim = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- Chapter 4, “Methods,” on page 71
- Chapter 5, “Properties,” on page 259

CubeName Property

Returns the file name of the cube for the active report.

Syntax

Report.**CubeName**

Applies To

Report Object

Discussion

Use this property to identify a cube by its name and fully qualified path.

Type

String

Access

Read

Example

This example returns the name of the cube for the active report. The name includes the path and cube file name.

```
Sub Main()
   Dim objPPRep As Object
   Set objPPRep = GetObject(,"CognosPowerPlay.Report")
   Msgbox "The " & objPPRep.Name & " report is
based " & _
         "on the following cube." & chr$(10) &
_
         chr$(10) & objPPRep.CubeName, , "PowerPlay
Cube"
   Set objPPRep = Nothing
End Sub
```

Related Topics

- Chapter 4, "Methods," on page 71
- Chapter 5, "Properties," on page 259

DataGridlines Property

Sets or returns whether the gridline settings are on or off for a crosstab.

Syntax

object.**DataGridlines**

Applies To

Graph Object

Graphs

Discussion

Use this method to control the gridlines associated with data in a crosstab. Set this property to True to turn gridlines on, or set this property to False to turn gridlines off.

Default: True

Type

Boolean

Access

Read/Write (Graph)

Write (Graphs)

Example

This example sets the graph object of the active report to a crosstab, sets the data and label gridlines properties to False, specifies the layout to Financial with Totals, and indents the totals to the next level.

```
Sub Main
   Dim objPPRep as Object
   Dim objPPGraph as Object
   Set objPPRep = GetObject(,"CognosPowerPlay.Report")
   Set objPPGraph = objPPRep.Graphs.Item(1)
   With objPPGraph
      .SetType 0
      .Layout = 2
      .IndentTotalsLevel = 1
      .DataGridlines = False
      .LabelGridlines = False
   Set objPPGraph = Nothing
   Set objPPRep = Nothing
   End With
End Sub
```

Related Topics

- "LabelGridlines Property" on page 329

DefaultAlternateDirectory Property

Sets or returns the directory to save updates to a read-only report.

Syntax

Application.**DefaultAlternateDirectory**

Applies To

Application Object

Discussion

When you save a report using the SaveAs method, it saves to the originating directory. However, read-only reports cannot be over-written. These reports are saved to the specified DefaultAlternateDirectory or to a temporary folder, if the DefaultAlternateDirectory is not specified.

To determine the setting for DefaultAlternateDirectory, use this property.

Type

String

Access

Read/Write

Example

This example sets the 'Alternate path for read-only reports' directory entry in the Directories tab of the Preferences dialog box (File menu). The folder must exist before running this macro.

```
Sub Main()
   Dim objPPRep As Object
   Dim strPath as string
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   strPath = objPPRep.Application.Path & "\Secure
Reports"
   objPPRep.Application.DefaultAlternateDirectory = strPath
   MsgBox "The Alternate path for read-only reports "
& _
      "has been changed to " & chr$(10) & chr$(10)
& _
      objPPRep.Application.DefaultAlternateDirectory
& _
      ".", 64, "New Path"
   Set objPPRep = Nothing
End Sub
```

DefaultCubeDirectory Property

Sets or returns the default path for multi-dimensional cube files (.mdc).

Syntax

Application.**DefaultCubeDirectory**

Applies To

Application Object

Discussion

After the directory is set, all macros will look for the MDC files in that directory. You can change the directory in the user interface and the change will be reflected

in the Application object when you use it, and vice versa.

Type

String

Access

Read/Write

Example

This example sets the default cube directory to point to a subdirectory of the one in which the IBM Cognos PowerPlay application is found.The cube subdirectory, MyCubeDirectory, must exist before you run this macro.

```
Sub Main()
   Dim objPPRep As Object
   Dim strCubeDirectory as String
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   strCubeDirectory = objPPRep.Application.Path &
"\MyCubeDirectory"
   objPPRep.Application.DefaultCubeDirectory = strCubeDirectory
   MsgBox "The default cube directory is " _
       &objPPRep.Application.DefaultCubeDirectory
   Set objPPRep = Nothing
 End Sub
```

Related Topics

- Chapter 4, "Methods," on page 71
- Chapter 5, "Properties," on page 259

DefaultMacroDirectory Property

Sets or returns the default path for macro files.

Syntax

Application.**DefaultMacroDirectory**

Applies To

Application Object

Discussion

After the directory is set, all macros will look for the macro files in that directory. Changes to the directory in the user interface will be reflected in the Application object, and vice versa.

Type

String

Access

Read/Write

Example

This example sets the default macro directory to point to a subdirectory of the IBM Cognos PowerPlay application directory. Once the directory is set, macros will be loaded from that directory unless a path is specified with the file name. The macro subdirectory, MyMacroDirectory, must exist before you run this macro.

```
Sub Main()
   Dim objPPRep As Object
   Dim strMacroDirectory as String
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   strMacroDirectory = objPPRep.Application.Path &
"\MyMacroDirectory"
   objPPRep.Application.DefaultMacroDirectory = strMacroDirectory
   MsgBox "The default macro directory is " _
       &objPPRep.Application.DefaultMacroDirectory
   Set objPPRep = Nothing
End Sub
```

Related Topics

- Chapter 4, "Methods," on page 71
- Chapter 5, "Properties," on page 259

DefaultReportDirectory Property

Sets or returns the default path for IBM Cognos PowerPlay report files.

Syntax

Application.**DefaultReportDirectory**

Applies To

Application Object

Discussion

After the directory is set, all macros will look for PowerPlay report files in that directory. Changes to the directory in the user interface will be reflected in the Application object, and vice versa.

Type

String

Access

Read/Write

Example

This example sets the default report directory to point to a subdirectory of the PowerPlay application directory. The report subdirectory, MyReportDirectory, must exist before you run this macro.

```
Sub Main()
   Dim objPPRep As Object
   Dim strReportDirectory as String
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   strReportDirectory = objPPRep.Application.Path &
"\MyReportDirectory"
   objPPRep.Application.DefaultReportDirectory = strReportDirectory
   MsgBox "The default report directory is " _
       &objPPRep.Application.DefaultReportDirectory
   Set objPPRep = Nothing
End Sub
```

Related Topics

- Chapter 4, "Methods," on page 71
- Chapter 5, "Properties," on page 259

Dimension Property

Sets or returns the dimension from which categories are returned.

Syntax

object.**Dimension**

Applies To

AdvancedQuery Object

FindQuery Object

ValueRestriction Object

Discussion

Use this property to set the name of a dimension line for a subset for the AdvancedQuery and FindQuery queries only. When searching for a specific dimension, get the top level dimension line settings first. An invalid dimension line value in the subset definition returns an exception error.

For an AdvancedQuery or FindQuery, use the Execute method to run the subset definition for the query once all related properties are set.

The Dimension property must appear first, in the AdvancedQuery and FindQuery subset definitions.

For a ValueRestriction, the order of the components is important. The Dimension property must be set before the Measure, Operator, Operand1, Operand2, and Count properties and the DimensionFilter method. The Name property can be set anywhere within the filter definition.

Default: Entire Cube (FindQuery)

Type

String

Access

Read/Write

Example

This example creates a FindQuery (type 1) subset definition that searches for all products that begin with the name Star.

```
Sub Main()
   Dim strCubePath As String
   Dim objPPRep As Object
   Dim objFind As Object
   strCubePath = "C:\Cubes and Reports\Great Outdoors.mdc"
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.New strCubePath, 1
   objPPRep.ExplorerMode = False
   objPPRep.Visible = True
   Set objFind = objPPRep.ReportQueries.Add(1)
   With objFind
      .Name = "Find Star"
      .Dimension = "Products"
      .SearchShortName = False
      .SearchText = "Star"
      .Pattern = 2
   End With
   Set objFind = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "AdvancedQuery Object" on page 8
- "FindQuery Object" on page 23

DimensionLineIndex Property

Returns the position of a dimension line item to maintain a list of Layer, Row, and Column objects.

Syntax

collection.**DimensionLineIndex**

Applies To

Layers

Columns

Rows

Discussion

Use this property to determine the position number (the index value) assigned to the dimensions within a dimension line.

This property is only available if the Report object is in Explorer mode (the ExplorerMode property is True).

Type

Long

Access

Read

Example

This example returns the Dimension Line index for rows, columns and layers in the active report.

```
Sub Main()
   Dim objPPRep As Object
   Set objPPRep = GetObject(,"CognosPowerPlay.Report")
   objPPRep.ExplorerMode = True
   Msgbox "The Dimension line index for columns is "
& _
      objPPRep.Columns.DimensionLineIndex & "."
   Msgbox "The Dimension line index for rows is " &
_
      objPPRep.Rows.DimensionLineIndex & "."
   Msgbox "The Dimension line index for layers is " &
_
      objPPRep.Layers.DimensionLineIndex & "."
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "DimensionFilter Method" on page 133
- "DimensionLine Method" on page 135
- "DimensionLine Object" on page 21
- "ValueRestriction Object" on page 46

DimensionSettings Property

Returns a comma-separated string of all the dimension line settings for the ValueRestriction.

Syntax

ValueRestriction.**DimensionSettings**

Applies To

ValueRestriction Object

Discussion

Each dimension has its own folder on the dimension line. For example, the dimensions in the Great Outdoors cube include Years, Products, Locations, Channels and Margin Ranges. Use this property when you want to know the current settings for a dimension line for the ValueRestriction.

Type

String

Access

Read

Example

This example creates an advanced subset that selects countries or regions from the Locations dimension. The value restriction (type 4) limits the results to return only those countries or regions whose Revenue values for Sports Chains are between 25,000 and 100,000. The dimension settings results for the Great Outdoors cube are Years, Products, Locations, Sports Chain, and Margin Ranges.

```
Sub Main()
   Dim strCubePath As String
   Dim objPPRep As Object
   Dim objValue As Object
   Dim objAdvanced As Object
   strCubePath = "C:\Cubes and Reports\Great Outdoors.mdc"
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.New strCubePath, 1
   objPPRep.ExplorerMode = False
   objPPRep.Visible = True
   Set objValue = objPPRep.ReportQueries.Add(4)
   With objValue
      .Name = "25000-100000"
      .Dimension = "Locations"
      .Measure = "Revenue"
      .Operator = "between"
      .Operand1 = 25000
      .Operand2 = 100000
```

```
      .DimensionFilter 4, "Sports Chain"
   End With
   Set objAdvanced = objPPRep.ReportQueries.Add(3)
   With objAdvanced
      .Name = "Locations"
      .Dimension = "Locations"
      .Level "Country or Region"
      .ValueRestriction objValue.Name
      .Execute
      .AddToReport 0,1,3
   End With
   Msgbox "The Dimension Line Settings for this " &
_
         "report are:" & chr$(10) & chr$(10)
& _
         objValue.DimensionSettings, , "Dimension Line"
   Set objAdvanced = Nothing
   Set objValue = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- Chapter 4, "Methods," on page 71
- Chapter 5, "Properties," on page 259

DrivingCategory Property

Returns the driving category for the Exception object.

Syntax

Exception.**DrivingCategory**

Applies To

Exception Object

Discussion

Use this property to determine the category whose values are compared with those in the Exception object.

Type

String

Access

Read

Example

This example opens a report and displays the driving category and dimension for the first Exception object.

```
Sub Main()
   Dim objPPRep as Object
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.Open "C:\Cubes and Reports\Exception.ppx"
   MsgBox "Driving Category:" _
      & objPPRep.Exceptions.Item(1).DrivingCategory
   MsgBox "Driving Dimension:" _
      & objPPRep.Exceptions.Item(1).DrivingDimension
   Set objPPRep = Nothing
End Sub
```

Related Topics

- Chapter 4, "Methods," on page 71
- Chapter 5, "Properties," on page 259
- "Graph Object" on page 25

DrivingDimension Property

Returns the driving dimension for the Exception object.

Syntax

Exception.**DrivingDimension**

Applies To

Exception Object

Discussion

Use this property to determine the Dimension whose values are compared with those in the Exception object.

Type

Long

Access

Read

Example

This example opens a report and displays the driving category and dimension for the first Exception object.

```
Sub Main()
   Dim objPPRep as Object
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.Open "C:\Cubes and Reports\Exception.ppx"
```

```
   MsgBox "Driving Category:" _
      & objPPRep.Exceptions.Item(1).DrivingCategory
   MsgBox "Driving Dimension:" _
      & objPPRep.Exceptions.Item(1).DrivingDimension
   Set objPPRep = Nothing
End Sub
```

Related Topics

- Chapter 4, "Methods," on page 71
- Chapter 5, "Properties," on page 259
- "Exception Object" on page 22

Each Property

Sets or returns whether all selected and new categories or just the new categories appear in the Report object.

Syntax

CategoryList.**Each**

Applies To

CategoryList Object

Discussion

This property must be used in conjunction with one of the Sum, Intersection, or Average properties set to True. If the Each property is set to True, all selected and new categories resulting from Sum, Intersection, or Average being set to True appear in the report. If False, only the Sum, Intersection, or Average categories appear in the report.

Default: False

Type

Boolean

Access

Read/Write

Example

This example creates a new report from the Great Outdoors.mdc, adds categories to the category list as a sum, but the Each property is set to False, so only the Sum category will be shown. The categories are added to the report.

```
Sub Main
   Dim objPPRep as Object
   Dim objCatList as Object
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.New "C:\Cubes and Reports\Great Outdoors.mdc"
```

```
   objPPRep.ExplorerMode = False
   objPPRep.Visible = True
   Set objCatList = objPPRep.CategoryList
   objCatList.Add 1, "Products", _
      "Environmental Line", "Alert Devices"
   objCatList.Each = False
   objCatList.Sum = True
   objPPRep.Columns.Add objCatList
   objCatList.Remove
   objCatList.Add 1, "Years"
   objCatList.Each = True
   objCatList.Sum = False
   objPPRep.Rows.Add objCatList
   Set objCatList = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- Chapter 4, "Methods," on page 71
- Chapter 5, "Properties," on page 259
- "CategoryList Object" on page 13

EnableUserColumnSummaryLabel Property

Sets or returns whether a user-defined label is used for the innermost summary column in a nested crosstab.

Syntax

object.**EnableUserColumnSummaryLabel**

Applies To

Graph Object

Graphs

Discussion

Use this property to specify whether the column uses the higher-level column category or a user-defined name as the label for a summary column. If True, you can use the UserColumnSummaryLabel property to define the name of the label to apply to the column.

The column label is only visible in a crosstab report when the ShowSummaryColumn property is set to True.

Default: False

Type

Boolean

Access

Read/Write (Graph)

Write (Graphs)

Example

This example resets the graph object to use user-defined summary labels for rows and columns.

```
Sub Main
   Dim objPPRep as Object
   Dim objPPGraph as Object
   Set objPPRep = GetObject(,"CognosPowerPlay.Report")
   Set objPPGraph = objPPRep.Graphs.Item(1)
   objPPGraph.EnableUserColumnSummaryLabel = True
   objPPGraph.UserColumnSummaryLabel = "Summary Total"
   objPPGraph.EnableUserRowSummaryLabel = True
   objPPGraph.UserRowSummaryLabel = "Summary Total"
   Set objPPGraph = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "EnableUserRowSummaryLabel Property"

EnableUserRowSummaryLabel Property

Sets or returns whether a user-defined label is used for the innermost summary row in a nested crosstab.

Syntax

object.**EnableUserRowSummaryLabel**

Applies To

Graph Object

Graphs

Discussion

Use this property to specify whether the row uses the higher-level row category or a user-defined name as the label for a summary row. If True, you can use the UserRowSummaryLabel property to define the name of the label to apply to the row.

The row label is only visible in a crosstab report when the ShowSummaryRow property is set to True.

Default: True

Type

Boolean

Access

Read/Write (Graph)

Write (Graphs)

Example

This example resets the graph object to use user-defined summary labels for rows and columns.

```
Sub Main
   Dim objPPRep as Object
   Dim objPPGraph as Object
   Set objPPRep = GetObject(,"CognosPowerPlay.Report")
   Set objPPGraph = objPPRep.Graphs.Item(1)
   objPPGraph.EnableUserColumnSummaryLabel = True
   objPPGraph.UserColumnSummaryLabel = "Summary Total"
   objPPGraph.EnableUserRowSummaryLabel = True
   objPPGraph.UserRowSummaryLabel = "Summary Total"
   Set objPPGraph = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "EnableUserColumnSummaryLabel Property" on page 303

Exception Property

Sets or returns the exception for one or more categories.

Syntax

object.**Exception**

Applies To

Column Object

Columns

Layers

Row Object

Rows

Discussion

To identify exceptions used for an object, a name for each exception is required.

To remove an exception, redefine the exception style.

Type

String

Access

Read/Write (Column, Layer, Row)

Write (Columns, Layers, Rows)

Example

This example applies the "Default" exception to the third row and then prints the report.

```
Sub Main()
   Dim objPPRep as Object
   Dim objRepPrt as Object
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.Open "C:\Cubes and Reports\Exception.ppx"
   objPPRep.Rows.Item(3).Exception = "Default"
   Set objRepPrt = objPPRep.Print
   objRepPrt.PrintAllCharts = False
   objRepPrt.SetListOfRowsToPrint objPPrep.Rows
   objRepPrt.SetListOfLayersToPrint objPPRep.Layers
   objRepPrt.SetChartToPrint objPPRep.Graphs.Item(1)
   objRepPrt.IncludeLegend = False
   objRepPrt.ChartTitleOnAllPages = True
   objRepPrt.SummariesOnAllPages = True
   objRepPrt.AxisOnAllPages = True
   objRepPrt.Collate = True
   objRepPrt.PrintOut
   Set objRepPrt = Nothing
   Set objPPrep = Nothing
End Sub
```

Related Topics

- "Column Object" on page 15
- "Layer Object" on page 28
- "Row Object" on page 41

ExplorerMode Property

Sets or returns whether the Report object is an Explorer or Reporter report.

Syntax

Report.**ExplorerMode**

Applies To

Report Object

Discussion

If True, the report is an Explorer report and certain methods and properties are not available. If False, the report is a Reporter report and all methods and properties are available.

Default: True

Type

Boolean

Access

Read/Write

Example

This example creates a new report, sets the report to Reporter mode, and adds categories to the category list as a sum.

```
Sub Main
   Dim objPPRep as Object
   Dim objCatList as Object
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.New "C:\Cubes and Reports\Great Outdoors.mdc"
   objPPRep.ExplorerMode = False
   objPPRep.Visible = True
   Set objCatList = objPPRep.CategoryList
   objCatList.Add 1, "Products", _
      "Environmental Line", "Alert Devices"
   objCatList.Each = False
   objCatList.Sum = True
   objPPRep.Columns.Add objCatList
   objCatList.Remove
   objCatList.Add 1, "Years"
   objCatList.Each = True
   objCatList.Sum = False
   objPPRep.Rows.Add objCatList
   Set objCatList = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "Row Object" on page 41

FitToPage Property

Sets or returns whether the report is scaled to fit on one page.

Syntax

Print.**FitToPage**

Applies To

Print Object

Discussion

Use this property when you want to scale down a multiple page report to a single page. Set this property to True to make a scrolling display print on one page. If False, the report prints on as many pages as required.

Default: False

Type

Boolean

Access

Read/Write

Example

This example sets the FitToPage property to True and prints the first display of the active report.

```
Sub Main()
   Dim objPPRep as Object
   Dim objRepPrt as Object
   Set objPPRep = GetObject(,"CognosPowerPlay.Report")
   Set objRepPrt = objPPRep.Print()
   With objRepPrt
      .PrintAllCharts = False
      .SetListOfRowsToPrint objPPrep.Rows
      .SetListOfLayersToPrint objPPRep.Layers
      .SetChartToPrint objPPRep.Graphs.Item(1)
      .IncludeLegend = False
      .ChartTitleOnAllPages = True
      .SummariesOnAllPages = True
      .AxisOnAllPages = True
      .Collate = True
      .Copies = 1
      .FitToPage = True
      .PrintOut
   End With
   Set objRepPrt = Nothing
```

```
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "SetChartToPrint Method" on page 223
- "SetListOfLayersToPrint Method" on page 229
- "SetListOfRowsToPrint Method" on page 232

FooterText Property

Sets or returns the text in the footer of a report.

Syntax

Report.FooterText(Format)

Applies To

Report Object

Discussion

Use this property to set or return the complete text in the footer of a report. Because you can add as many lines to the footer as required, the footer may contain information that is not visible when you view or print the report. You can use automation to view information that the report author included in the footer but did not want the users to see. You can set or return the footer of a report in text or HTML format.

If you set the footer as text, then the footer is left justified and uses the default font size, font color, and font type. To specify multiple lines in a footer, use the ASCII value of a new line character, chr$(10), between the lines of the footer.

If you set the footer of a report using HTML, you can specify complex formats. You can specify the font size, font color, font type, and alignment of the footer text. For example, the following code aligns the footer text to the right, and specifies the font size and color.

```
strFooter = strFooter + "<P ALIGN=""Right""><FONT
SIZE=6 COLOR=""#0000FF"">"
```

When you assign HTML tags to a string, use two sets of quotation marks to distinguish characters within the HTML tag from end of string quotation marks. For example, to assign <P ALIGN="Right"> to a string, use the following syntax:

```
objPPRep.FooterText(2) = "<P ALIGN=""Right"">"
```

If you use an external editor to specify HTML, you may have to modify the HTML to get the format you require for the report footer.

A report footer can include IBM Cognos PowerPlay variables or variables expanded to the values that they represent. To specify a variable use the following syntax:

```
[PPVAR]Variable[PPVAR]
```

For example,

```
objPPRep.FooterText(1) = "[PPVAR]Page #[PPVAR]"
```

You can return the value of the variable or the expanded variable. To return the variable expanded to the value it represents, use the parameter value 11 for text or 12 for HTML.

For example,

```
objPPRep.FooterText(1) = "Report Printed [PPVAR]Date(ddd,
dd MMM, yyy)[PPVAR]"
MsgBox objPPRep.FooterText(1)
```

returns

Report Printed [PPVAR]Date(ddd, dd MMM, yyy)[PPVAR]

```
MsgBox objPPRep.FooterText(11)
```

returns

Report Printed Thursday, 13 January, 2000

Parameter	Description
Format	Optional. Specifies the format of the text in the footer of a report. If not specified, the text and variable are returned. Valid set and return values are 1 = Text 2 = HTML Valid return values are 11 = Text with variables expanded 12 = HTML with variables expanded **Default:** 1 Type: Integer

Type

String

Access

Read/Write

Example

This example specifies two lines of text for the footer of the open report, and shows the footer text.

```
Sub Main()
   Dim objRep As Object
   Dim strFooterTextLine1 as String
   Dim strFooterTextLine2 as String
   Dim strNewLine as String
   Set objRep = GetObject( ,"CognosPowerPlay.Report")
```

```
   strFooterTextLine1 = "Annual Report"
   strFooterTextLine2 = "[PPVAR]Date(dddd, MMMM dd, yyyy)[PPVAR]"
   strNewLine = chr$(10)
   objRep.FooterText(1) = strFooterTextLine1 + chr$(10)
_
       +  strFooterTextLine2
   MsgBox "The report footer text is:  " _
      &objRep.FooterText(11)
   objRep.Save
   Set objRep = Nothing
End Sub
```

Related Topics

- "HeaderText Property" on page 313
- "Report Object" on page 37
- "Reports" on page 65
- "TitleText Property" on page 408

FullName Property

Returns the full name, including the location, of either the Application object or the Report object.

Syntax

object.**FullName**

Applies To

Application Object

Report Object

Discussion

The FullName property of the Application object cannot be modified while the one for a Report object can. To modify the FullName property for a report, use the SaveAs method.

Type

String

Access

Read

Example

This example creates an instance of IBM Cognos PowerPlay and shows the name and location of the Application object.

```
Sub Main()
   Dim objPPlayApp as Object
```

```
   Set objPPlayApp = CreateObject("CognosPowerPlay.Application")
   objPPlayApp.Visible = 1
   MsgBox "The name and location of the application is
" _
      &objPPLayApp.FullName
   MsgBox "The application version is " &objPPLayApp.Version
   Set objPPlayApp = Nothing
End Sub
```

Related Topics

- "Application Object" on page 11
- "Report Object" on page 37

GetDataAutomatically Property

Sets or returns whether the Report object retrieves data automatically each time it is modified.

Syntax

Report.**GetDataAutomatically**

Applies To

Report Object

Discussion

If True, the data is updated with each report modifying action such as ranking and calculations. If False, calculations, ranking, and other report modifying actions will update the number of columns and rows, but the data will not be updated. Set the property to False when using a large amount of data. Once completed, you can get the data using the GetDataNow method. This way, the Application object is not constantly polling the cube for new information.

Default: False

Type

Boolean

Access

Read/Write

Example

This example opens a report, and updates the data in the report using the GetDataNow method, if the GetDataAutomatically property is set to False.

```
Sub Main()
   Dim objPPRep as Object
   Set objPPRep = CreateObject ("CognosPowerPlay.Report")
   objPPRep.Open "C:\Cubes and Reports\Sample2.ppx"
   If objPPRep.GetDataAutomatically = 0 then
```

```
      objPPRep.GetDataNow
   End if
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "Report Object" on page 37

HeaderText Property

Sets or returns the text in the header of a report.

Syntax

Report.**HeaderText**(Format)

Applies To

Report Object

Discussion

Use this property to set or return the complete text in the header of a report. Because you can add as many lines to the header as required, the header may contain information that is not visible when you view or print the report. You can use automation to view information that the report author included in the header but did not want the users to see. You can set or return the header of a report in text or HTML format.

If you set the header as text, then the header is left justified and uses the default font size, font color, and font type. To specify multiple lines in a footer, use the ASCII value of a new line character, chr$(10), between the lines of the header.

If you set the header of a report using HTML, you can specify complex formats. You can specify the font size, font color, font type and alignment of the header text. For example, the following code aligns the header text to the right and specifies the font size and color:

```
strFooter = strFooter + "<P ALIGN=""Right""><FONT
SIZE=6 COLOR=""#0000FF"">"
```

When you assign HTML tags to a string, use two sets of quotation marks to distinguish characters within the HTML tag from end of string quotation marks. For example, to assign <P ALIGN="Right"> to a string, use the following syntax:

```
objPPRep.HeaderText(2) = "<P ALIGN=""Right"">"
```

If you use an external editor to specify HTML, you may have to modify the HTML to get the format you require for the report header.

A report header can include IBM Cognos PowerPlay variables or variables expanded to the values that they represent. To specify a variable, use the following syntax

```
[PPVAR]Variable[PPVAR]
```

For example,

```
objPPRep.HeaderText(1) = "[PPVAR]Page #[PPVAR]"
```

You can return the value of the variable or the expanded variable. To return the variable expanded to the value it represents, use the parameter value 11 for text or 12 for HTML.

For example,

```
objPPRep.HeaderText(1) = "Report Printed [PPVAR]Date(ddd,
dd MMM, yyy)[PPVAR]"
MsgBox objPPRep.HeaderText(1)
```

returns

Report Printed [PPVAR]Date(ddd, dd MMM, yyy)[PPVAR]

```
MsgBox objPPRep.HeaderText(11)
```

returns

Report Printed Thursday, 13 January, 2000

Parameter	Description
Format	Optional. Specifies the format of the text in the header of a report. If not specified, the text and variable are returned. Valid set and return values are 1 = Text 2 = HTML Valid return values are 11 = Text with variables expanded 12 = HTML with variables expanded **Default:** 1 Type: Integer

Type

String

Access

Read/Write

Example

This example specifies the text for the header of the open report and shows the current date.

```
Sub Main()
   Dim objRep As Object
   Dim strHeaderTextLine1 as String
   Dim strNewLine as String
   Set objRep = GetObject( ,"CognosPowerPlay.Report")
   strHeaderTextLine1 = "[PPVAR]Date(dddd, MMMM dd, yyyy)[PPVAR]"
```

```
   objRep.HeaderText(1) = strHeaderTextLine1
   MsgBox "The report header text is:  " _
      &objRep.HeaderText(11)
   objRep.Save
   Set objRep = Nothing
End Sub
```

Related Topics

- "FooterText Property" on page 309
- "Report Object" on page 37
- "Reports" on page 65
- "TitleText Property" on page 408

HideRankCategory Property

Sets or returns whether the rank category is hidden.

Syntax

object.**HideRankCategory**

Applies To

Graph Object

Graphs

Discussion

A rank category is a row or column that identifies ordinal values for categories. If True, the rank category is hidden. If False, it is not hidden. This property is not available with the crosstab display.

Default: True

Type

Boolean

Access

Read/Write (Graph)

Write (Graphs)

Example

This example hides the rank category when a bar display is used in the report "c:\cognos\sample.ppx".

```
Sub Main()
   Dim objPPRep As Object
   Dim objPPGraph As Object
   Set objPPRep = GetObject ("C:\Cubes and Reports\Sample2.ppx")
```

```
    objPPRep.Visible = True
    objPPRep.Activate
    Set objPPGraph = objPPRep.Graphs (1)
    objPPGraph.SetType 3, 1, 1
    objPPGraph.HideRankCategory = True
    Set objPPGraph = Nothing
    Set objPPRep = Nothing
End Sub
```

Related Topics

- Chapter 4, "Methods," on page 71
- Chapter 5, "Properties," on page 259
- "Graph Object" on page 25
- "Graphs" on page 56

IncludeLegend Property

Sets or returns whether the legend appears in a printed report or PDF.

Syntax

object.**IncludeLegend**

Applies To

Print Object

SaveAsPDF Object

Discussion

Use this property when you want to determine whether to show an explanatory list of categories in the report. The list shows the category name and color representing the associated data. The legend does not appear in crosstab, simple bar, single line, and three-dimensional bar displays.

If True, any legend associated with the display appears on a separate page of the report. If False, the legend is not included on the printed report or PDF.

Use the ColorsAsPatterns property to determine if the colors in the legend appear as true colors or as patterns when they are printed.

To use this property when printing a report, the PrintAllCharts property must be set to False.

To use this property when saving a report as a PDF, the SaveAllCharts property must be set to False, and the SaveEntireReport must be set to True.

Default: False

Type

Boolean

Access

Read/Write

Example

This example opens the report and prints one copy of all the data for the second graphical display only. This example includes the legend in the printed report.

```
Sub Main()
   Dim objPPRep as Object
   Dim objRepPrt as Object
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.Open "C:\Cubes and Reports\Sample1.ppx"
   Set objRepPrt = objPPRep.Print
   objRepPrt.PrintAllCharts = False
   objRepPrt.SetListOfRowsToPrint objPPrep.Rows
   objRepPrt.SetListOfLayersToPrint objPPRep.Layers
   objRepPrt.SetChartToPrint objPPRep.Graphs.Item(2)
   objRepPrt.IncludeLegend = True
   objRepPrt.ChartTitleOnAllPages = True
   objRepPrt.SummariesOnAllPages = True
   objRepPrt.AxisOnAllPages = True
   objRepPrt.Copies = 1
   objRepPrt.PrintOut
   Set objRepPrt = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "SaveEntireReport Property" on page 379

IndentTotalsLevel Property

Sets or returns the current indent level for summary cells in a nested crosstab.

Syntax

object.**IndentTotalsLevel**

Applies To

Graph Object

Graphs

Discussion

Rows and columns in nested crosstabs can be summarized to display totals. IndentTotalsLevel controls how to indent these total categories. IndentTotalsLevel takes effect in Explorer mode when the Layout property is set to 2-Indent Layout 2. The available options are current level, previous level (if it exists) and align right.

Default: Current level

Type

Integer

Access

Read/Write (Graph)

Write (Graphs)

Example

This example sets the graph object of the active report to a crosstab, sets the data and label gridlines properties to false, specifies the layout to Financial with Totals, and indents the totals to the next level.

```
Sub Main
   Dim objPPRep as Object
   Dim objPPGraph as Object
   Set objPPRep = GetObject(,"CognosPowerPlay.Report")
   Set objPPGraph = objPPRep.Graphs.Item(1)
   With objPPGraph
      .SetType 0
      .Layout = 2
      .IndentTotalsLevel = 1
      .DataGridlines = False
      .LabelGridlines = False
   End With
   Set objPPGraph = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "Layout Property" on page 331

Index Property

Returns the position of an object in a collection.

Syntax

object.**Index**

Applies To

Column Object

Dimension Object

Graph Object

Layer Object

Range Object

Row Object

Report Object

Discussion

Use this property to determine the numerical position of an object within a collection.

Type

Long

Access

Read

Example

This example ranks report columns by 1996 results and shows the position of the "GO Sport Line" column to determine its rank among other products.

```
Sub Main()
   Dim objPPRep as Object
   Dim objPPRank as Object
   Set objPPRep = GetObject( ,"CognosPowerPlay.Report")
   objPPRep.Rows.Item("1996").Rank , , 0, 1
   Set objPPRank = objPPRep.Columns.Item("Go Sport Line")
   MsgBox "The rank of Go Sport Line is " & objPPRank.Index
   Set objPPRank = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "Column Object" on page 15
- "Dimension Object" on page 19
- "Graph Object" on page 25
- "Layer Object" on page 28
- "Range Object" on page 35
- "Row Object" on page 41

Intersect Property

Sets or returns whether to determine the values at the intersection of selected categories from different Dimension objects.

Syntax

CategoryList.**Intersect**

Applies To

CategoryList Object

Discussion

If True, a new category is created to show the values at the intersection of the selected categories from different dimensions in the CategoryList object.

This property is only available in Reporter reports.

Default: False

Type

Boolean

Access

Read/Write

Example

This example selects Years as columns and shows the intersection of North American and Independent markets of Outdoor Products as rows.

```
Sub Main
   Dim objPPRep as Object
   Dim objCatList as Object
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.New "C:\Cubes and Reports\Great Outdoors.mdc",False
   objPPRep.ExplorerMode = False
   objPPRep.Visible = True
   Set objCatList = objPPRep.CategoryList
   objCatList.Add 1, "Years"
   objPPRep.Columns.Add objCatList
   objCatList.Remove
   objCatList.Add 0, "Products", "Outdoor Products"
   objCatList.Add 0, "Channels",  "Independent"
   objCatList.Add 0, "Locations", "Americas"
   objCatList.Each = False
   objCatList.Intersect = True
   objPPRep.Rows.Add objCatList
   Set objCatList = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "CategoryList Object" on page 13
- "Dimension Object" on page 19

IsAlternate Property

Returns whether the drill-down path is primary or alternate.

Syntax

object.**IsAlternate**

Applies To

Column Object

Dimension Object

Layer Object

Row Object

Discussion

Use this property to identify the type of path for a category. If True (-1), the category is along an alternate drill-down path. False (0) indicates that the category is along the primary drill-down path.

Type

Boolean

Access

Read

Example

This example uses the IsAlternate property to determine the type of drill-down path (primary or alternate).

```
Sub Main
   Dim strCubePath As String
   Dim objPPRep As Object
   Dim objDimension As Object
   Dim strDrill As String
   strCubePath = "C:\Cubes and Reports\Great Outdoors.mdc"
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.New strCubePath, 1
   Set objDimension = objPPRep.DimensionLine.Item(4)
   objDimension.Change "By Region"
   If objDimension.IsAlternate = True Then
      strDrill = "alternate"
   Else
      strDrill = "primary"
   End If
   MsgBox "The " & objDimension.Name & " dimension
" & _
```

```
        "is from the " & strDrill & " drill
path."
   Set objDimension = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "Level Object" on page 30
- "Levels" on page 61
- "Levels Method" on page 169

IsCalculatedCategory Property

Returns whether the category is a calculated category.

Syntax

object.**IsCalculatedCategory**

Applies To

Column Object

Dimension Object

Layer Object

Row Object

Discussion

Use this property to identify levels that have calculated categories. If True (-1), the category is a calculated category. False (0) indicates that the category is not a calculated category.

Type

Boolean

Access

Read

Example

This example uses the IsCalculatedCategory property to determine there is a calculated category.

```
Sub Main
   Dim objPPRep As Object
   Dim objLayer As Object
   Dim objLevel As Object
   Dim strLevel As String
   Dim strDrill As String
   Dim strCategory As String
```

```
   Set objPPRep = GetObject(,"CognosPowerPlay.Report")
   Set objLayer = objPPRep.Layers.Item(1)
   Set objLevel = objLayer.Level
   strLevel = objLevel.Name
   If objLayer.IsAlternate = True Then
      strDrill = "alternate"
   Else
      strDrill = "primary"
   End If
   If objLayer.IsCalculatedCategory = True Then
      strCategory = ""
   Else
      strCategory = "not"
   End If
   MsgBox "The " & objLayer.Name & " category
is a " & _
      strDrill & " drill-down and is " & strCategory
& _
      " a calculated category." & chr$(10) &
chr$(10) & _
      "It is a member of the " & strLevel & "
level."
   Set objLayer = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "Level Object" on page 30
- "Levels" on page 61
- "Levels Method" on page 169

KeepSummaryVisible Property

Sets or returns whether the summary category will remain visible on all scrolled pages.

Syntax

object.**KeepSummaryVisible**

Applies To

Graph Object

Graphs

Discussion

Use this property to set the summary category to visible on every page of data, or to visible on only the last page of data. If the summary category is hidden or does not exist, this property is not effective.

This property is not available for crosstab, 3-D bar, scatter, and pie displays.

Default: False

Type

Boolean

Access

Read/Write (Graph)

Write (Graphs)

Example

This example changes the scroll bar settings of an active report. The scrolling feature is set to True, maximum visible bars is set to six, maximum printed bars is set to ten, and the summary category is kept hidden until the end of the report.

```
Sub Main()
  Dim PPRep As Object
  Set PPRep = GetObject(,"CognosPowerPlay.Report")
  PPRep.Graphs.Item(1).UseScrolling = TRUE
  PPRep.Graphs.Item(1).MaxVisibleBars = 6
  PPRep.Graphs.Item(1).MaxPrintedBars = 10
  PPRep.Graphs.Item(1).KeepSummaryVisible = FALSE
  Set PPRep = Nothing
End Sub
```

Related Topics

- "Graph Object" on page 25
- "Graphs" on page 56

LabelAlignment Property

Returns the alignment applied to a cell label in a report.

Syntax

object.**LabelAlignment**

Applies To

Column Object

Row Object

Discussion

Use this property to determine the alignment for row or column labels, especially for a crosstab that uses different formats for each category.

When you return the label alignment setting for a row or column, the LabelAlignment property applies to the active Report object only. The property does not apply to row or column labels in other reports in the Reports Collection.

The default column or row label alignment setting for other reports remains unchanged, unless previously set in the application.

Valid alignment values are

0 = left aligned 1 = center aligned 2 = right aligned 3 = default

Default: 3 (report setting, if no alignment set for label)

Type

Long

Access

Read

Example

This example returns the alignment for the second row and third column labels.

```
Sub Main()
   Dim objPPRep as Object
   Set objPPRep = CreateObject ("CognosPowerPlay.Report")
   objPPRep.Open "C:\Cubes and Reports\Sample1.ppx"
   MsgBox "The alignment of the second row label is "
_
      &objPPRep.Rows.Item(2).LabelAlignment
   MsgBox "The alignment of the third row label is "
_
      &objPPRep.Columns.Item(3).LabelAlignment
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "LabelFontColor Property"
- "LabelFontName Property" on page 327
- "LabelFontSize Property" on page 328

LabelFontColor Property

Returns the font color applied to a category label in a report.

Syntax

object.**LabelFontColor**

Applies To

Column Object

Layer Object

Row Object

Discussion

Use this property to determine the font color applied to a row, column, or layer label, especially for a crosstab that uses multiple colors to emphasize specific categories. A report author may choose to use different colors for category labels to make the report easier to read.

When you return the font color setting for a row, column or layer, the LabelFontColor property applies to the active Report object only. The property does not apply to other reports in the Reports Collection. The default font color setting for the other reports remains unchanged, unless previously set in the application.

Valid font colors are

0 = Black

128 = Brown

32768 = Green

32896 = Olive

8388608 = Navy

8388736 = Purple

8421376 = Teal

8421504 = Gray

12632256 = Silver

255 = Red

65280 = Lime

65535 = Yellow

16711680 = Blue

16711935 = Fuschia

16776960= Aqua

16777215 = White

Default: 0 (black)

Type

Long

Access

Read

Example

This example returns the font color of the second row, third column, and first layer labels.

```
Sub Main()
   Dim objPPRep as Object
   Set objPPRep = CreateObject ("CognosPowerPlay.Report")
   objPPRep.Open "C:\Cubes and Reports\Sample1.ppx"
   MsgBox "The color of the font for the second row label
is " _
      &objPPRep.Rows.Item(2).LabelFontColor
   MsgBox "The color of the font for the third column
label is " _
      &objPPRep.Columns.Item(3).LabelFontColor
   MsgBox "The color of the font for the first layer
label is " _
      &objPPRep.Layers.Item(1).LabelFontColor
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "LabelAlignment Property" on page 324
- "LabelFontName Property"
- "LabelFontSize Property" on page 328

LabelFontName Property

Returns the name of the font applied to a category label in a report.

Syntax

object.**LabelFontName**

Applies To

Column Object

Layer Object

Row Object

Discussion

Use this property to determine the name of the font applied to a row, column, or layer label, especially for a crosstab that uses multiple fonts to emphasize specific categories. A report author may choose to use different fonts for category labels to make the report easier to read.

When you return the font name setting a row, column, or layer, the LabelFontName property applies to the active Report object only. The property does not apply to other reports in the Reports Collection. The default font name setting for the other reports remains unchanged, unless previously set in the application.

Default: Arial

Type

String

Access

Read

Example

This example returns the name of the font applied to the second row, third column, and first layer labels.

```
Sub Main()
   Dim objPPRep as Object
   Set objPPRep = CreateObject ("CognosPowerPlay.Report")
   objPPRep.Open "C:\Cubes and Reports\Sample1.ppx"
   MsgBox "The name of the font for the second row label
is " _
      &objPPRep.Rows.Item(2).LabelFontName
   MsgBox "The name of the font for the third column
label is " _
      &objPPRep.Columns.Item(3).LabelFontName
   MsgBox "The name of the font for the first layer label
is " _
      &objPPRep.Layers.Item(1).LabelFontName
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "LabelAlignment Property" on page 324
- "LabelFontColor Property" on page 325
- "LabelFontSize Property"

LabelFontSize Property

Returns the size of the font applied to a category label in a report.

Syntax

object.**LabelFontSize**

Applies To

Column Object

Layer Object

Row Object

Discussion

Use this property to determine the size of the font applied to a row, column, or layer label, especially for a crosstab that uses multiple font sizes to emphasize specific categories. A report author may choose to use different font sizes for category labels to make the report easier to read.

When you return the font size setting a row, column, or layer, the LabelFontSize property applies to the active Report object only. The property does not apply to other reports in the Reports Collection. The default font size setting for the other reports remains unchanged, unless previously set in the application.

Default: 10

Type

Long

Access

Read

Example

This example returns the name of the font applied to the second row, third column, and first layer labels.

```
Sub Main()
   Dim objPPRep as Object
   Set objPPRep = CreateObject ("CognosPowerPlay.Report")
   objPPRep.Open "C:\Cubes and Reports\Sample1.ppx"
   MsgBox "The size of the font for the second row label
is " _
      &objPPRep.Rows.Item(2).LabelFontSize
   MsgBox "The size of the font for the third column
label is " _
      &objPPRep.Columns.Item(3).LabelFontSize
   MsgBox "The size of the font for the first layer label
is "
      &objPPRep.Layers.Item(1). LabelFontSize
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "LabelAlignment Property" on page 324
- "LabelFontColor Property" on page 325
- "LabelFontName Property" on page 327

LabelGridlines Property

Sets or returns whether the gridlines are on or off for category labels in a nested crosstab.

Syntax

object.**LabelGridlines**

Applies To

Graph Object

Graphs

Discussion

Use this method to control the gridlines associated with category labels in nested crosstabs. Set this property to True to turn gridlines on, or set this property to False to turn gridlines off.

Default: True

Type

Boolean

Access

Read/Write (Graph)

Write (Graphs)

Example

This example sets the graph object of the active report to a crosstab, sets the data and label gridlines properties to false, specifies the layout to Financial with Totals, and indents the totals to the next level.

```
Sub Main
   Dim objPPRep as Object
   Dim objPPGraph as Object
   Set objPPRep = GetObject(,"CognosPowerPlay.Report")
   Set objPPGraph = objPPRep.Graphs.Item(1)
   With objPPGraph
      .SetType 0
      .Layout = 2
      .IndentTotalsLevel = 1
      .DataGridlines = False
      .LabelGridlines = False
   End With
   Set objPPGraph = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "DataGridlines Property" on page 291)

Layout Property

Sets or returns the current layout style in a nested crosstab.

Syntax

object.**Layout**

Applies To

Graph Object

Graphs

Discussion

Sets the layout of nested crosstabs to one of three styles

- 0 = standard
- 1 = indent layout 1
- 2 = indent layout 2 (only available in Explorer mode)

Default: 0

Type

Integer

Access

Read/Write (Graph)

Write (Graphs)

Example

This example sets the graph object of the active report to a crosstab, sets the data and label gridlines properties to false, specifies the layout to Financial with Totals, and indents the totals to the next level.

```
Sub Main
   Dim objPPRep as Object
   Dim objPPGraph as Object
   Set objPPRep = GetObject(,"CognosPowerPlay.Report")
   Set objPPGraph = objPPRep.Graphs.Item(1)
   With objPPGraph
      .SetType 0
      .Layout = 2
      .IndentTotalsLevel = 1
      .DataGridlines = False
      .LabelGridlines = False
   End With
   Set objPPGraph = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "IndentTotalsLevel Property" on page 317)

Level Property

Returns the level of the category in a dimension.

Syntax

object.**Level**

Applies To

Column Object

Dimension Object

Layer Object

Row Object

Discussion

Use this property to return the level (not the nesting level) of a category. Top level categories in a dimension do not have levels. Alternate categories, calculated categories and some measures, depending on the cube type, do not have levels. If you use this property on a category that has no level, the name of the level returned will be an empty string.

Type

Object

Access

Read

Example

This example uses the Level property to determine the name of the Column object.

```
Sub Main
   Dim objPPRep As Object
   Dim objColumn As Object
   Dim objLevel As Object
   Dim strLevel As String
   Dim strDrill As String
   Dim strCategory As String
   Set objPPRep = GetObject(,"CognosPowerPlay.Report")
   Set objColumn = objPPRep.Columns.Item(1)
   Set objLevel = objColumn.Level
   strLevel = objLevel.Name
   If objColumn.IsAlternate = True Then
      strDrill = "alternate"
```

```
   Else
      strDrill = "primary"
   End If
   If objColumn.IsCalculatedCategory = True Then
      strCategory = ""
   Else
      strCategory = "not"
   End If
   MsgBox "The " & objColumn.Name & " category
is a " & _
      strDrill & " drill-down and is " & strCategory
& _
      " a calculated category." & chr$(10) &
chr$(10) & _
      "It is a member of the " & strLevel & "
level."
   Set objColumn = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "Levels Method" on page 169

LevelList Property

Returns the list of levels for a specified drill-down path.

Syntax

AdvancedQuery.**LevelList**

Applies To

AdvancedQuery Object

Discussion

Use this property to display a list of levels that you specified in the AdvancedQuery subset definition using the Levels method.

When users drill down on a dimension, they drill down on categories from one level to another. The Levels method returns objects that represent all the levels available in a dimension, starting from the top-level category, for a specified path. You can then use this property to display each of these categories. For example, you may have the following categories at the Country or Region level belonging to Europe:

- Belgium
- Germany
- Spain
- Sweden
- United Kingdom
- France

- Italy

The Levels method returns the categories at this level; however, it is the LevelList property that displays the levels specified within the subset definition.

Type

String

Access

Read

Example

This example creates an AdvancedQuery (type 3) subset definition that retrieves all categories at the Country or Region level belonging to Europe. The resulting subset is then added to the report as rows. The MsgBox uses the LevelList property to display the levels specified within the subset definition.

```
Sub Main()
   Dim strCubePath As String
   Dim objPPRep As Object
   Dim objAdvanced As Object
   strCubePath = "C:\Cubes and Reports\Great Outdoors.mdc"
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.New strCubePath, 1
   objPPRep.ExplorerMode = False
   objPPRep.Visible = True
   Set objAdvanced = objPPRep.ReportQueries.Add(3)
   With objAdvanced
      .Name = "European Countries"
      .Dimension = "Locations"
      .Level "Country"
      .Include "Europe"
      .Execute
      .AddToReport 1,1,3
   End With
   Msgbox "Name:  " & objAdvanced.Name & chr$(10)
& _
      "Dimension:  " & objAdvanced.Dimension &
chr$(10) & _
      "Level List:  " & objAdvanced.LevelList &
chr$(10) & _
      "Query Type Code:  " & objAdvanced.Type &
chr$(10) & _
      "Number of Categories:  " & objAdvanced.Count
& _
      chr$(10) & _
      "First Category:  " & objAdvanced.Item(1).Name,
_
      ,"Subset"
   Set objAdvanced = Nothing
```

```
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "Levels Method" on page 169

LevelsDown Property

Sets the number of levels down the hierarchy for specifying the next level ParentageQuery subsets.

Syntax

ParentageQuery.**LevelsDown**

Applies To

ParentageQuery Object

Discussion

Use this property to identify the level below the current dimension for specifying the subset definition for a ParentageQuery. The level specified in the subset definition determines the categories for the subset. If the LowestLevel property is True, do not include the LevelsDown property in the subset definition, because a True value indicates that it is already at the lowest possible level.

The order of the components in the subset definition is important. First, specify the Category method, followed by the LevelsDown and LowestLevel properties, and then the Execute and AddToReport methods. Set the Name property anywhere within the subset definition.

Type

Integer

Access

Write

Example

This example creates a ParentageQuery (type 2) subset definition that returns all categories one level below Channels. Then the categories that are one level below Channels are added to the report as the first nesting level of rows.

```
Sub Main()
   Dim strCubePath As String
   Dim objPPRep As Object
   Dim objCategory As Object
   Dim objParent As Object
   strCubePath = "C:\Cubes and Reports\Great Outdoors.mdc"
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.New strCubePath, 1
   objPPRep.ExplorerMode = False
```

```
   objPPRep.Visible = True
   Set objParent = objPPRep.ReportQueries.Add(2)
   With objParent
      .Name = "Sales Channels"
      .Category "Channels"
      .LowestLevel = False
      .LevelsDown = 1
      .Execute
      .AddToReport 0,1,6
   End With
   Msgbox "The first category added was " & _
      objParent.Item(1).Name & ".", ,"Subset"
   Set objParent = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "LowestLevel Property" on page 338
- "ParentageQuery Object" on page 31

LogonPrompt Property

Sets or returns whether the application prompts for logon or security information.

Syntax

Application.**LogonPrompt**

Applies To

Application Object

Discussion

Set to True to display a standard logon dialog when a user opens a cube or report with security settings or that requires a username and password to access a database.

If the cube or report being opened is secure and the user has more than one user class, the user class that the user wishes to use will also be prompted for.

Default: True

Type

Boolean

Access

Read/Write

Example

This example sets whether the user is prompted with a logon dialog when a cube or report requires user authentication. The default is true (-1), and causes the authentication prompts to be displayed. If you set this property to false (0), an error message displays stating that the user is not authorized to open the datacube.

```
Sub Main()
   Dim objPPRep As Object
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.Application.LogonPrompt False
   objPPRep.Open "C:\Cubes and Reports\Sample1.ppx"
   Set objPPRep = Nothing
End Sub
```

LowerBoundary Property

Sets or returns the value defined for the lower boundary range of the Range object.

Syntax

Range.**LowerBoundary**

Applies To

Range Object

Discussion

Use this method to determine the lower boundary to apply formatting when the information in the report meets the conditions set by the exception range. The lower boundary sets the minimum value for the range. For example, you may want to highlight sales when they reach at least $50,000.

Use the UpperBoundary property to determine the upper boundary of the range.

Type

Variant

Access

Read/Write

Example

This example returns the value defined for the lower boundary of the range in an Exception object.

```
Sub Main
   Dim objPPRep As Object
   Dim objPPRange As Object
   Set objPPRep = GetObject("C:\Cubes and Reports\Exception.ppx")
   objPPRep.Visible = 1
   Set objPPRange = objPPRep.Exceptions.item(1).Ranges.Item(1)
   MsgBox "Lower boundary is " &objPPRange.LowerBoundary
```

```
   Set objPPRange = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "Range Object" on page 35

LowestLevel Property

Sets whether the query uses the next lower level or lowest level of the parent category.

Syntax

ParentageQuery.**LowestLevel**

Applies To

ParentageQuery Object

Discussion

Use this property to specify whether a subset is based on the lowest level for the specified category. If True, it indicates that the query uses the lowest level of the parent category. If False, use the LevelsDown property to specify how far down the hierarchy to go for the level. False indicates that it uses the next level ParentageQuery. If True, the LevelsDown property is not required in the subset definition, because it indicates that it is already at the lowest possible level.

Default: True

The order of the components in the subset definition is important. First, specify the Category method, followed by the LevelsDown and LowestLevel properties, and then the Execute and AddToReport methods. Set the Name property anywhere within the subset definition.

Type

Boolean

Access

Write

Example

This example creates a ParentageQuery (type 2) subset definition that returns all categories one level below Channels. Then the categories that are one level below Channels are added to the report as the first nesting level of rows.

```
Sub Main()
   Dim strCubePath As String
   Dim objPPRep As Object
   Dim objCategory As Object
   Dim objParent As Object
```

```
   strCubePath = "C:\Cubes and Reports\Great Outdoors.mdc"
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.New strCubePath, 1
   objPPRep.ExplorerMode = False
   objPPRep.Visible = True
   Set objParent = objPPRep.ReportQueries.Add(2)
   With objParent
      .Name = "Sales Channels"
      .Category "Channels"
      .LowestLevel = False
      .LevelsDown = 1
      .Execute
      .AddToReport 0,1,6
   End With
   Msgbox "The first category added was " & _
      objParent.Item(1).Name & ".", ,"Subset"
   Set objParent = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "LevelsDown Property" on page 335
- "ParentageQuery Object" on page 31

MacroName Property

Sets or returns the name of the macro associated with an Exception object.

Syntax

Exception.**MacroName**

Applies To

Exception Object

Discussion

To define the initial value of this property, use the SetMacro method. This avoids defining a MacroName without associating a style with it.

Type

String

Access

Read/Write

Example

This example determines the name of the macro associated with an Exception object.

```
Sub Main()
   Dim objPPRep as Object
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.Open "C:\Cubes and Reports\Exception.ppx"
   MsgBox "Driving Category:" & _
      objPPRep.Exceptions.Item(1).DrivingCategory
   MsgBox "Driving Dimension:" & _
      objPPRep.Exceptions.Item(1).DrivingDimension
   MsgBox "The name of the macro associated with the
" & _
      "Exception is " &objPPRep.Exceptions.Item(1).MacroName
   MsgBox "The style of the macro associated with the
" & _
      "Exception is " &objPPRep.Exceptions.Item(1).MacroStyle
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "Exception Object" on page 22

MacroStyle Property

Sets or returns the name of the style associated with the macro used by an Exception object.

Syntax

Exception.**MacroStyle**

Applies To

Exception Object

Discussion

To define the initial value of this property, use the SetMacro method. This avoids associating a style with a non-existent macro.

Type

String

Access

Read/Write

Example

This example determines the style associated with a macro for an Exception object.

```
Sub Main()
   Dim objPPRep as Object
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.Open "C:\Cubes and Reports\Exception.ppx"
   MsgBox "Driving Category:" & _
      objPPRep.Exceptions.Item(1).DrivingCategory
   MsgBox "Driving Dimension:" & _
      objPPRep.Exceptions.Item(1).DrivingDimension
   MsgBox "The name of the macro associated with the
" & _
      "Exception is " &objPPRep.Exceptions.Item(1).MacroName
   MsgBox "The style of the macro associated with the
" & _
      "Exception is " &objPPRep.Exceptions.Item(1).MacroStyle
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "Exception Object" on page 22
- "SetMacro Method" on page 235

MaximumNumberOfRanges Property

Returns the maximum number of Range objects definable for an Exception object.

Syntax

Ranges.**MaximumNumberOfRanges**

Applies To

Ranges

Discussion

Use this property to determine the maximum number of ranges that you can define for an exception. Currently an Exception object can have a maximum of 5 ranges. Therefore this property always returns 5.

Type

Long

Access

Read

Example

This example shows the maximum number of ranges definable for an exception.

```
Sub Main
   Dim objPPRep As Object
   Dim objPPRange As Object
   Set objPPRep = GetObject(,"CognosPowerPlay.Report")
   Set objPPRange = objPPRep.Exceptions.item(1).Ranges
   MsgBox "Maximum Number of Ranges = " & _
         objPPRange.MaximumNumberOfRanges
   Set objPPRange = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "Exception Object" on page 22
- "Range Object" on page 35

MaxPrintedBars Property

Sets or returns the maximum number of bars on a single printed page.

Syntax

object.**MaxPrintedBars**

Applies To

Graph Object

Graphs

Discussion

Use this property to set the maximum number of bars to print on each page of the display.

This property is not available for crosstab, 3-D bar, scatter, and pie displays.

Default: 8

Type

Long

Access

Read/Write (Graph)

Write (Graphs)

Example

This example changes the scroll bar settings of an active report. The scrolling feature is set to True, maximum visible bars is set to six, maximum printed bars is set to ten, and the summary category is kept hidden until the end of the report.

```
Sub Main()
   Dim objPPRep As Object
   Set objPPRep = GetObject(,"CognosPowerPlay.Report")
   objPPRep.Graphs.Item(1).UseScrolling = True
   objPPRep.Graphs.Item(1).MaxVisibleBars = 6
   objPPRep.Graphs.Item(1).MaxPrintedBars = 10
   objPPRep.Graphs.Item(1).KeepSummaryVisible = False
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "Graph Object" on page 25
- "Graphs" on page 56

MaxVisibleBars Property

Sets or returns the maximum number of bars visible on a single page of scrolled data.

Syntax

object.**MaxVisibleBars**

Applies To

Graph Object

Graphs

Discussion

Use this property to set the number of bars visible on each page of the display.

You must specify a minimum value between 5 and 50.

This property is not available for crosstab, 3-D bar, scatter, and pie displays.

Default: 8

Type

Long

Access

Read/Write (Graph)

Write (Graphs)

Example

This example changes the scroll bar settings of an active report. The scrolling feature is set to True, maximum visible bars is set to six, maximum printed bars is set to ten, and the summary category is kept hidden until the end of the report.

```
Sub Main()
   Dim objPPRep As Object
   Set objPPRep = GetObject(,"CognosPowerPlay.Report")
   objPPRep.Graphs.Item(1).UseScrolling = True
   objPPRep.Graphs.Item(1).MaxVisibleBars = 6
   objPPRep.Graphs.Item(1).MaxPrintedBars = 10
   objPPRep.Graphs.Item(1).KeepSummaryVisible = False
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "Graph Object" on page 25
- "Graphs" on page 56

Measure Property

Sets or returns the name of measure whose values are used for a value restriction.

Syntax

ValueRestriction.**Measure**

Applies To

ValueRestriction Object

Discussion

Use this property to specify the name of a measure that a ValueRestriction object uses to limit the number of categories for the results of an AdvancedQuery.

For a ValueRestriction, the order of the components is important. The Dimension property must be set before the Measure, Operator, Operand1, Operand2, and Count properties and the DimensionFilter method. The Name property can be set anywhere within the filter definition.

Type

String

Access

Read/Write

Example

This example creates an advanced subset that selects countries or regions from the Locations dimension. The value restriction (type 4) limits the results to return only those countries or regions whose Revenue Values for Sports Chains are between 25,000 and 100,000.

```
Sub Main()
   Dim strCubePath As String
   Dim objPPRep As Object
   Dim objValue As Object
   Dim objAdvanced As Object
   strCubePath = "C:\Cubes and Reports\Great Outdoors.mdc"
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.New strCubePath, 1
   objPPRep.ExplorerMode = False
   objPPRep.Visible = True
   Set objValue = objPPRep.ReportQueries.Add(4)
   With objValue
      .Name = "25000-100000"
      .Dimension = "Locations"
      .Measure = "Revenue"
      .Operator = "between"
      .Operand1 = 25000
      .Operand2 = 100000
      .DimensionFilter 4, "Sports Chain"
   End With
   Set objAdvanced = objPPRep.ReportQueries.Add(3)
   With objAdvanced
      .Name = "Locations"
      .Dimension = "Locations"
      .Level "Country or Region"
      .ValueRestriction objValue.Name
      .Execute
      .AddToReport 0,1,3
   End With
   Set objAdvanced = Nothing
   Set objValue = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- Chapter 4, “Methods,” on page 71
- Chapter 5, “Properties,” on page 259

MeasureCurrency Property

Sets or returns the value and symbol for a specified currency.

Syntax

Dimension.**MeasureCurrency**

Applies To

Dimension Object

Discussion

In IBM Cognos PowerPlay, currency values can be displayed in many of the principal currencies of global commerce. Use MeasureCurrency to change the basic currency used in monetary values and, where applicable, the currency symbol. Type the applicable three-character code as a string parameter.

When used with an empty parameter, this property returns the current code.

When you change the value of MeasureCurrency, PowerPlay recomputes the monetary values using an exchange-rate table supplied by the cube designer via PowerPlay Transformer. For example, if the cube shows currency in dollars and you use the code for Great Britain, "gbr", the cube recomputes the value, switches to values in pounds, and then replaces the dollar sign with the pound currency symbol.

The three-letter country or region codes are based on an international standard, ISO/IEC (International Standards Organization, International Electrotechnical Commission) specification 3166. For example, the country or region codes for Sweden, the United States, and Spain are "SWE", "USA", and "ESP", respectively.

The displayed currencies are limited to cube population.

Type

String

Access

Read/Write

Example

This example checks the present currency code and changes "USA" to "CAN."

```
Sub Main()
   Dim objPPRep As Object
   Dim objDimension As Object
   Set objPPRep = GetObject(, "CognosPowerPlay.Report")
   Set objDimension = objPPRep.DimensionLine.Item _
      ("Measures")
   objDimension.ChangeToTop
   objDimension.Change "Product Cost"
   If objDimension.MeasureCurrency = "USA" Then
      objDimension.MeasureCurrency = "CAN"
      MsgBox "The Currency has been changed from" &
_
```

```
         " American to Canadian."
   Else
      MsgBox "The currency code is " & _
         objDimension.MeasureCurrency & "."
   End If
   Set objDimension = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- Chapter 4, “Methods,” on page 71
- Chapter 5, “Properties,” on page 259

Name Property

Sets or returns the name of the object.

Syntax

object.**Name**

Applies To

AdvancedQuery Object

Application Object

Column Object

Dimension Object

Exception Object

FindQuery Object

Layer Object

ParentageQuery Object

Report Object

Row Object

ValueRestriction Object

Discussion

Change the name of a Column, Exception, Layer, or Row object using an equal sign (=). The name of a Report object can be modified only by using the SaveAs method. You cannot alter the name of the Application, Dimension, or Exception object.

For an AdvancedQuery, FindQuery, or ParentageQuery, use the Name property to specify the name for the query.

For the Name property associated with a ValueRestriction, do not include functions applied to measures. For example, "Average(Revenue)>500" and "Revenue - Cost in Top 10" are not valid. For a Value Restriction query, only specify one value restriction name at a time. For example, "Revenue> 10,000 and Units Sold < 30" is not a valid name for the restriction.

Type

String

Access

Read (Application, Dimension, Exception, Report)

Read/Write (AdvancedQuery, Column, FindQuery, Layer, ParentageQuery, Row, Value Restriction)

Example

This example creates an instance of IBM Cognos PowerPlay and shows the name, location, and version of the application.

```
Sub Main()
   Dim objPPlayApp as Object
   Set objPPlayApp = CreateObject("CognosPowerPlay.Application")
   objPPlayApp.Visible = 1
   MsgBox "The name of the Application is " &objPPLayApp.Name
   MsgBox "The location of the Application is " _
      &objPPLayApp.Path
   MsgBox "The Application version is " &objPPLayApp.Version
   Set objPPlayApp = Nothing
End Sub
```

Related Topics

- "Application Object" on page 11
- "Column Object" on page 15
- "Dimension Object" on page 19
- "Layer Object" on page 28
- "Report Object" on page 37
- "Row Object" on page 41

NamesShown Property

Sets or returns whether category names appear beside pie chart slices.

Syntax

object.**NamesShown**

Applies To

Graph Object

Graphs

Discussion

This property applies to pie charts only. Set this property to True to label slices by using a category name instead of a category value. However, the legend includes both the category names and values.

If you set the NestedCharts property to True for a single pie chart, the pie will automatically change so that the slice labels are category names instead of values.

Setting the NamesShown property to True automatically sets the ValuesShown property to False.

If you set NamesShown to False for nested charts, no labels appear beside pie chart slices.

Default: False (single charts), True (nested charts)

Type

Boolean

Access

Read/Write (Graph)

Write (Graphs)

Example

This example sets the type for the active graph to a pie chart and uses category names to label the pie chart slices.

```
Sub Main
   Dim objPPRep as Object
   Dim objPPGraph as Object
   Set objPPRep = GetObject(,"CognosPowerPlay.Report")
   Set objPPGraph = objPPRep.Graphs.Item(1)
   With objPPGraph
      .SetType 1
      .NamesShown = True
      .ValuesFontColor = 10
      .ValuesFontSize = 10
      .ValuesFontName = "Times New Roman"
   End With
   Set objPPGraph = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "ValuesShown Property" on page 428)

NestedCharts Property (Explorer)

Sets or returns whether multiple charts that represent summarized data appear in one display.

Syntax

Graph.**NestedCharts**

Applies To

Graph Object

Discussion

Use this property to view multiple charts of summarized data in one display. You can show comparisons, relationships, and trends of the report dimensions efficiently, especially if you have a large amount of summary data. For example, use this property for a nested report that generates one single chart for each summary row on ten different displays, to generate a display with multiple charts that show the same information.

If you set this property to True for data that is not nested, the data will show in a single chart, but in nested chart format with row and column labels.

Nested charts plot all the categories that belong in the chart, unlike single charts where you can specify a number by using the MaxVisibleBars property.

When you set this property to True, properties or methods that apply to only single charts are ignored. These properties include

- KeepSummaryVisible
- MaxPrintedBars
- MaxVisibleBars
- ShowSummaryColumn
- ShowSummaryRow
- ShowTies
- UseScrolling
- ValuesShown

You can use this property with the SetType method to select the type of charts to show. This property is not shown for crosstabs.

This property is available in Explorer mode only.

Default: False

Type

Boolean

Access

Read/Write

Example

This example sets the display of the active graph to show multiple charts.

```
Sub Main
   Dim objPPRep as Object
   Dim objPPGraph as Object
   Set objPPRep = GetObject(,"CognosPowerPlay.Report")
   Set objPPGraph = objPPRep.Graphs.Item(1)
   With objPPGraph
      .SetType 3
      .HideRankCategory = False
      .KeepSummaryVisible = True
      .MaxVisibleBars = 5
      .ShowTies = True
   End With
   objPPGraph.NestedCharts = True
   objPPRep.SaveAs "NewReport.ppx"
   Set objPPGraph = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "NamesShown Property" on page 348
- "ValuesShown Property" on page 428

NestedName Property

Returns the nested name for a category.

Syntax

object.**NestedName**

Applies To

Column Object

Layer Object

Row Object

Discussion

Use this property to return a name for a category that takes nesting into account. For example, for the following nested report, the name of the second row is Europe, and the nested name for that row is 2008.Europe.

Year	**Region**
2008	Far East

Year	Region
	Europe
	Middle East

Type

String

Access

Read

Example

This example returns the nested name for the first row, column, and layer within an active report.

```
Sub Main()
   Dim objPPRep As Object
   Set objPPRep = GetObject(,"CognosPowerPlay.Report")
   Msgbox "First column:  " & _
      objPPRep.Columns.Item(1).NestedName
   Msgbox "First row:  " & _
      objPPRep.Rows.Item(1).NestedName
   Msgbox "First layer:  " & _
      objPPRep.Layers.Item(1).NestedName
   Set objPPRep = Nothing
End Sub
```

Related Topics

- Chapter 4, "Methods," on page 71
- Chapter 5, "Properties," on page 259

Operand1 Property

Sets or returns the value used to compare report cell values based on a specified operator.

Syntax

ValueRestriction.**Operand1**

Applies To

ValueRestriction Object

Discussion

Use this property to specify the value used to limit the number of categories for the results of an AdvancedQuery.

When using the Between operator type, you must specify values for the Operand1 and Operand2 properties.

For a ValueRestriction, the order of the components is important. The Dimension property must be set before the Measure, Operator, Operand1, Operand2, and Count properties and the DimensionFilter method. The Name property can be set anywhere within the filter definition.

Do not specify Operand1 if the operator is Largest or Smallest. When you use the Largest or Smallest operator, only the Count property is used. The default value for Count is 10.

Type

Double

Access

Read/Write

Example

This example creates an advanced subset that selects countries or regions from the Locations dimension. The value restriction (type 4) limits the results to return the countries or regions whose sales revenue exceed 125,000.

```
Sub Main()
   Dim strCubePath As String
   Dim objPPRep As Object
   Dim objValue As Object
   Dim objAdvanced As Object
   strCubePath = "C:\Cubes and Reports\Great Outdoors.mdc"
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.New strCubePath, 1
   objPPRep.ExplorerMode = False
   objPPRep.Visible = True
   Set objValue = objPPRep.ReportQueries.Add(4)
   With objValue
      .Name = "Revenue exceeding 125000"
      .Dimension = "Locations"
      .Measure = "Revenue"
      .Operator = ">"
      .Operand1 = 125000
   End With
   Set objAdvanced = objPPRep.ReportQueries.Add(3)
   With objAdvanced
      .Name = "Locations"
      .Dimension = "Locations"
      .Level "Country or Region"
      .ValueRestriction objValue.Name
      .Execute
      .AddToReport 0,1,3
```

```
    End With
    Set objAdvanced = Nothing
    Set objValue = Nothing
    Set objPPRep = Nothing
End Sub
```

Related Topics

- "Operand2 Property"
- "Operator Property" on page 355

Operand2 Property

Sets or returns the second value when the Between operator is used to specify a range.

Syntax

ValueRestriction.**Operand2**

Applies To

ValueRestriction Object

Discussion

Use this property to specify the second value when you use the Between operator to limit the number of categories for the results of an AdvancedQuery.

When you use the Between operator type, you must specify values for the Operand1 and Operand2 properties. If the Between operator type is not specified, any value assigned to this property is ignored.

For a ValueRestriction, the order of the components is important. The Dimension property must be set before the Measure, Operator, Operand1, Operand2, and Count properties and the DimensionFilter method. The Name property can be set anywhere within the filter definition.

Type

Double

Access

Read/Write

Example

This example creates an advanced subset that selects countries or regions from the Locations dimension. The value restriction (type 4) limits the results to return only those countries or regions whose Revenue values for Sports Chains are between 25,000 and 100,000.

```
Sub Main()
    Dim strCubePath As String
    Dim objPPRep As Object
```

```
   Dim objValue As Object
   Dim objAdvanced As Object
   strCubePath = "C:\Cubes and Reports\Great Outdoors.mdc"
      Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.New strCubePath, 1
   objPPRep.ExplorerMode = False
   objPPRep.Visible = True
   Set objValue = objPPRep.ReportQueries.Add(4)
   With objValue
      .Name = "25000-100000"
      .Dimension = "Locations"
      .Measure = "Revenue"
      .Operator = "between"
      .Operand1 = 25000
      .Operand2 = 100000
      .DimensionFilter 4, "Sports Chain"
   End With
   Set objAdvanced = objPPRep.ReportQueries.Add(3)
   With objAdvanced
      .Name = "Locations"
      .Dimension = "Locations"
      .Level "Country or Region"
      .ValueRestriction objValue.Name
      .Execute
      .AddToReport 0,1,3
   End With
   Set objPPRep = Nothing
   Set objAdvanced = Nothing
   Set objValue = Nothing
End Sub
```

Related Topics

- "Operand1 Property" on page 352
- "Operator Property"

Operator Property

Sets or returns the operator used for a value restriction.

Syntax

ValueRestriction.**Operator**

Applies To

ValueRestriction Object

Discussion

Use this property to specify the operator used to limit the number of categories for the results of an AdvancedQuery.

Operator types available for a value restriction comparison are

- > (greater than)
- < (less than)
- = (equal to)
- >= (greater than or equal to)
- <= (less than or equal to)
- Largest (top n largest values)
- Smallest (bottom n smallest values)
- Between (values within to a specified range)

When using the Between operator, you must specify values for the Operand1 and Operand2 properties. When using the >, <, >=, or <= operator, you must specify a value for the Operand 1 property. When using the Largest or Smallest operator, you must specify a value for the Count property.

For a ValueRestriction, the order of the components is important. The Dimension property must be set before the Measure, Operand1, Operand2, and Count properties and the DimensionFilter method. The Name property can be set anywhere within the filter definition.

Type

String

Access

Read/Write

Example

This example creates an advanced subset that selects countries or regions from the Locations dimension. The value restriction (type 4) limits the results to return only those countries or regions whose Revenue values for Sports Chains are between 25,000 and 100,000.

```
Sub Main()
   Dim strCubePath As String
   Dim objPPRep As Object
   Dim objValue As Object
   Dim objAdvanced As Object
   strCubePath = "C:\Cubes and Reports\Great Outdoors.mdc"
      Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.New strCubePath, 1
   objPPRep.ExplorerMode = False
   objPPRep.Visible = True
   Set objValue = objPPRep.ReportQueries.Add(4)
   With objValue
      .Name = "25000-100000"
      .Dimension = "Locations"
      .Measure = "Revenue"
      .Operator = "between"
      .Operand1 = 25000
```

```
      .Operand2 = 100000
      .DimensionFilter 4, "Sports Chain"
   End With
   Set objAdvanced = objPPRep.ReportQueries.Add(3)
   With objAdvanced
      .Name = "Locations"
      .Dimension = "Locations"
      .Level "Country or Region"
      .ValueRestriction objValue.Name
      .Execute
      .AddToReport 0,1,3
   End With
   Set objPPRep = Nothing
   Set objAdvanced = Nothing
   Set objValue = Nothing
End Sub
```

Related Topics

- "Operand1 Property" on page 352
- "Operand2 Property" on page 354

ParentCategory Property

Returns the name of the parent category for the object.

Syntax

object.**ParentCategory**

Applies To

Column Object

Layer Object

Row Object

Discussion

This property is used during drill up to get the name of the parent category of a row, column, or layer. This property returns an empty string if it is a calculation. The property returns an error if the category is the top level category.

In Reporter reports, this property returns a blank for complex categories such as intersection, rank, and calculation categories where a single rollup parent is unavailable.

In Explorer reports, the parent category name returned for any row or column item applies to all rows and columns.

Type

String

Access

Read

Example

This example returns the parent name of the first row and the parent name of the first column of the currently open report.

```
Sub Main()
   Dim objPPRep As Object
   Dim objRows As Object
   Dim objCols As Object
   Set objPPRep = GetObject (,"CognosPowerPlay.Report")
   objPPRep.Visible = True
   Set objRows = objPPRep.Rows
   Set objCols = objPPRep.Columns
   MsgBox "Row 1's parent is:" _
      &objRows.Item(1).ParentCategory
   MsgBox "Column 1's parent is:" _
      &objCols.Item(1).ParentCategory
  Set objRows = Nothing
  Set objCols = Nothing
  Set objPPRep = Nothing
End Sub
```

Related Topics

- "Column Object" on page 15
- "Layer Object" on page 28
- "Row Object" on page 41

Path Property

Returns the path of the Report object or the Application object.

Syntax

object.**Path**

Applies To

Application Object

Report Object

Discussion

The path of an Application object cannot be changed. The path of a Report object can be changed, but only using the SaveAs method.

Type

String

Access

Read

Example

This example creates an instance of IBM Cognos PowerPlay and shows the name, location, and version of the application.

```
Sub Main()
   Dim objPPlayApp as Object
   Set objPPlayApp = CreateObject("CognosPowerPlay.Application")
   objPPlayApp.Visible = 1
   MsgBox "The name of the Application is " &objPPLayApp.Name
   MsgBox "The location of the Application is " _
      &objPPLayApp.Path
   MsgBox "The Application version is " &objPPLayApp.Version
   Set objPPlayApp = Nothing
End Sub
```

Related Topics

- "Application Object" on page 11
- "Report Object" on page 37

Pattern Property

Sets search criteria for a subset definition.

Syntax

FindQuery.**Pattern**

Applies To

FindQuery Object

Discussion

Use this property to control how the SearchText property is used by a subset definition. With Pattern, you can determine if the

- search string is found anywhere in a word, at the start of a word only, or the end of a word only
- pattern matching is on or off
- search text is to match the whole word or any part of the word
- search is case sensitive

Each of the search options has a numeric value. For example, matching the text case equals 32, while matching the text with the end of the word equals 4. Add the different search option values to provide a unique value that sets the overall search options.

Unless a pattern value is given, the search option is not in effect. For example, to make a search case sensitive, the value 32 must be included. Add the values to set various search options. Adding values that cannot be used together causes an

exception error. For example, do not combine the Contains, Begins With, or Ends With values because they are mutually exclusive, and do not use any of these with MatchWhole. Also, do not use Pattern Matching with MatchCase or MatchWhole.

When the Pattern Matching value is used (Pattern = 8), the Find operation recognizes some characters in the text string as wildcards and some metacharacters are treated as reserved characters when Pattern Matching is used. If the metacharacters are included, an error message appears. If pattern matching is not used, wildcards and metacharacters are treated as normal characters.

Examples of valid combinations include

- Begins With + Pattern Matching = (10)
- MatchCase + Contains = (33)
- MatchCase + Begins With = (34)
- MatchCase + Ends With = (36)
- MatchCase + Use Patterns = (40)
- MatchCase + MatchWhole = (48)

The order of the components in the subset definition are important. First, set the Dimension property, followed by the SearchShortName, SearchText, and Pattern properties, and then the Execute and AddToReport methods. Set the Name property anywhere within the subset definition.

Parameter	Description
PatternValue	Required. Specifies a numeric value that determines the search options. It defaults to 1. 1 - Contains: the search text can be found anywhere in a word 2 - Begins With: the search text must be found at the start of a word 4 - Ends With: the search text must be found at the end of a word 8 - Pattern Matching: certain characters are treated as wildcards 16 - MatchWhole: the search text must match the whole word 32 - MatchCase: the search is case sensitive

Type

Integer

Access

Write

Wildcard Characters for Pattern Matching

Wildcard	Description
^	Match anchored at beginning of string (^ball matches balloon but not baseball).

Wildcard	Description
$	Match anchored at end of string (^ball matches baseball but not balloon).
?	Match any character
#	Match zero or occurrences of the preceding character (or sub-expression)
@	Match one or more occurrences of the preceding character (or sub-expression)
~	Match zero or one occurrences of the preceding character (or sub-expression)
*	Matches zero or more occurrences of any characters.
|	Match either the preceding character (or sub-expression) or the following one; for example, abc|def matches either abcef or abdef.
[ ] [-] [!]	Match any character within the brackets; for example, [aeiou] matches a, e, i, o, u. A hyphen between two characters within the brackets indicates a range; for example, [am-px] matches a, m through p and x. A hyphen at the start or end matches itself; for example, [-] matches a hyphen. An exclamation point at the beginning causes the set of characters to be inverted; for example, [!a-m] matches everything except a through m.
( )	Sub-expression. Enables nesting of an expression within the expression, so that repetition and alternative operators can be applied more generally. For example, ab(cd)#e matches a followed by b followed by zero or more cd combinations followed by e.
\	Escape. Match the next character literally. Generally used to allow metacharacters to be treated as normal characters; for example, \? Matches ?.

Metacharacters

Metacharacter	Description
<	Match the beginning of a word.

Metacharacter	Description
>	Match the end of a word.
{m,n}	Match at least m and no more than n occurrences of the preceding character (or sub-expression). {n} matches n times. {n,} matches at least n times.
\:x	Extended metacharacter x (where x represents any letter or digit).

Example

This example creates a FindQuery (type 1) subset definition that searches for all products that begin with the name Star.

```
Sub Main()
   Dim strCubePath As String
   Dim objPPRep As Object
   Dim objFind As Object
   strCubePath = "C:\Cubes and Reports\Great Outdoors.mdc"
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.New strCubePath, 1
   objPPRep.ExplorerMode = False
   objPPRep.Visible = True
   Set objFind = objPPRep.ReportQueries.Add(1)
   With objFind
      .Name = "Find Star"
      .Dimension = "Products"
      .SearchShortName = False
      .SearchText = "Star"
      .Pattern = 2
   End With
   Set objFind = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "FindQuery Object" on page 23

Precedence Property

Sets or returns the precedence used in complex calculations.

Syntax

object.**Precedence**

Applies To

Column Object

Layer Object

Row Object

Discussion

For complex calculations (those containing several operations), the operations are performed in a certain order. The precedence rules for IBM Cognos PowerPlay are

1. accumulation and % accumulate
2. a percent, average, minimum, or maximum operation take precedence over division or multiplication
3. division or multiplication take precedence over addition or subtraction
4. addition or subtraction take precedence over non-calculations (such as, categories)
5. where a calculation in a row intersects with a calculation in a column, the row takes precedence
6. layers are calculated after rows and columns

Precedence allows you to set the precedence of calculations. Your choices are "Normal" or "Override." (They are not case-sensitive.)

When precedence is explicitly assigned to a category (Override is set), the cell value no longer changes when the user swaps rows and columns. If used without the switch parameter, it returns the current setting. If Precedence is used with a non-calculation, an error occurs.

You cannot set precedence for business calculations (for accumulation, % accumulate only).

Type

String

Access

Read/Write

Example

This example creates a small report that includes several calculations. The precedence on the "Total" column is set to override any rows that have calculations.

```
Sub Main()
   Dim objPPRep As Object
   Dim objCategoryList As Object
   Dim objIncreaseRow As Object
   Dim objPercentRow As Object
   Dim objSummaryColumn As Object
   Dim intColumns As Integer
```

```
   Const CubePath = "C:\Cubes and Reports\Great Outdoors.mdc"
   Set objPPRep = GetObject(CubePath)
   Set objCategoryList = objPPRep.CategoryList()
   objCategoryList.Add 1, "Years"
   objPPRep.Rows.Add objCategoryList
   objCategoryList.Add 1, "Products"
   objPPRep.Columns.Add objCategoryList
   objPPRep.ExplorerMode = False
   objPPRep.Rows.Item("Years").Activate
   Set objIncreaseRow = objPPRep.Rows.Item _
      ("1997").Subtraction(objPPRep.Rows.Item("1996"))
   objIncreaseRow.Name = "Increase"
   objIncreaseRow.Activate
   Set objPercentRow = objIncreaseRow.Percent _
      (objPPRep.Rows.Item("1996"))
   objPercentRow.Name = "Percent Increase"
   objPPRep.Columns.Item("Products").Remove
   intColumns = objPPRep.Columns.Count
   objPPRep.Columns.Item(intColumns).Activate
   Set objSummaryColumn = _
      objPPrep.Columns.SubSet(1, 3).Addition()
   intColumns = objPPRep.Columns.Count
   objPPRep.Columns.Item(intColumns).Name "Total"
   objPPRep.Columns.Item(intColumns).Precedence = _
      "Override"
   objPPRep.SaveAs "C:\Cubes and Reports\Precedence.ppx"
   Set objSummaryColumn = Nothing
   Set objPercentRow = Nothing
   Set objIncreaseRow = Nothing
   Set objCategoryList = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "Accumulation Method" on page 78
- "Addition Method (Collections)" on page 95
- "Division Method" on page 136
- "Subtraction Method (Collections)" on page 244

PrintAllCharts Property

Sets or returns whether all displays print on the same page.

Syntax

Print.**PrintAllCharts**

Applies To

Print Object

Discussion

If True, all displays visible in page layout or page width view print on the same page. If False, the selected or currently active Graph object prints on as many pages as required.

This property must be set to False when used with the following properties:

- AxisOnAllPages
- ChartTitleOnAllPages
- IncludeLegend
- SetChartToPrint
- SummariesOnAllPages

Default: True

Type

Boolean

Access

Read/Write

Example

This example opens a report and prints one copy of all the data for the second graphical display only. This example includes the display title, summary category, and axis on all pages; turns on the collating option; and excludes the legend.

```
Sub Main()
   Dim objPPRep as Object
   Dim objRepPrt as Object
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.Open "C:\Cubes and Reports\Sample2.ppx"
   Set objRepPrt = objPPRep.Print
   objRepPrt.PrintAllCharts = False
   objRepPrt.SetListOfRowsToPrint objPPrep.Rows
   objRepPrt.SetListOfLayersToPrint objPPRep.Layers
   objRepPrt.SetChartToPrint objPPRep.Graphs.Item(2)
   objRepPrt.IncludeLegend = True
   objRepPrt.ChartTitleOnAllPages = True
   objRepPrt.SummariesOnAllPages = True
   objRepPrt.AxisOnAllPages = True
   objRepPrt.Collate = True
   objRepPrt.Copies = 1
   objRepPrt.PrintOut
   Set objRepPrt = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- Chapter 4, "Methods," on page 71
- Chapter 5, "Properties," on page 259
- "Print Object" on page 33

PrintColorsAsPatterns Property

Sets or returns whether colors print as patterns or as colors.

Syntax

Print.**PrintColorsAsPatterns**

Applies To

Print Object

Discussion

If True, colors print as patterns. If False, colors print as they appear (if you have a color printer).

Default: False

Type

Boolean

Access

Read/Write

Example

This example opens a report and prints one copy of all displays in the report including all rows and layers. It also includes the legend, prints colors as patterns, and turns on the collating option.

```
Sub Main()
   Dim objPPRep as Object
   Dim objRepPrt as Object
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.Open "C:\Cubes and Reports\Sample2.ppx"
   Set objRepPrt = objPPRep.Print
   objRepPrt.PrintAllCharts = False
   objRepPrt.SetListOfRowsToPrint objPPrep.Rows
   objRepPrt.SetListOfLayersToPrint objPPRep.Layers
   objRepPrt.SetChartToPrint objPPRep.Graphs.Item(2)
   objRepPrt.IncludeLegend = True
   objRepPrt.PrintColorsAsPatterns = True
   objRepPrt.ChartTitleOnAllPages = True
   objRepPrt.SummariesOnAllPages = True
   objRepPrt.AxisOnAllPages = True
```

```
   objRepPrt.Collate = True
   objRepPrt.Copies = 1
   objRepPrt.PrintOut
   Set objRepPrt = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "Print Object" on page 33

PrintEntireReport Property

Sets or returns whether to print the entire report, including all displays, layers, and rows.

Syntax

Print.**PrintEntireReport**

Applies To

Print Object

Discussion

Print options are initialized to the options saved with the report. Unless a print macro overrides a property, its state will correspond to the way in which the author saved the report.

Only one of the print properties PrintEntireReport, PrintPageLayout, and PrintSelectedDisplays can be true at one time.

If true, the entire report, including all displays, layers, and rows, will print.

Type

Boolean

Access

Read/Write

Example

This example opens and prints an entire report. It also includes the summary row and column categories on all pages.

```
Sub Main()
   Dim objPPRep as Object
   Dim objRepPrt as Object
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.Open "C:\Cubes and Reports\Sample1.ppx"
   Set objRepPrt = objPPRep.Print
   objRepPrt.PrintEntireReport = True
   objRepPrt.SummaryRowOnAllPages = True
```

```
   objRepPrt.SummaryColumnOnAllPages = True
   objRepPrt.PrintOut
   Set objRepPrt = Nothing
   Set objPPrep = Nothing
End Sub
```

Related Topics

- Chapter 4, “Methods,” on page 71
- Chapter 5, “Properties,” on page 259
- “Print Object” on page 33

PrintPageLayout Property

Sets or returns whether to print all displays visible in the page layout view or page width view on the same page.

Syntax

Print.**PrintPageLayout**

Applies To

Print Object

Discussion

Print options are initialized to the options saved with the report. Unless a print macro overrides a property, its state will correspond to the way in which the author saved the report.

Only one of the print properties PrintEntireReport, PrintPageLayout, and PrintSelectedDisplays can be true at one time.

If true, all displays visible in the page layout or page width view will print on the same page.

Type

Boolean

Access

Read/Write

Example

This example opens and prints all displays visible in the page layout view or page width view on the same page, using the default print settings.

```
Sub Main()
   Dim objPPRep as Object
   Dim objRepPrt as Object
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.Open "C:\Cubes and Reports\Sample1.ppx"
   Set objRepPrt = objPPRep.Print
```

```
   objRepPrt.ResetPrintOptionsToDefault
   objRepPrt.PrintPageLayout = True
   objRepPrt.PrintOut
   Set objRepPrt = Nothing
   Set objPPrep = Nothing
End Sub
```

Related Topics

- Chapter 4, "Methods," on page 71
- Chapter 5, "Properties," on page 259
- "Print Object" on page 33

PrintSelectedDisplay Property

Sets or returns whether to print the selected or currently active Graph object.

Syntax

Print.**PrintSelectedDisplay**

Applies To

Print Object

Discussion

Print options are initialized to the options saved with the report. Unless a print macro overrides a property, its state will correspond to the way in which the author saved the report.

Only one of the print properties PrintEntireReport, PrintPageLayout, and PrintSelectedDisplays can be true at one time.

If true, the selected or currently active Graph object will print.

Type

Boolean

Access

Read/Write

Example

This example opens a report and prints the selected or currently active Graph object.

```
Sub Main()
   Dim objPPRep as Object
   Dim objRepPrt as Object
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.Open "C:\Cubes and Reports\Sample1.ppx"
   Set objRepPrt = objPPRep.Print
```

```
   objRepPrt.PrintSelectedDisplay = True
   objRepPrt.SetChartToPrint objPPRep.Graphs.Item(1)
   objRepPrt.PrintOut
   Set objRepPrt = Nothing
   Set objPPrep = Nothing
End Sub
```

Related Topics

- Chapter 4, "Methods," on page 71
- Chapter 5, "Properties," on page 259
- "Print Object" on page 33

PromptForCurrency Property

Sets or returns whether the report consumer can change the currency in a report published to the IBM Cognos portal

Syntax

DeploymentOptions.**PromptForCurrency**

Discussion

Use this property when the cube on which the report is based supports multiple currencies. By default the report uses the currency or currencies specified by the report author. If the cube supports multiple currencies, but the report does not contain currency measures, the author can still set this property to True.

The consumer cannot set different currencies for different measures.

If this property is set to True, the report consumer can set some formatting features for the currency in the IBM Cognos portal.

An error will occur if the cube doesn't support currencies or if the report contains a mix of currencies.

Default: False

Type

Boolean

Access

Read/Write

Example

This example specifies the prompts that the report consumer sees when the report is opened in the IBM Cognos portal.

```
Sub Main()
   Dim objPPRep as Object
   Dim objDeploymentOptions as Object
   Set objPPRep = GetObject(,"CognosPowerPlay.Report")
```

```
   Set objDeploymentOptions = objPPRep.DeploymentOptions
   objDeploymentOptions.PromptForCurrency = True
   objDeploymentOptions.PromptForLongShortNames = True
   objDeploymentOptions.PromptForZeroSuppression = True
   objDeploymentOptions.PromptForSwapRowsAndColumns =
True
   objPPRep.Save
   Set objDeploymentOptions = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "PromptForDimension Property"
- "PromptForLongShortNames Property" on page 372
- "PromptForZeroSuppression Property" on page 374

PromptForDimension Property

Sets or returns whether a report consumer can filter the specified Dimension object when a report is opened in the IBM Cognos portal.

Syntax

DeploymentOptions.**PromptForDimension**(Index)

Discussion

Use this property to specify on which dimensions a report consumer can filter when a report is opened in the IBM Cognos portal. You can select or clear each dimension, one at a time. To select all dimensions, use the SelectAllDimensions method. To clear all dimensions, use the UnselectAllDimensions method.

By default, the property is set to false for each dimension to prevent a user from filtering on that dimension.

Default: False

Parameters	Description
Index	Required. Specifies the name of the Dimension object or its position in the DimensionLine object. The position in the DimensionLine object starts at 1 and increments by 1. Type: String or Integer

Type

Boolean

Access

Read/Write

Example

This example specifies the dimensions on which the report consumer can filter when the report is opened in the IBM Cognos portal.

```
Sub Main()
   Dim objPPRep as Object
   Dim objDeploymentOptions as Object
   Set objPPRep = GetObject(,"CognosPowerPlay.Report")
   Set objDeploymentOptions = objPPRep.DeploymentOptions
   objDeploymentOptions.PromptForCurrency = True
   objDeploymentOptions.PromptForLongShortNames = True
   objDeploymentOptions.PromptForZeroSuppression = True
   objDeploymentOptions.PromptForSwapRowsAndColumns =
True
   objDeploymentOptions.PromptForDimension(1) = True
   objDeploymentOptions.PromptForDimension(2)= True
   objDeploymentOptions.PromptForDimension("Locations")=
True
   objPPRep.Save
   Set objDeploymentOptions = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "PromptForCurrency Property" on page 370
- "PromptForLongShortNames Property"
- "PromptForZeroSuppression Property" on page 374

PromptForLongShortNames Property

Sets or returns whether the report consumer can change between long and short category names in a report published to the IBM Cognos portal.

Syntax

DeploymentOptions.**PromptForLongShortNames**

Discussion

Use this property when the cube on which the report is based supports long and short category names. By default, the report uses the category name specified by the report author.

This property applies to all dimensions. The consumer cannot use long names for one dimension and short names for another, but the report author can.

You can use this property when the cube supports short names only or long names only.

By default, the report consumer will not be prompted to change between long and short category names.

An error will occur if the report contains some dimensions with long names and some dimensions with short names.

Default: False

Type

Boolean

Access

Read/Write

Example

This example specifies the prompts that the report consumer sees when the report is opened in the IBM Cognos portal.

```
Sub Main()
   Dim objPPRep as Object
   Dim objDeploymentOptions as Object
   Set objPPRep = GetObject(,"CognosPowerPlay.Report")
   Set objDeploymentOptions = objPPRep.DeploymentOptions
   objDeploymentOptions.PromptForCurrency = True
   objDeploymentOptions.PromptForLongShortNames = True
   objDeploymentOptions.PromptForZeroSuppression = True
   objDeploymentOptions.PromptForSwapRowsAndColumns =
True
   objPPRep.Save
   Set objDeploymentOptions = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "PromptForCurrency Property" on page 370
- "PromptForDimension Property" on page 371
- "PromptForZeroSuppression Property" on page 374

PromptForSwapRowsAndColumns Property

Sets or returns whether the report consumer can swap rows and columns in a report published to the IBM Cognos portal.

Syntax

DeploymentOptions.**PromptForSwapRowsAndColumns**

Discussion

Use this property when distinct report consumer groups require the same information, but presented in different ways. If you set this property to True, users

can analyze information or follow trends by using a different report axis as reference. Report consumers can use this property so that all the data in a report will fit the current page size.

By default, the report uses the rows and columns as the author created.

Default: False

Type

Boolean

Access

Read/Write

Example

This example specifies the prompts that the report consumer sees when the report is opened in the IBM Cognos portal.

```
Sub Main()
   Dim objPPRep as Object
   Dim objDeploymentOptions as Object
   Set objPPRep = GetObject(,"CognosPowerPlay.Report")
   Set objDeploymentOptions = objPPRep.DeploymentOptions
   objDeploymentOptions.PromptForCurrency = True
   objDeploymentOptions.PromptForLongShortNames = True
   objDeploymentOptions.PromptForZeroSuppression = True
   objDeploymentOptions.PromptForSwapRowsAndColumns =
True
   objPPRep.Save
   Set objDeploymentOptions = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "PromptForCurrency Property" on page 370
- "PromptForDimension Property" on page 371
- "PromptForLongShortNames Property" on page 372
- "PromptForZeroSuppression Property"

PromptForZeroSuppression Property

Sets or returns whether the report consumer can apply or turn off zero suppression in a report published to the IBM Cognos portal.

Syntax

DeploymentOptions.**PromptForZeroSuppression**

Discussion

Use this property when a row or column in a report contains all zeros, or all zeros and a rank category, and this information is not necessary. This reduces the size and improves the readability of the report. The report consumer can customize the report in the IBM Cognos portal by selecting whether to suppress zeros on rows, columns, or on both rows and columns.

By default, the report uses the suppression mode specified by the report author when it was created.

Default: False

Type

Boolean

Access

Read/Write

Example

This example specifies the prompts that the report consumer sees when the report is opened in the IBM Cognos portal.

```
Sub Main()
   Dim objPPRep as Object
   Dim objDeploymentOptions as Object
   Set objPPRep = GetObject(,"CognosPowerPlay.Report")
   Set objDeploymentOptions = objPPRep.DeploymentOptions
   objDeploymentOptions.PromptForCurrency = True
   objDeploymentOptions.PromptForLongShortNames = True
   objDeploymentOptions.PromptForZeroSuppression = True
   objDeploymentOptions.PromptForSwapRowsAndColumns =
True
   objPPRep.Save
   Set objDeploymentOptions = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "PromptForCurrency Property" on page 370
- "PromptForDimension Property" on page 371
- "PromptForLongShortNames Property" on page 372

RefreshSubCube Property

Sets or returns whether the sub-cube is refreshed automatically.

Syntax

Application.**RefreshSubCube**

Applies To

Application Object

Discussion

Use this property to set the Refresh sub-cube preference setting to determine if sub-cube synchronization is performed during automation. A sub-cube is only a portion of a cube. This property sets the Refresh sub-cube preference setting if a Boolean value follows the property name. For example

```
ppObjApp.RefreshSubCube True
```

If a Boolean value does not follow the property, it returns the current setting for the application.

True indicates that IBM Cognos PowerPlay will automatically refresh the specified sub-cube. False indicates that the cube will be opened without refreshing the sub-cube with the cube.

Set this property before opening a sub-cube. If this property is set to True and you did not want a sub-cube refresh to take place, the report may not be as expected.

Default: True

Type

Boolean

Access

Read/Write

Example

This example returns the current preference setting for sub-cube synchronization.

```
Sub Main()
   Dim objPPApp As Object
   Dim strSetting As String
   Set objPPApp = CreateObject("CognosPowerPlay.Application")
   If objPPApp.RefreshSubCube = True Then
      strSetting = "True"
   Else
      strSetting = "False"
   End If
   MsgBox "The Refresh sub-cubes setting is " &strSetting
   Set objPPApp = Nothing
End Sub
```

Related Topics

- Chapter 4, “Methods,” on page 71
- Chapter 5, “Properties,” on page 259

SaveAllCharts Property

Sets or returns whether all Graph objects are saved in a PDF.

Syntax

SaveAsPDF.**SaveAllCharts**

Applies To

SaveAsPDF Object

Discussion

Use this method to determine whether to save the currently active Graph objects, or all Graph objects in the report. If True, all the displays visible in page layout or page width view save to the same page in the PDF. If set to False, only the selected Graph objects, or currently active graph object saves in the PDF over as many pages as required. If set to false, use the SetChartToSave method to select a chart to be saved.

When saving a report as a PDF, this property must be set to False to use the following properties:

- AxisOnAllPages
- ChartTitleOnAllPages
- IncludeLegend
- SetChartToSave
- SummariesOnAllPages

Default: False

Type

Boolean

Access

Read/Write

Example

This example opens a report, sets options for saving the report, and then saves as a PDF.

```
Sub Main()
   Dim objPDF as Object
   Dim objPPRep as Object
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.Open( "c:\Cognos\sample.ppx" )
   objPPRep.visible( TRUE )
   Set objPDF = objPPRep.PDFFile( "c:\Cognos\PDFSample"
, True )
   With objPDF
      .SaveEntireReport = False
```

```
      .SaveAllCharts = True
      .AxisOnAllPages = True
      .ChartTitleOnAllPages = False
      .IncludeLegend = True
      .SetListOfLayersToSave objPPRep.Layers
      .SetListOfRowsToSave objPPRep.Rows
   End With
   objPDF.Save
   Set objPPRep = Nothing
   Set objPDF = Nothing
End Sub
```

Related Topics

- "Save Method" on page 217
- "SaveEntireReport Property" on page 379

Saved Property

Returns whether the Report object has been saved.

Syntax

Report.**Saved**

Applies To

Report Object

Discussion

If True, the report has been saved. If False, the report has not been saved.

Type

Boolean

Access

Read

Example

This example determines whether an open report has been saved. If it hasn't, the macro saves it.

```
Sub Main()
   Dim objRep As Object
   Set objRep = GetObject( ,"CognosPowerPlay.Report")
   MsgBox "The name of the current report is " &objRep.Name
   If objRep.Saved = False Then
      objRep.Save
      MsgBox "Changes to the report have been saved."
   Else
```

```
      MsgBox "No changes have been made to the report."
   End If
   objRep.Application.Activate
   Set objRep = Nothing
End Sub
```

Related Topics

- "Report Object" on page 37

SaveEntireReport Property

Sets or returns whether to save the entire report as a PDF.

Syntax

SaveAsPDF.**SaveEntireReport**

Applies To

SaveAsPDF Object

Discussion

Use this property to save the entire report no matter which other properties are set for saving reports as a PDF. If True, the entire report saves as a PDF file; this property overrides all other property settings associated with saving a report in portable document format. If False, you can set other properties to determine what you want to appear in the saved PDF.

Default: True

Type

Boolean

Access

Read/Write

Example

This example opens a report, sets options for saving the report, and then saves the report as a PDF.

```
Sub Main()
   Dim objPDF as Object
   Dim objPPRep as Object
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.Open( "c:\Cognos\sample.ppx" )
   objPPRep.visible( TRUE )
   Set objPDF = objPPRep.PDFFile( "c:\Cognos\PDFSample"
, True )
   With objPDF
      .SaveEntireReport = False
      .SaveAllCharts = True
```

```
        .AxisOnAllPages = True
        .ChartTitleOnAllPages = False
        .IncludeLegend = True
        .SetListOfLayersToSave objPPRep.Layers
        .SetListOfRowsToSave objPPRep.Rows
    End With
    objPDF.Save
    Set objPPRep = Nothing
    Set objPDF = Nothing
End Sub
```

Related Topics

- "AxisOnAllPages Property" on page 272
- "ChartTitleOnAllPages Property" on page 285
- "IncludeLegend Property" on page 316
- "SaveAllCharts Property" on page 377
- "SetListOfLayersToSave Method" on page 231
- "SetListOfRowsToSave Method" on page 233
- "SummariesOnAllPages Property" on page 401

SearchDescription Property

Sets or returns whether the FindQuery object searches the category descriptions in a cube.

Syntax

FindQuery.**SearchDescription**

Applies To

FindQuery Object

Discussion

Use a FindQuery object to locate categories within a cube that match the search string specified by the SearchDescription property.

To search the long or short names, set the SearchDescription property to False. To search the description, set this property to True. The FindQuery object returns a list of category labels that match the search text.

The order of the components in the subset definition is important. First, set the Dimension property, followed by the SearchShortName, or SearchDescription, SearchText, and Pattern properties, and then the Execute and AddToReport methods. Set the Name property anywhere within the subset definition.

Default: False

Type

Boolean

Access

Read/Write

Example

This example creates a FindQuery (type 1) subset definition that searches for all products that begin with the name Star.

```
Sub Main()
   Dim strCubePath As String
   Dim objPPRep As Object
   Dim objFind As Object
   strCubePath = "C:\Cubes and Reports\Great Outdoors.mdc"
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.New strCubePath, 1
   objPPRep.ExplorerMode = False
   objPPRep.Visible = True
   Set objFind = objPPRep.ReportQueries.Add(1)
   With objFind
      .Name = "Find Star"
      .Dimension = "Products"
      .SearchShortName = False
      .SearchDescription = True
      .SearchText = "Star"
      .Pattern = 2
   End With
   Set objFind = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "SearchText Property" on page 383

SearchShortName Property

Sets or returns whether the FindQuery object searches short or long category names.

Syntax

FindQuery.**SearchShortName**

Applies To

FindQuery Object

Discussion

Use a FindQuery object to locate categories within a cube that match the search string specified by the SearchText property. Normally, the operation searches the category label text used in the short names for cube data. Set this property to False to search for long names.

The result is dependent on how the user views data. If the user specifies long names, searching returns a list of matching category labels with the long name format; similarly, using short names returns the short name format.

To search for a description, set this property to false and the SearchDescription property to True.

The order of the components in the subset definition is important. First, set the Dimension property, followed by the SearchShortName, SearchDescription, SearchText, and Pattern properties, and then the Execute and AddToReport methods. Set the Name property anywhere within the subset definition.

Default: False

Type

Boolean

Access

Read/Write

Example

This example creates a FindQuery (type 1) subset definition that searches for all products that begin with the name Star.

```
Sub Main()
   Dim strCubePath As String
   Dim objPPRep As Object
   Dim objFind As Object
   strCubePath = "C:\Cubes and Reports\Great Outdoors.mdc"
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.New strCubePath, 1
   objPPRep.ExplorerMode = False
   objPPRep.Visible = True
   Set objFind = objPPRep.ReportQueries.Add(1)
   With objFind
      .Name = "Find Star"
      .Dimension = "Products"
      .SearchShortName = False
      .SearchText = "Star"
      .Pattern = 2
   End With
   Set objFind = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "SearchText Property" on page 383

SearchText Property

Sets or returns the search string used in the subset definition of a FindQuery query.

Syntax

FindQuery.**SearchText**

Applies To

FindQuery Object

Discussion

Use this property to specify the name of the category that the query searches for. Set this property to a text before executing a Find query (a search of a cube). Unless specified by the Pattern property, the search is not case sensitive. The search string must contain characters to create a subset. The search only applies to category labels, not to the data. If there are no categories that match the specified search criteria at the time that the subset is created, then an empty subset is returned.

The order of the components in the subset definition is important. First, set the Dimension property, followed by the SearchShortName, SearchDescription, SearchText and Pattern properties, and then the Execute and AddToReport methods. Set the Name property anywhere within the subset definition.

Type

String

Access

Read/Write

Example

This example creates a FindQuery (type 1) subset definition that searches for all products that begin with the name Star.

```
Sub Main()
   Dim strCubePath As String
   Dim objPPRep As Object
   Dim objFind As Object
   strCubePath = "C:\Cubes and Reports\Great Outdoors.mdc"
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.New strCubePath, 1
   objPPRep.ExplorerMode = False
   objPPRep.Visible = True
   Set objFind = objPPRep.ReportQueries.Add(1)
   With objFind
      .Name = "Find Star"
      .Dimension = "Products"
      .SearchShortName = False
```

```
      .SearchText = "Star"
      .Pattern = 2
   End With
   Set objFind = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "SearchShortName Property" on page 381

ShareDimensionLine Property

Sets or returns whether open reports share a dimension line.

Syntax

object.**ShareDimensionLine**

Applies To

Application Object

Report Object

Discussion

If set to True, changes to the dimension line in the first report are automatically reflected in any reports that

- are open or subsequently opened,
- share the same multidimensional cube, and
- have the Shared Dimensions command turned on.

If set to False, changes to the dimension line of this report are not reflected in other reports. Changes to other reports that have this property set to True are also not reflected in this report.

If this property is set to True for the Report object, the existing setting of this property in the Application object is ignored. Conversely, when this property is set to True for the Application object, only reports created after this property is set are affected.

Default: True

Type

Boolean

Access

Read/Write

Example

This example sets the ShareDimensionLine property to True for all reports in the IBM Cognos PowerPlay application.

```
Sub Main()
   Dim objPPApp As Object
   Set objPPApp = CreateObject ("CognosPowerPlay.Application")
   objPPApp.Visible = True
   objPPApp.Activate
   objPPApp.ShareDimensionLine = True
   MsgBox objPPApp.ShareDimensionLine
   Set objPPApp = Nothing
End Sub
```

Related Topics

- "Application Object" on page 11
- "Report Object" on page 37

ShareOf Property

Sets or returns whether to show the values in selected categories as a percentage of their higher-level category.

Syntax

CategoryList.**ShareOf**

Applies To

CategoryList Object

Discussion

If True, this property shows the values in the selected categories as a percentage of their higher-level category.

Default: False

Type

Boolean

Access

Read/Write

Example

This example sets the categories in the report as a share of a higher-level category.

```
Sub Main
   Dim objPPRep as Object
   Dim objCatList as Object
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
```

```
   objPPRep.New "C:\Cubes and Reports\Great Outdoors.mdc"
   objPPRep.ExplorerMode = False
   objPPRep.Visible = True
   Set objCatList = objPPRep.CategoryList
   objCatList.Add 1, "Products" ,"Outdoor Products"
   objCatList.Each = False
   objCatList.ShareOf = True
   objPPRep.Columns.Add objCatList
   objCatList.Remove
   objCatList.Add 0, "Measures", "Revenue"
   objCatList.Each = True
   objCatList.ShareOf = False
   objPPRep.Rows.Add objCatList
   Set objCatList = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "CategoryList Object" on page 13

ShowSummaryBreakdown Property (Explorer)

Sets or returns whether to show the breakdown of summary rows and columns in a crosstab.

Syntax

object.**ShowSummaryBreakdown**

Applies To

Graph Object

Graphs

Discussion

Use this property to show subtotals for each outer level summary. When this property is set to True, the report inserts automatic subtotals derived from details in the crosstab rows and columns. In a crosstab with a long list of data, you can use the summary data to outline the information in the report. Set this property to true for nested crosstabs so that the nested level summary is visible.

The ShowSummaryRow or the ShowSummaryColumn and the ShowSummaryBreakdown properties must both be set to True to see the details for each outer level summary row or column. If the ShowSummaryRow or ShowSummaryColumn property is set to True, but the ShowSummaryBreakdown is set to False, only a single row or column appears for each summary. If the ShowSummaryRow or ShowSummaryColumn property is set to False, but the ShowSummaryBreakdown is set to True, no summary rows or columns appear.

Default: True

Type

Boolean

Access

Read/Write (Graph)

Write (Graphs)

Example

This example sets the ShowSummaryBreakdown property to false so that the crosstab hides the subtotal summary for the nested categories.

```
Sub Main()
   Dim objCubeCategories As Object
   Dim objPPRep As Object
   Dim objGraph As Object
   Const level_0 = 0
   Const level_1 = 1
   Const add_to_current = 0
   Const add_to_all = 1
   Const as_parent = 0
   Const as_child = 1
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.New "D:\Cubes and Reports\Great Outdoors.mdc",
-1
   objPPRep.ExplorerMode = True
   objPPRep.Visible = True
   Set objCubeCategories = objPPRep.CategoryList()
   objCubeCategories.Add level_1, "Locations"
   objPPRep.Rows.AddLevel objCubeCategories, level_0,
_
         add_to_all, as_child
   Set objGraph = objPPRep.Graphs.Item(1)
   objGraph.SetType 0
   objPPRep.Graphs.ShowSummaryRow = True
   objPPRep.Graphs(1).ShowSummaryBreakdown = False
   Set objCubeCategories = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "ShowSummaryColumn Property (Explorer)"
- "ShowSummaryRow Property (Explorer)" on page 388

ShowSummaryColumn Property (Explorer)

Sets or returns whether to show the summary column.

Syntax

object.**ShowSummaryColumn**

Applies To

Graph Object

Graphs

Discussion

If True, this property shows the summary column. If False, the summary column is hidden.

This property is not available for pie graphs and only applies to Explorer reports.

Default: True

Type

Boolean

Access

Read/Write (Graph)

Write (Graphs)

Example

This example adds two graphs to the currently open report and shows the summary column for only the third graph of the report.

```
Sub Main()
   Dim objPPRep As Object
   Set objPPRep = GetObject (, "CognosPowerPlay.Report")
   objPPRep.Graphs.Add 1
   objPPRep.Graphs.Add 2
   objPPRep.Graphs.ShowSummaryColumn = False
   objPPRep.Graphs(3).ShowSummaryColumn = True
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "Graph Object" on page 25
- "Graphs" on page 56

ShowSummaryRow Property (Explorer)

Sets or returns whether to show the summary row.

Syntax

object.**ShowSummaryRow**

Applies To

Graph Object

Graphs

Discussion

If True, this property shows the summary row. If False, the summary row is hidden.

This property is not available for pie, single bar, simple bar, and correlation graphs, and only applies to Explorer reports.

Default: True

Type

Boolean

Access

Read/Write (Graph)

Write (Graphs)

Example

This example adds two graphs to the currently open report and shows the summary column for only the third graph of the report.

```
Sub Main()
   Dim objPPRep As Object
   Set objPPRep = GetObject (, "CognosPowerPlay.Report")
   objPPRep.Graphs.Add 1
   objPPRep.Graphs.Add 2
   objPPRep.Graphs.ShowSummaryRow = False
   objPPRep.Graphs(3).ShowSummaryRow = True
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "Graph Object" on page 25
- "Graphs" on page 56

ShowTies Property

Sets or returns whether to show label ties.

Syntax

object.**ShowTies**

Applies To

Graph Object

Graphs

Discussion

For some graphs, labels along an axis can be connected or tied together. Set this property to True to display these ties.

ShowTies does not apply to pie graphs, scatter graphs, crosstabs, or a three-dimensional-bar graph.

Default: False

Type

Boolean

Access

Read/Write (Graph)

Write (Graphs)

Example

This example sets the ShowTies property for each applicable graph object in the active report.

```
Sub Main()
   Dim objPPRep As Object
   Dim objGraph As Object
   Dim intx As Integer
   Set objPPRep = GetObject(, "CognosPowerPlay.Report")
   intx=0
   Do
      intx=intx+1
      Set objGraph = objPPRep.Graphs.Item(intx)
      Select Case objGraph.Type
         Case 0,1,2,9
            MsgBox "ShowTies property does not apply
to " & _
                   "these graph types. "
         Case Else
            If objGraph.ShowTies = 0 Then
               objGraph.ShowTies = True
            Else
               objGraph.ShowTies = False
            End If
      End Select
   Loop Until intx = objPPRep.Graphs.Count
```

```
   Set objGraph = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "StatsLineOn Property" on page 395

ShowValuesAs Property (Explorer)

Sets or returns how to show values in a report.

Syntax

Report.**ShowValuesAs**

Applies To

Report Object

Discussion

These are the possible values that ShowAs can be:

1=Shows values as number

2=Shows values as a percentage of the row total

3=Shows values as a percentage of the column total

4=Shows values as a percentage of the layer total

5=Shows values as a percentage of the grand total

This property is only available if the Report object is in Explorer Mode. (ExplorerMode property set to True.)

Type

Long

Access

Read/Write

Example

This example returns an open report, and shows values as a percentage of the grand total.

```
Sub Main()
   Dim objPPRep as Object
   Dim objRepPrt as Object
   Set objPPRep = GetObject (, "CognosPowerPlay.Report")
   Set objRepPrt = objPPRep.Print
   objPPRep.ExplorerMode = True
```

```
   objPPRep.ShowValuesAs(5)
   objRepPrt.PrintOut
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "SaveEntireReport Property" on page 379
- "Report Object" on page 37

StatsLineCaption Property

Sets or returns the caption for a given statistical line on a graph.

Syntax

object.**StatsLineCaption**(LineNumber)

Applies To

Graph Object

Graphs

Discussion

Many graphs can be enhanced by the addition of statistical lines, such as a mean or average line superimposed on a bar graph of sales figures. Up to three lines can be added, as determined by the StatsLineOn property. Each statistical line is explained in an automatically generated legend. StatsLineCaption sets the line's label in the legend.

If not set, the caption defaults to the line type, such as "Maximum."

The applicable graph types are simple bar, clustered bar, correlation, simple line, multiple line, and scatter.

Parameters	Description
LineNumber	Required. Specifies which statistical line to reference: 1 = the minimum line (default caption: Minimum) 2 = the maximum line (default caption: Maximum) 3 = the mean or average line (default caption: Mean) 4 = the standard deviation line (default caption: Standard Deviation) 5 = the regression line (default caption: Regression) 6 = the first user-created line (default caption: Custom 1) 7 = the second user-created line (default caption: Custom 2) Type: Short

Type

String

Access

Read/Write

Example

This example sets the first Graph object to the clustered bar type and displays a custom statistical line. The custom line is referred to as the Limit, and it is set to 100,000.

```
Sub Main()
   Dim objPPRep As Object
   Dim objPPGraph As Object
   Set objPPRep = GetObject(, "CognosPowerPlay.Report")
   Set objPPGraph = objPPRep.Graphs.Item(1)
   With objPPGraph
      .SetType 4,1,1
      .StatsLineOn 6,True        'custom line
      .StatsLineColor 6,2        'green
      .StatsLineStyle 6,1        'dashed style
      .StatsLineUserValue 6,100000
      .StatsLineCaption 6,"Limit"
   End With
   Set objPPGraph = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "StatsLineColor Property"
- "StatsLineOn Property" on page 395
- "StatsLineStyle Property" on page 396
- "StatsLineUserValue Property" on page 398

StatsLineColor Property

Sets or returns the color for a given statistical line on a graph.

Syntax

object.**StatsLineColor(LineNumber)**

Applies To

Graph Object

Graphs

Discussion

Many graphs can be enhanced by the addition of statistical lines. Use StatsLineColor to change the color of a statistical line. If the property is used without the Color parameter, the current color is returned.

Up to three lines can be added, as determined by the StatsLineOn property.

The applicable graph types are simple bar, clustered bar, correlation, simple line, multiple line, and scatter.

Default: 0

Parameters	Description
LineNumber	Required. Specifies which statistical line to reference: 1 = the minimum line 2 = the maximum line 3 = the mean or average line 4 = the standard deviation line 5 = the regression line 6 = the first user-created line 7 = the second user-created line Type: Integer

Type

Integer

Access

Read/Write

Example

This example sets the first Graph object to the clustered bar type and displays a custom statistical line. The custom line is referred to as the Limit, and it is set to 100,000.

```
Sub Main()
   Dim objPPRep As Object
   Dim objPPGraph As Object
   Set objPPRep = GetObject(, "CognosPowerPlay.Report")
   Set objPPGraph = objPPRep.Graphs.Item(1)
   With objPPGraph
      .SetType 4,1,1
      .StatsLineOn 6,True        'custom line
      .StatsLineColor 6,2        'green
      .StatsLineStyle 6,1        'dashed style
      .StatsLineUserValue 6,100000
      .StatsLineCaption 6,"Limit"
   End With
   Set objPPGraph = Nothing
```

```
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "StatsLineCaption Property" on page 392
- "StatsLineOn Property"
- "StatsLineStyle Property" on page 396
- "StatsLineUserValue Property" on page 398

StatsLineOn Property

Sets or returns a statistical line on a graph.

Syntax

object.**StatsLineOn**(LineNumber)

Applies To

Graph Object

Graphs

Discussion

Many graphs can be enhanced by the addition of statistical lines. Use StatsLineOn to pick the types of statistical lines to turn on or off. If the Switch property is used without the parameters, the current Boolean value is returned.

Up to three lines can be added, as determined by the StatsLineOn property.

The applicable graph types are simple bar, clustered bar, correlation, simple line, multiple line, and scatter.

Parameters	Description
LineNumber	Required. Specifies which statistical line to reference: 1 = minimum: a line that spans across the minimum value 2 = maximum: a line that spans across the maximum value 3 = mean: a line that spans across the average value 4 = standard deviation: two lines spanning pre-defined deviants from the mean 5 = regression: a logarithmic regression line 6 = the first user-created line 7 = the second user-created line Type: Short

Return Type

Long

Access

Read/Write

Example

This example sets the first Graph object to the clustered bar type and displays a custom statistical line. The custom line is referred to as the Limit, and it is set to 100,000.

```
Sub Main()
   Dim objPPRep As Object
   Dim objPPGraph As Object
   Set objPPRep = GetObject(, "CognosPowerPlay.Report")
   Set objPPGraph = objPPRep.Graphs.Item(1)
   With objPPGraph
      .SetType 4,1,1
      .StatsLineOn 6,True        'custom line
      .StatsLineColor 6,2        'green
      .StatsLineStyle 6,1        'dashed style
      .StatsLineUserValue 6,100000
      .StatsLineCaption 6,"Limit"
   End With
   Set objPPGraph = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "StatsLineCaption Property" on page 392
- "StatsLineColor Property" on page 393
- "StatsLineStyle Property"
- "StatsLineUserValue Property" on page 398

StatsLineStyle Property

Sets or returns the line style of a given statistical line on a graph.

Syntax

object.**StatsLineStyle**(LineNumber)

Applies To

Graph Object

Graphs

Discussion

Many graphs can be enhanced by the addition of statistical lines. Use StatsLineStyle to change the style of a statistical line. If the property is used without the Style parameter, the current style is returned.

Up to three lines can be added, as determined by the StatsLineOn property.

The applicable graph types are simple bar, clustered bar, correlation, simple line, multiple line, and scatter.

Default: 0

Parameters	Description
LineNumber	Required. Specifies statistical line to reference: 1 = minimum line 2 = maximum line 3 = mean or average line 4 = standard deviation line 5 = logarithmic regression line 6 = first line created by the user 7 = second user-created line 8 = linear regression line Type: Short

Type

Integer

Access

Read/Write

Example

This example sets the first Graph object to the clustered bar type and displays a custom statistical line. The custom line is referred to as the Limit, and it is set to 100,000.

```
Sub Main()
   Dim objPPRep As Object
   Dim objPPGraph As Object
   Set objPPRep = GetObject(, "CognosPowerPlay.Report")
   Set objPPGraph = objPPRep.Graphs.Item(1)
   With objPPGraph
      .SetType 4,1,1
      .StatsLineOn 6,True        'custom line
      .StatsLineColor 6,2        'green
      .StatsLineStyle 6,1        'dashed style
      .StatsLineUserValue 6,100000
      .StatsLineCaption 6,"Limit"
```

```
    End With
    Set objPPGraph = Nothing
    Set objPPRep = Nothing
End Sub
```

Related Topics

- "StatsLineCaption Property" on page 392
- "StatsLineColor Property" on page 393
- "StatsLineOn Property" on page 395
- "StatsLineUserValue Property"

StatsLineUserValue Property

Sets a custom value for a statistical line on a graph.

Syntax

object.**StatsLineUserValue**(LineNumber)

Applies To

Graph Object

Graphs

Discussion

You can enhance graphs by the addition of statistical lines. Use StatsLineUserValue to place a statistical line at a value of your choosing, such as a quota line superimposed on a bar graph of sales figures. Up to three lines can be added, as determined by the StatsLineOn property. If the property is used without the value parameter, the current value is returned.

The applicable graph types are simple bar, clustered bar, correlation, simple line, multiple line, and scatter.

Default: 0

Parameters	Description
LineNumber	Required. Specifies which user-created statistical line to reference: 6 = the first user-created line 7 = the second user-created line Type: Short

Type

Double

Access

Write

Example

This example sets the first Graph object to the clustered bar type and displays a custom statistical line. The custom line is referred to as the Limit, and it is set to 100,000.

```
Sub Main()
   Dim objPPRep As Object
   Dim objPPGraph As Object
   Set objPPRep = GetObject(, "CognosPowerPlay.Report")
   Set objPPGraph = objPPRep.Graphs.Item(1)
   With objPPGraph
      .SetType 4,1,1
      .StatsLineOn 6,True        'custom line
      .StatsLineColor 6,2        'green
      .StatsLineStyle 6,1        'dashed style
      .StatsLineUserValue 6,100000
      .StatsLineCaption 6,"Limit"
   End With
   Set objPPGraph = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "StatsLineCaption Property" on page 392
- "StatsLineColor Property" on page 393
- "StatsLineOn Property" on page 395
- "StatsLineStyle Property" on page 396

Style Property

Sets or returns the style used for a category, an exception range or set of categories.

Syntax

object.**Style**

Applies To

Column Object

Columns

Layer Object

Layers

Row Object

Rows

Discussion

Only existing styles can be applied. If the style does not exist, an error occurs.

Type

String

Access

Read/Write (Row, Column, Layer)

Write (Rows, Columns, Layers)

Example

This example applies a specific style to all the rows in the report.

```
Sub Main()
   Dim objPPRep as Object
   Set objPPRep = CreateObject ("CognosPowerPlay.Report")
   objPPRep.Open "C:\Cubes and Reports\Sample.ppx.ppx"
   objPPRep.Rows.Style = "Good News"
   objPPRep.Save
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "Column Object" on page 15
- "Layer Object" on page 28
- "Row Object" on page 41

Sum Property

Sets or returns whether to calculate the sum of selected categories.

Syntax

CategoryList.**Sum**

Applies To

CategoryList Object

Discussion

If True, a new category is created to show the sum of the selected categories. If False, the sum is not calculated.

Default: False

Type

Boolean

Access

Read/Write

Example

This example adds categories to the columns and rows and then calculates the sum of the categories in a report.

```
Sub Main()
   Dim objPPRep as Object
   Dim objCatList as Object
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.New "C:\Cubes and Reports\Great " & _
      "Outdoors.mdc", False
   Set objCatList = objPPRep.CategoryList
   objPPRep.Visible = True
   objCatList.Add 1, "Products", "Outdoor Products"
   objPPRep.Columns.Add objCatList
   objCatList.Add 1, "Locations", "Far East"
   objPPRep.Rows.Add objCatList
   MsgBox "The sum of the categories is " &objCatList.Sum
   Set objCatList = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "CategoryList Object" on page 13

SummariesOnAllPages Property

Sets or returns whether existing summaries appear on every page of a printed report.

Syntax

Print.**SummariesOnAllPages**

Applies To

Print Object

Discussion

Use this property to determine how much detail appears in a printed report. If True, the summary categories (if any) appear on every printed page of the report. If False, the summary categories appear only once on the last page of the printed report.

Use this property only if the PrintAllCharts property is set to False.

Default: False

Type

Boolean

Access

Read/Write

Example

This example opens a report and prints one copy of all the data for the second graphical display only. This example includes the display title, summary category, and axis on all pages; turns on the collating option; and excludes the legend in the printed report.

```
Sub Main()
   Dim objPPRep as Object
   Dim objRepPrt as Object
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.Open "C:\Cubes and Reports\Sample2.ppx"
   Set objRepPrt = objPPRep.Print
   objRepPrt.PrintAllCharts = False
   objRepPrt.SetListOfRowsToPrint objPPrep.Rows
   objRepPrt.SetListOfLayersToPrint objPPRep.Layers
   objRepPrt.SetChartToPrint objPPRep.Graphs.Item(2)
   objRepPrt.IncludeLegend = True
   objRepPrt.ChartTitleOnAllPages = True
   objRepPrt.SummariesOnAllPages = True
   objRepPrt.AxisOnAllPages = True
   objRepPrt.Collate = True
   objRepPrt.Copies = 1
   objRepPrt.PrintOut
   Set objRepPrt = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "Print Method" on page 198

SummaryColumnOnAllPages Property

Sets or returns whether to show the summary column category on every page of a PDF.

Syntax

object.**SummaryColumnOnAllPages**

Applies To

SaveAsPDF Object

Discussion

Use this property to determine how much detail appears in a PDF. When you save a report as a PDF, the Print object properties are replaced with the PDF options.

If True, the summary column category appears on every page of the report. If False, the summary column category appears only on the last page of the report.

When saving a report as a PDF file, use this property only if the SaveAllCharts property is set to False.

This property is valid for Explorer reports only.

Default: False

Type

Boolean

Access

Read/Write

Example

This example sets the summary column to show on every page of a report published to Upfront.

```
Sub Main()
   Dim objPPRep as Object
   Dim objDeploymentOptions as Object
   Set objPPRep = GetObject(,"CognosPowerPlay.Report")
   Set objDeploymentOptions = objPPRep.DeploymentOptions
   objDeploymentOptions.PromptForCurrency = True
   objDeploymentOptions.PromptForLongShortNames = True
   objDeploymentOptions.PromptForZeroSuppression = True
   objDeploymentOptions.PromptForSwapRowsAndColumns =
True
   objDeploymentOptions.PromptForDimension(1) = True
   objDeploymentOptions.PromptForDimension(2)= True
   objDeploymentOptions.PromptForDimension("Years")=
True
   objDeploymentOptions.SummaryColumnOnAllPages = True
   objPPRep.Save
   Set objDeploymentOptions = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "Print Method" on page 198
- "PrintAllCharts Property" on page 364
- "SaveAllCharts Property" on page 377
- "SaveEntireReport Property" on page 379

SummaryRowOnAllPages Property

Sets or returns whether to show the summary row category on every page of a PDF.

Syntax

object.**SummaryRowOnAllPages**

Applies To

SaveAsPDF Object

Discussion

Use this property to determine how much detail appears in a PDF. When you save a report as a PDF, the Print object properties are replaced with the PDF options.

If True, the summary row category appears on every page of the report. If False, the summary row category appears once on the last page of the report.

When saving a report as a PDF file, use this method only if the SaveAllCharts property is set to False.

This property is valid for Explorer reports only.

Default: False

Type

Boolean

Access

Read/Write

Example

This example sets the summary row to show on every page of a report published to Upfront.

```
Sub Main()
   Dim objPPRep as Object
   Dim objDeploymentOptions as Object
   Set objPPRep = GetObject(,"CognosPowerPlay.Report")
   Set objDeploymentOptions = objPPRep.DeploymentOptions
   objDeploymentOptions.PromptForCurrency = True
   objDeploymentOptions.PromptForLongShortNames = True
   objDeploymentOptions.PromptForZeroSuppression = True
   objDeploymentOptions.PromptForSwapRowsAndColumns =
True
   objDeploymentOptions.PromptForDimension(1) = True
   objDeploymentOptions.PromptForDimension(2)= True
   objDeploymentOptions.PromptForDimension("Years")=
True
```

```
    objDeploymentOptions.SummaryRowOnAllPages = True
    objPPRep.Save
    Set objDeploymentOptions = Nothing
    Set objPPRep = Nothing
End Sub
```

Related Topics

- "Print Method" on page 198
- "PrintAllCharts Property" on page 364
- "SaveAllCharts Property" on page 377
- "SaveEntireReport Property" on page 379

Suppress8020 Property (Explorer)

Sets or returns the 80/20 suppress mode for report dimensions.

Syntax

Report.**Suppress8020**

Applies To

Report Object

Discussion

Use this property to refine the focus of a report by identifying categories that are significant contributors to a business measure and by applying rollup to the unimportant ones. You can set this property so that it groups rows and columns that are not meaningful into a single category called Other. 80/20 suppression means that 20 percent of the sample contributes 80 percent of the information. Therefore your view of report data depends on the category population and distribution.

You can suppress less critical items in rows or columns or in both rows and columns. Whatever dimension choice you make, the 80/20 suppression rule applies to all layers in a report. There are four settings for this property:

- 0 (off)
- 1 (on rows only)
- 2 (on columns only)
- 3 (on rows and columns)

It is possible that all categories are significant and there will not be an Other category.

You cannot drill, rank, insert, or delete the Other category. You cannot apply 80/20 suppression when a report dimension includes multiple measures.

You can apply Exception highlighting and select the Other category.

All Other categories along rows, columns, or layers will share the same characteristics, including formatting, name, and hidden state. You cannot apply separate formatting for each Other category in a nested, filtered, or drilled report.

Default: 0

Type

Integer

Access

Read/Write

Example

This example opens a IBM Cognos PowerPlay report and applies the Suppress8020 property to rows and columns.

```
Sub Main()
   Dim objPPRep As Object
   Set objPPRep = CreateObject ("CognosPowerPlay.Report")
   objPPRep.Open ("C:\Cubes and Reports\Sample1.ppx")
   objPPRep.ExplorerMode = True
   objPPRep.Visible = True
   objPPRep.ShowValuesAs = 2
   objPPRep.ShareDimensionLine = 1
   objPPRep.Suppress8020 = 3
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "SuppressZeros Property"

SuppressZeros Property

Sets or returns the suppress mode for the Report object.

Syntax

Report.**SuppressZeros**

Applies To

Report Object

Discussion

This property is used to set the suppression mode and also perform suppress or unsuppress operations based on the mode.

Return values for this property are:

- 1 (not suppressed)
- 2 (suppress rows and columns)
- 3 (suppress rows only)
- 4 (suppress columns only)

Type

Long

Access

Read/Write

Example

This example opens a IBM Cognos PowerPlay report and sets the SuppressZeros property to True for rows only.

```
Sub Main()
   Dim objPPRep As Object
   Set objPPRep = CreateObject ("CognosPowerPlay.Report")
   objPPRep.Open ("C:\Cubes and Reports\Sample1.ppx")
   objPPRep.Visible (1)
   objPPRep.ShowValuesAs (2)
   objPPRep.ShareDimensionLine (1)
   objPPRep.SuppressZeros (3)
   objPPrep.Save
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "Report Object" on page 37

Threshold Property

Sets or returns the maximum printing page limit for the Print object.

Syntax

Print.**Threshold**

Applies To

Print Object

Discussion

Use this property to limit the number of pages to print in an automation script. Printing from the user interface is not affected by this property. The value for the number of copies does not affect the maximum print range limit.

Default: 999

Type

Integer

Access

Read/Write

Example

This example sets the printing threshold to 10 pages for the Print object.

```
Sub Main()
   Dim objPPRep as Object
   Dim objRepPrt as Object
   Set objPPRep = CreateObject("CognosPowerPlay.Report")
   objPPRep.Open "C:\Cubes and Reports\Sample1.ppx"
   Set objRepPrt = objPPRep.Print
   objRepPrt.Threshold = 10
   objRepPrt.PrintAllCharts = False
   objRepPrt.SetListOfRowsToPrint objPPrep.Rows
   objRepPrt.SetListOfLayersToPrint objPPRep.Layers
   objRepPrt.SetChartToPrint objPPRep.Graphs.Item(1)
   objRepPrt.IncludeLegend = False
   objRepPrt.ChartTitleOnAllPages = True
   objRepPrt.SummariesOnAllPages = True
   objRepPrt.AxisOnAllPages = True
   objRepPrt.Collate = True
   objRepPrt.PrintOut
   Set objRepPrt = Nothing
   Set objPPrep = Nothing
End Sub
```

Related Topics

- "Print Object" on page 33

TitleText Property

Sets or returns the text in the title of a report.

Syntax

Report.**TitleText(Format)**

Applies To

Report Object

Discussion

Use this property to set or return the complete text in the title of a report. Because you can add as many lines to the title as required, the title may contain information that is not visible when you view or print the report. You can use automation to view information that the report author included in the title but did not want the users to see. You can set or return the title of a report in text or HTML format.

If you set the footer as text, then the title is left justified and uses the default font size, font color, and font type. To specify multiple lines in a title, use the ASCII value of a new line character, chr$(10), between the lines of the title.

If you set the title of a report using HTML, you can specify complex formats. You can specify the font size, font color, font type and alignment of the title text. For example, the following code aligns the title text to the right and specifies the font size and color.

```
strTitle = "<P ALIGN=""Right""><FONT SIZE=6 COLOR=""#0000FF"">"
+ "MyNewTitle center<BR>On a New Line"
```

When you assign HTML tags to a string, use two sets of quotation marks to distinguish characters within the HTML tag from end of string quotation marks. For example, to assign <P ALIGN="Right"> to a string, use the following syntax:

```
objPPRep.TitleText(2) = "<P ALIGN=""Right"">"
```

If you use an external editor to specify HTML, you may have to modify the HTML to get the format you require for the report title.

A report title can include IBM Cognos PowerPlay variables or variables expanded to the values that they represent. To specify a variable use the following syntax:

```
[PPVAR]Variable[PPVAR]
```

For example,

```
objPPRep.TitleText(1) = "[PPVAR]Page #[PPVAR]"
```

You can return the value of the variable or the expanded variable. To return the variable expanded to the value it represents, use the parameter value 11 for text or 12 for HTML.

For example,

```
objPPRep.TitleText(1) = "Report Printed [PPVAR]Date(ddd,
dd MMM, yyy)[PPVAR]"
MsgBox objPPRep.TitleText(1)
```

returns

Report Printed [PPVAR]Date(ddd, dd MMM, yyy)[PPVAR]

```
MsgBox objPPRep.TitleText(11)
```

returns

Report Printed Thursday, 13 January, 2000

Parameter	Description
Format	Optional. Specifies the format of the text in the title of a report. If not specified, the text and variable are returned. Valid set and return values are 1 = Text 2 = HTML Valid return values are 11 = Text with variables expanded 12 = HTML with variables expanded **Default:** 1 Type: Integer

Type

String

Access

Read/Write

Example

This example specifies three lines of text for the title of the open report, and returns the full title and the date.

```
Sub Main()
   Dim objRep As Object
   Dim strTitleTextLine1 as String
   Dim strTitleTextLine2 as String
   Dim strTitleTextLine3 as String
   Dim strNewLine as String
   Set objRep = GetObject( ,"CognosPowerPlay.Report")
   strTitleTextLine1 = "Annual Sales Report"
   strTitleTextLine2 = "Northwest Region"
   strTitleTextLine3 = "[PPVAR]Date(dddd, MMMM dd, yyyy)[PPVAR]"
   strNewLine = chr$(10)
   objRep.TitleText(1) = strTitleTextLine1 + chr$(10)
_
       +  strTitleTextLine2 + chr$(10) + strTitleTextLine3
   MsgBox "The report title text is:  " _
      &objRep.TitleText(11)
   objRep.Save
   Set objRep = Nothing
End Sub
```

Related Topics

- "FooterText Property" on page 309
- "HeaderText Property" on page 313
- "Report Object" on page 37
- "Reports" on page 65

TopLevelCategory Property (Explorer)

Returns the name of the dimension for the collection.

Syntax

collection.**TopLevelCategory**

Applies To

Columns

Layers

Rows

Discussion

Use this property to determine the name of the dimension that the selected categories belong to in a report. By returning the dimension name, you can identify data on a specific portion of a business, such as products, dates, or markets.

This property applies to Explorer reports only.

Type

String

Access

Read

Example

This example issues a message box telling the user which dimensions the rows and columns of the open Explorer report are from.

```
Sub Main()
   Dim objPPRep As Object
   Dim objRows As Object
   Dim objCols As Object
   Set objPPRep = GetObject (,"CognosPowerPlay.Report")
   Set objRows = objPPRep.Rows
   Set objCols = objPPRep.Columns
   MsgBox "Rows are from dimension: " _
      &objRows.TopLevelCategory
   MsgBox "Columns are from dimension: " _
```

```
      &objCols.TopLevelCategory
   Set objRows = Nothing
   Set objCols = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "Columns" on page 52
- "Layers" on page 59
- "Rows" on page 66

TopLevelParentCategory Property

Returns the name of the dimension for the object.

Syntax

object.**TopLevelParentCategory**

Applies To

Column Object

Layer Object

Row Object

Discussion

This property is used to get the name of the dimension that a category belongs to.

This property will return an empty string if the category is the top level or if it is a calculation.

Type

String

Access

Read

Example

This example issues a message box telling the user which dimensions the second row and second column of the open report are from.

```
Sub Main()
   Dim objPPRep As Object
   Dim objRow As Object
   Dim objCol As Object
   Set objPPRep = GetObject (,"CognosPowerPlay.Report")
   Set objRow = objPPRep.Rows(2)
   Set objCol = objPPRep.Columns(2)
```

```
   MsgBox objRow.Name & " is from dimension: " _
      &objRow.TopLevelParentCategory
   MsgBox objCol.Name & " is from dimension: " _
      &objCol.TopLevelParentCategory
   Set objRow = Nothing
   Set objCol = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "Column Object" on page 15
- "Layer Object" on page 28
- "Row Object" on page 41

Type Property

Returns the Graph object type or Query object type.

Syntax

object.**Type**

Applies To

AdvancedQuery Object

FindQuery Object

Graph Object

ParentageQuery Object

Discussion

For a Graph object, use the following list to identify the display type:

- 0 (Crosstab)
- 1 (Pie)
- 2 (3-D)
- 3 (Bar)
- 4 (Cluster)
- 5 (Stack)
- 6 (Line)
- 7 (Multi-Line)
- 8 (Correlated)
- 9 (Scatter)

For a Set object, use the following list to identify the Query type:

- 1 = FindQuery
- 2 = ParentageQuery
- 3 = AdvancedQuery
- 4 = ValueRestriction

A Graph object is the same as a display in the user interface.

Return Type

Long

Example

This example changes the display type for the first Graph object to a three-dimensional cluster bar, and displays the settings for the Graph object for an open report.

```
Sub Main()
   Dim objPPRep as Object
   Dim objPPGph as Object
   Set objPPRep = GetObject( , "CognosPowerPlay.Report")
   Set objPPGph = objPPRep.Graphs.Item(1)
   objPPGph.SetType 4, 1, 1
   MsgBox "The Graph object type is " & objPPGph.Type
& "."
   If objPPGph.Depth = -1 Then
      MsgBox "The graph is not 3D."
   Else
       MsgBox "The graph is  3D."
   End If
   Set objPPGph = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "Graphs Method" on page 157

UpperBoundary Property

Sets or returns the value defined for the upper boundary of a Range object.

Syntax

Range.**UpperBoundary**

Applies To

Range Object

Discussion

Use this method to determine the upper boundary to apply formatting when the information in the report meets the conditions set by the exception range. The upper boundary sets the maximum value for the range. For example, you may want to highlight sales when they don't exceed $50,000.

Use the LowerBoundary property to determine the lower boundary of the range.

Type

Variant

Access

Read/Write

Example

This example displays the upper boundary for the range in an Exception object.

```
Sub Main
   Dim objPPRep As Object
   Dim objPPRange As Object
   Set objPPRep = GetObject("C:\Cubes and Reports\Exception.ppx")
   objPPRep.Visible = 1
   Set objPPRange = objPPRep.Exceptions.item(1).Ranges.Item(1)
   MsgBox "Upper boundary is " &objPPRange.UpperBoundary
   Set objPPRange = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "Range Object" on page 35

UseFontSubstitution Property

Sets or returns whether to save full font information in a report published to the IBM Cognos portal.

Syntax

DeploymentOptions.**UseFontSubstitution**

Discussion

Use this property to substitute fonts on your system for those that are not available. If True, when you publish a report to the IBM Cognos portal, the DeploymentOptions object contains substituted system fonts that best match the size and appearance of the missing font. If False, font substitution is not used and the report appears exactly as the original, even if it is opened on a computer that does not have these fonts installed.

Default: False

Type

Boolean

Access

Read/Write

Example

This example sets the deployment options, including the use of system fonts, for a report published to the IBM Cognos portal.

```
Sub Main()
   Dim objPPRep as Object
   Dim objDeploymentOptions as Object
   Set objPPRep = GetObject(,"CognosPowerPlay.Report")
   Set objDeploymentOptions = objPPRep.DeploymentOptions
   objDeploymentOptions.PromptForCurrency = True
   objDeploymentOptions.PromptForLongShortNames = True
   objDeploymentOptions.PromptForZeroSuppression = True
   objDeploymentOptions.PromptForSwapRowsAndColumns =
True
   objDeploymentOptions.SelectAllDimensions
   objDeploymentOptions.UseFontSubstitution = True
   objPPRep.Save
   Set objDeploymentOptions = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "SetChartToPrint Method" on page 223
- "SetListOfLayersToPrint Method" on page 229
- "SetListOfRowsToPrint Method" on page 232

UserControl Property

Sets or returns whether the Application object is under user control.

Syntax

Application.**UserControl**

Applies To

Application Object

Discussion

If True, this property prevents the Application object from closing when the last report is closed. If False, then the Application object is hidden when the last report is closed.

Default: True

Type

Boolean

Access

Read/Write

Example

This example gives control of an open report to the user.

```
Sub Main()
   Dim objPPApp As Object
   Dim objTest As Object
   Set objPPApp = CreateObject("CognosPowerPlay.Application")
   objPPApp.Visible = True
   Set objTest = objPPApp.Reports.Add _
      ("C:\Cubes and Reports\Great Outdoors.mdc")
   Set objTest = objPPApp.Reports.Open _
      ("C:\Cubes and Reports\Exception.ppx")
   Msgbox "The number of reports in the collection is
" _
      &objPPApp.Reports.Count
   Msgbox "The name of the first report in the collection
is " _
      &objPPApp.Reports(1).Name
   objPPApp.Reports(1).Visible = True
   MsgBox objPPApp.UserControl
   Set objTest = Nothing
   Set objTest = Nothing
   Set objPPApp = Nothing
End Sub
```

Related Topics

- "Application Object" on page 11

UserColumnSummaryLabel Property

Sets or returns the user-defined label for the innermost summary column in a nested crosstab.

Syntax

object.**UserColumnSummaryLabel**

Applies To

Graph Object

Graphs

Discussion

Use this property to define a summary column label. Set the EnableUserColumnSummaryLabel property to True to use this property instead of using the default category label for the column.

The column label is only visible in a crosstab report when the ShowSummaryColumn property is set to True.

Type

String

Access

Read/Write (Graph)

Write (Graphs)

Example

This example resets the graph object to use user-defined summary labels for rows and columns.

```
Sub Main
   Dim objPPRep as Object
   Dim objPPGraph as Object
   Set objPPRep = GetObject(,"CognosPowerPlay.Report")
   Set objPPGraph = objPPRep.Graphs.Item(1)
   objPPGraph.EnableUserColumnSummaryLabel = True
   objPPGraph.UserColumnSummaryLabel = "Summary Total"
   objPPGraph.EnableUserRowSummaryLabel = True
   objPPGraph.UserRowSummaryLabel = "Summary Total"
   Set objPPGraph = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "EnableUserColumnSummaryLabel Property" on page 303
- "EnableUserRowSummaryLabel Property" on page 304

UserRowSummaryLabel Property

Sets or returns the user-defined label for the innermost summary row in a nested crosstab.

Syntax

object.**UserRowSummaryLabel**

Applies To

Graph Object

Graphs

Discussion

Use this property to define a summary row label. Set the EnableUserColumnSummaryLabel property to True to use this property instead of using the default category label for the row.

The row label is only visible in a crosstab report when the ShowSummaryRow property is set to True.

Type

String

Access

Read/Write (Graph)

Write (Graphs)

Example

This example resets the graph object to use user-defined summary labels for rows and columns.

```
Sub Main
   Dim objPPRep as Object
   Dim objPPGraph as Object
   Set objPPRep = GetObject(,"CognosPowerPlay.Report")
   Set objPPGraph = objPPRep.Graphs.Item(1)
   objPPGraph.EnableUserColumnSummaryLabel = True
   objPPGraph.UserColumnSummaryLabel = "Summary Total"
   objPPGraph.EnableUserRowSummaryLabel = True
   objPPGraph.UserRowSummaryLabel = "Summary Total"
   Set objPPGraph = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "EnableUserColumnSummaryLabel Property" on page 303
- "EnableUserRowSummaryLabel Property" on page 304

UseScrolling Property

Sets or returns whether scrolling is enabled.

Syntax

object.**UseScrolling**

Applies To

Graph Object

Graphs

Discussion

Use this property to turn the scrolling feature on or off. If the value is True (-1), scrolling is enabled when the total number of bars exceed the maximum number of visible bars. If the value is False (0), scrolling is disabled.

If this property is False, IBM Cognos PowerPlay will attempt to graph all data up to a maximum of 5000 bars. If this limit is exceeded, a warning message appears.

Scrolling is not available for crosstab, 3-D bar, scatter, and pie displays.

Default: True

Type

Boolean

Access

Read/Write (Graph)

Write (Graphs)

Example

This example changes the scroll bar settings of an active report. The scrolling feature is set to True, maximum visible bars is set to six, maximum printed bars is set to ten, and the summary category is kept hidden until the end of the report.

```
Sub Main()
   Dim objPPRep As Object
   Set objPPRep = GetObject(,"CognosPowerPlay.Report")
   objPPRep.Graphs.Item(1).UseScrolling = TRUE
   objPPRep.Graphs.Item(1).MaxVisibleBars = 6
   objPPRep.Graphs.Item(1).MaxPrintedBars = 10
   objPPRep.Graphs.Item(1).KeepSummaryVisible = FALSE
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "Graph Object" on page 25
- "Graphs" on page 56

ValuesAutoFit Property

Sets or returns whether value labels fit within graph bars and pie segments.

Syntax

object.**ValuesAutoFit**

Applies To

Graph Object

Graphs

Discussion

When creating graphs, you can optionally display the associated numerical values inside the bars or pie sections. Set the ValuesShown property to True to display

such values. Set this property to True to automatically adjust the value labels to fit the available bar or pie width. ValuesAutoFit uses the specified font but adjusts the size as required to make values fit. (The largest size it uses is the specified font size.) If set to False, the default font size may cause the value to overwrite bars and other labels.

Default: True

Type

Boolean

Access

Read/Write (Graph)

Write (Graphs)

Example

This example sets the first graph object of the active report to a horizontal simple bar chart and sets the properties for the values shown on the bars.

```
Sub Main
   Dim objPPRep as Object
   Dim objPPGraph as Object
   Set objPPRep = GetObject(,"CognosPowerPlay.Report")
   Set objPPGraph = objPPRep.Graphs.Item(1)
   With objPPGraph
      .SetType 3,0,0
      .ValuesShown = True
      .ValuesPosition = 0
      .ValuesAutoFit = True
      .ValuesFontColor = 10
      .ValuesFontSize = 10
      .ValuesFontName = "Times New Roman"
   End With
   Set objPPGRaph = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "ValuesFontColor Property"
- "ValuesFontName Property" on page 423
- "ValuesFontSize Property" on page 424
- "ValuesPosition Property" on page 427
- "ValuesShown Property" on page 428

ValuesFontColor Property

Sets or returns the font color used for graph value labels.

Syntax

object.**ValuesFontColor**

Applies To

Graph Object

Graphs

Discussion

Use this property change or display the numerical values for the color associated with the bar, pie section, or line graph. Set the ValuesShown property to True to display these values.

The font color types are

0 = Black

1 = Maroon

2 = Green

3 = Olive

4 = Navy

5 = Purple

6 = Teal

7 = Gray

8 = Silver

9 = Red

10 = Lime

11 = Yellow

12 = Blue

13 = Fuschia

14 = Aqua

15 = White

Default: 0 (black)

Type

Long

Access

Read/Write (Graph)

Write (Graphs)

Example

This example sets the first graph object of the active report to a horizontal simple bar chart and sets the properties for the values shown on the bars.

```
Sub Main
   Dim objPPRep as Object
   Dim objPPGraph as Object
   Set objPPRep = GetObject(,"CognosPowerPlay.Report")
   Set objPPGraph = objPPRep.Graphs.Item(1)
   With objPPGraph
      .SetType 3,0,0
      .ValuesShown = True
      .ValuesPosition = 0
      .ValuesAutoFit = False
      .ValuesFontColor = 10
      .ValuesFontSize = 10
      .ValuesFontName = "Times New Roman"
   End With
   Set objPPGRaph = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "ValuesAutoFit Property" on page 420
- "ValuesFontName Property"
- "ValuesFontSize Property" on page 424
- "ValuesPosition Property" on page 427
- "ValuesShown Property" on page 428

ValuesFontName Property

Sets or returns the font name used for graph value labels.

Syntax

object.**ValuesFontName**

Applies To

Graph Object

Graphs

Discussion

When you create graphs, you can optionally display the associated numerical values that go with the bars, pie sections, or lines. Set the ValuesShown property to True to display such values. Use the ValuesFontName property to change the current font type. Use ValuesFontSize to change the size.

Default: Arial

Type

String

Access

Read/Write (Graph)

Write (Graphs)

Example

This example sets the first graph object of the active report to a horizontal simple bar chart and sets the properties for the values shown on the bars.

```
Sub Main
   Dim objPPRep as Object
   Dim objPPGraph as Object
   Set objPPRep = GetObject(,"CognosPowerPlay.Report")
   Set objPPGraph = objPPRep.Graphs.Item(1)
   With objPPGraph
      .SetType 3,0,0
      .ValuesShown = True
      .ValuesPosition = 0
      .ValuesAutoFit = False
      .ValuesFontColor = 10
      .ValuesFontSize = 10
      .ValuesFontName = "Times New Roman"
   End With
   Set objPPGRaph = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "ValuesAutoFit Property" on page 420
- "ValuesFontColor Property" on page 421
- "ValuesFontSize Property"
- "ValuesPosition Property" on page 427
- "ValuesShown Property" on page 428

ValuesFontSize Property

Sets or returns the font size used for graph value labels.

Syntax

object.**ValuesFontSize**

Applies To

Graph Object

Graphs

Discussion

When you create graphs, you can optionally display the associated numerical values that go with bars, pie sections, or lines. Set the ValuesShown property to True to display such values. Use the ValuesFontSize property to change the size of the current font. Use ValuesFontName to change the font.

Default: 10 point

Type

Long

Access

Read/Write (Graph)

Write (Graphs)

Example

This example sets the first graph object of the active report to a horizontal simple bar chart and sets the properties for the values shown on the bars.

```
Sub Main
   Dim objPPRep as Object
   Dim objPPGraph as Object
   Set objPPRep = GetObject(,"CognosPowerPlay.Report")
   Set objPPGraph = objPPRep.Graphs.Item(1)
   With objPPGraph
      .SetType 3,0,0
      .ValuesShown = True
      .ValuesPosition = 0
      .ValuesAutoFit = False
      .ValuesFontColor = 10
      .ValuesFontSize = 10
      .ValuesFontName = "Times New Roman"
   End With
   Set objPPGRaph = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "ValuesAutoFit Property" on page 420
- "ValuesFontColor Property" on page 421
- "ValuesFontName Property" on page 423
- "ValuesPosition Property" on page 427
- "ValuesShown Property" on page 428

ValuesFontStyle Property

Sets or returns the font style used for graph value labels.

Syntax

object.**ValuesFontStyle**

Applies To

Graph Object

Graphs

Discussion

Use this property change or display the font style associated with the Simple Bar, Clustered Bar, Single Line, and Correlation graphs.

The available font styles are:

0 - Regular 1 - Italic 2 - Bold 3 - BoldItalic

Default: 0

Type

Long

Access

Read/Write (Graph)

Write (Graphs)

Example

This example sets the first graph object of the active report to a horizontal simple bar chart and sets the properties for the values shown on the bars.

```
Sub Main
   Dim objPPRep as Object
   Dim objPPGraph as Object
   Set objPPRep = GetObject(,"CognosPowerPlay.Report")
   Set objPPGraph = objPPRep.Graphs.Item(1)
   With objPPGraph
      .SetType 3,0,0
```

```
      .ValuesShown = True
      .ValuesPosition = 0
      .ValuesAutoFit = False
      .ValuesFontColor = 10
      .ValuesFontName = "Times New Roman"
      .ValuesFontStyle = 3
   End With
   Set objPPGRaph = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "ValuesAutoFit Property" on page 420
- "ValuesFontColor Property" on page 421
- "ValuesFontName Property" on page 423
- "ValuesPosition Property"
- "ValuesShown Property" on page 428

ValuesPosition Property

Sets or returns the position of value labels on some graph types.

Syntax

object.**ValuesPosition**

Applies To

Graph Object

Graphs

Discussion

When you create graphs, you can optionally display the associated numerical values with some bar graphs. The positioning of values is only valid for simple bar or cluster bar graphs.

Set the ValuesShown property to True to display these values.

Default: 0 (False)

Type

Short

Access

Read/Write (Graph)

Write (Graphs)

Example

This example sets the first graph object of the active report to a horizontal simple bar chart and sets the properties for the values shown on the bars.

```
Sub Main
   Dim objPPRep as Object
   Dim objPPGraph as Object
   Set objPPRep = GetObject(,"CognosPowerPlay.Report")
   Set objPPGraph = objPPRep.Graphs.Item(1)
   With objPPGraph
      .SetType 3,0,0
      .ValuesShown = True
      .ValuesPosition = 0
      .ValuesAutoFit = False
      .ValuesFontColor = 10
      .ValuesFontSize = 10
      .ValuesFontName = "Times New Roman"
   End With
   Set objPPGRaph = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "ValuesAutoFit Property" on page 420
- "ValuesFontColor Property" on page 421
- "ValuesFontName Property" on page 423
- "ValuesFontSize Property" on page 424
- "ValuesShown Property"

ValuesShown Property

Sets or returns whether value labels appear next to pie chart slices.

Syntax

object.**ValuesShown**

Applies To

Graph Object

Graphs

Discussion

This property applies to single pie charts only. An error occurs if you set this property to True for nested charts.

Set ValuesShown to True to label slices by using a category value instead of a category name. The legend includes category names only.

Setting the NamesShown property to True automatically sets the ValuesShown property to False. If both properties are set to False, then no labels appear beside pie chart slices.

Default: True

Type

Boolean

Access

Read/Write (Graph)

Write (Graphs)

Example

This example sets the first graph object of the active report to a horizontal simple bar chart and sets the properties for the values shown on the bars.

```
Sub Main
   Dim objPPRep as Object
   Dim objPPGraph as Object
   Set objPPRep = GetObject(,"CognosPowerPlay.Report")
   Set objPPGraph = objPPRep.Graphs.Item(1)
   With objPPGraph
      .SetType 3,0,0
      .ValuesShown = True
      .ValuesPosition = 0
      .ValuesAutoFit = False
      .ValuesFontColor = 10
      .ValuesFontSize = 10
      .ValuesFontName = "Times New Roman"
   End With
   Set objPPGRaph = Nothing
   Set objPPRep = Nothing
End Sub
```

Related Topics

- "NamesShown Property" on page 348

Version Property

Returns the version number of IBM Cognos PowerPlay.

Syntax

Application.**Version**

Applies To

Application Object

Discussion

Use this property to determine the version of PowerPlay you are using.

Type

String

Access

Read

Example

This example creates an instance of the PowerPlay Application object and shows the name, location, and version.

```
Sub Main()
   Dim objPPlayApp as Object
   Set objPPlayApp = CreateObject("CognosPowerPlay.Application")
   objPPlayApp.Visible = 1
   MsgBox "The name of the Application is " &objPPLayApp.Name
   MsgBox "The location of the Application is " _
      &objPPLayApp.Path
   MsgBox "The Application version is " &objPPLayApp.Version
   Set objPPlayApp = Nothing
End Sub
```

Related Topics

- "Application Object" on page 11

Visible Property

Sets or returns whether the object is visible to the user.

Syntax

object.**Visible**

Applies To

Application Object

Dimension Object

DimensionLine Object

Report Object

Discussion

By default, IBM Cognos PowerPlay runs invisibly when invoked from within a macro. To see what a macro is doing, set the Visible property to True. By displaying PowerPlay, you can debug the macros more effectively. In addition, if errors occur when PowerPlay is running invisibly or when the macro has not

specified a Quit method, PowerPlay remains in memory. If you continue to leave PowerPlay in memory every time you run the macro or it fails, you will eventually run out of memory.

This property is not valid while under user control. For example, when the Applications objects is visible, the UserControl property is set to True.

Default: False

Type

Boolean

Access

Read/Write

Example

This example creates a PowerPlay Application object and sets the application to visible.

```
Sub Main()
   Dim objPPlayApp as Object
   Set objPPlayApp = CreateObject("CognosPowerPlay.Application")
   objPPlayApp.Visible = 1
   MsgBox "The title of the application is " &objPPlayApp.Caption
   MsgBox "The name of the Application is " &objPPLayApp.Name
   MsgBox "The location of the Application is " _
      &objPPLayApp.Path
   MsgBox "The Application version is " &objPPLayApp.Version
   Set objPPlayApp = Nothing
End Sub
```

Related Topics

- "Application Object" on page 11
- "Dimension Object" on page 19
- "DimensionLine Object" on page 21
- "Report Object" on page 37

Chapter 6. Administrative Macros

IBM Cognos PowerPlay uses administrative macros to perform tasks for specific administrative events. An administrative macro contains a customized set of instructions to perform one or more actions

- when the PowerPlay application opens and closes
- when a report or cube opens and closes
- after PowerPlay opens a report or cube, for example, to change the appearance, perform calculations, or drill down to the lowest level categories
- when exceptions are triggered in a report

Create Administrative Macros

You create an administrative macro just as you do a report-level macro, however you must assign the specific name of the macro you want IBM Cognos PowerPlay to run automatically. For example, if you want PowerPlay to recognize the AppOpen macro, the file AppOpen.mac must exist in the Macros directory.

The table below describes the PowerPlay administrative macros, the names to assign the files for PowerPlay to automatically run them, and identifies the order in which PowerPlay executes them.

Administrative macro	**Description**
AppOpen Macro	Performs startup operations when PowerPlay starts and before it opens a cube or report.
DocOpen Macro	Runs administrative tasks when PowerPlay opens a report or cube. Any instructions in the macro are executed before the report or cube opens.
After Doc Open Macro	Used for local document processing, such as performing calculations. This macro must have the same file name (with the .mac extension) and be stored in the same folder location as the report.
Highlight Exceptions Macro	Highlights exceptions. After you specify the use of this function in the Custom Exceptions dialog box, PowerPlay runs this function once for every cell in a report.
DocClose Macro	Performs cleanup operations required after running the DocOpen macro. This macro starts when you close the report.
AppClose Macro	Performs cleanup operations required after running the AppOpen macro. This macro starts when you close PowerPlay.

Examples

The following examples demonstrate some specific uses for administrative macros:

- To ensure that users have the latest cube, use the DocOpen macro to verify the version when PowerPlay is opened.
- To delete any empty or unused files, use the DocClose macro when you close PowerPlay.
- To automatically drill down to the lowest level of a category, use the After Doc Open macro.
- To open PowerPlay and set preferences, use the AppOpen macro.
- To reset any preferences after a user quits the application, use the AppClose macro.
- To highlight any exceptions in a report, use the Exception macro.
- To preset security passwords for a cube or report, use the AppOpen macro.

After Doc Open Macro

A macro used for local document processing, such as changing the display or performing calculations.

Discussion

Use this administrative macro to automate many of the routine tasks to perform after IBM Cognos PowerPlay opens a report, such as drilling down to the lowest level categories or displaying a view other than the crosstab. If this macro exists in the Macros directory, PowerPlay runs it immediately after the DocOpen macro completes, and the report has opened. For example, to open a popular sales report, Sales.rpt, and add your own exception highlighting and display selections, you include the scripting for these preferences in the After Doc Open macro file, Sales.Mac.

Requirements

- An After Doc Open macro must have the same file name (with the .mac extension) and be stored in the same folder location as the report. Only one After Doc Open macro file can be associated with a single report.
- The only function that must be included in the macro is the Main subroutine.
- An administrator should maintain this macro centrally in the Macros directory specified in the PowerPlay Preferences dialog box.

Example

This example automatically adds the lowest-level categories for the first row and then deletes the first row using the report (Reporter mode) that is open.

```
Sub Main()
  On Error Resume Next
  Dim objPPRep as Object
  Dim objRow as Object
  'Get the report object
  Set objPPRep = GetObject(,"CognosPowerPlay.Report")
  if Err <> 0 Then
```

```
   Msgbox "Error getting PowerPlay report. " + Error$
  Else
  'Make sure that there is at least one row in the report
   If (objPPRep.Rows.Count > 0) then
  'Get the first row category object
    Set objRow = objPPRep.Rows(1)
  'Add lowest-level categories for this row
    objRow.AddLowestLevelCategories
  'Remove first row category
    objRow.Remove
   End if
  End If
  Set objRow = Nothing
  Set objPPRep = Nothing
End sub
```

Related Topics

- "AppClose Macro"
- "AppOpen Macro" on page 436
- "DocClose Macro" on page 437
- "DocOpen Macro" on page 438
- "Highlight Exceptions Macro" on page 439

AppClose Macro

A macro that performs cleanup operations required after running the AppOpen macro and before exiting IBM Cognos PowerPlay.

Discussion

Use this administrative macro to

- close any documents the macro opened but the user no longer needs
- save, or prompt the user to save, a file that should be saved
- restore the settings of any options that the macro may have changed
- delete any temporary files

If this macro exists in the Macros directory, PowerPlay runs it when exiting PowerPlay. For example, to reset any preferences after a user exits the application, include scripting in AppClose.mac.

Requirements

- The only function that must be included in the macro is the Main subroutine.
- The Macros and Custom Menu File box in the Preferences dialog box must name the folder that contains the AppClose macro.
- An administrator should maintain this macro centrally in the Macros directory specified in the PowerPlay Preferences dialog box.

Example

This example uses a message box to display the words "AppClose Macro" to indicate the macro ended successfully.

```
Sub Main
  msgbox "AppClose Macro"
End sub
```

Related Topics

- "After Doc Open Macro" on page 434
- "AppOpen Macro"
- "DocClose Macro" on page 437
- "DocOpen Macro" on page 438
- Chapter 6, "Administrative Macros," on page 433

AppOpen Macro

A macro that performs startup operations when IBM Cognos PowerPlay launches and before it opens a report or cube.

Syntax

Function AppOpenMacro *(StartMode As Long) As Long* **End Function**

Discussion

Use this macro to automate redundant or repetitive operations typically completed after PowerPlay starts, such as

- setting user preferences
- automatically opening a particular report
- presetting security passwords for a cube or report for specific user access to avoid entering a password on the user's personal computer

If this macro exists in the Macros directory, it runs when PowerPlay launches and returns True or False to indicate the status of the macro execution. If the macro returns False, the application is not opened. For example, to set any preferences immediately after launching PowerPlay and before opening a cube or report, include scripting in the AppOpen.mac administrative macro file.

Requirements

- The macro must contain a function called AppOpenMacro, which returns True or False (-1 or 0 respectively). If the macro does not exist, the function name in the macro is incorrect, or the number of parameters are mismatched, the macro does not run and the application opens normally.
- An administrator should maintain this macro centrally in the Macros directory specified in the PowerPlay Preferences dialog box.

Parameter	Description
StartMode	Required. Specifies how the application starts. 1 = Normal startup 2 = OLE Type: Long

Return Type

Long

Example

This example automatically opens the report "C:\Cognos\Sample.ppx" when PowerPlay starts.

```
Function AppOpenMacro(lStartMode as Long) as Long
  On Error Resume Next
  Dim objPPRep as Object
  Dim ObjPPApp as object
  'Get the application object
  Set objPPApp = GetObject(, "CognosPowerPlay.Application")
  'Open the specified report
  Set objPPRep = objPPApp.reports.open("c:\cognos\sample.ppx")
  if Err <> 0 then
    Msgbox "Unable to open PowerPlay report."
    AppOpenMacro = False
  Else
  'Make the report visible
    objPPRep.visible = True
  'Set return code to indicate successful completion
   AppOpenMacro = True
  End if
  Set objPPRep = Nothing
  Set objPPApp = Nothing
End Function
```

Related Topics

- "After Doc Open Macro" on page 434
- "AppClose Macro" on page 435
- "DocClose Macro"
- "DocOpen Macro" on page 438
- "Highlight Exceptions Macro" on page 439
- Chapter 6, "Administrative Macros," on page 433

DocClose Macro

A macro that performs cleanup operations required after running the DocOpen macro.

Discussion

Use this macro when you want IBM Cognos PowerPlay to perform closing activities, such as

- deleting any unused or empty files
- notifying the user when the macro has completed
- saving, or prompting the user to save, a file that should be saved

If this macro exists in the Macros directory, it runs only when you exit the report. For example, to delete any temporary files when PowerPlay closes a report, include scripting in the DocClose.mac administrative macro file.

Requirements

The only function that must be included in the macro is the Main subroutine.

A central administrator should maintain this macro in the Macros directory specified in the PowerPlay Preferences dialog box.

Example

This example displays a message box with the words "DocClose Macro" to indicate the macro has run.

```
Sub Main
  msgbox "DocClose Macro"
End sub
```

Related Topics

- "After Doc Open Macro" on page 434
- "AppClose Macro" on page 435
- "AppOpen Macro" on page 436
- "DocOpen Macro"
- "Highlight Exceptions Macro" on page 439
- Chapter 6, "Administrative Macros," on page 433

DocOpen Macro

A macro that runs administrative tasks when IBM Cognos PowerPlay opens a report or cube.

Syntax

Function DocOpenMacro *(StartMode As Long, ReportName As String, CubeName As String, CubePassword As String) As Long* **End Function**

Discussion

Use this macro to add administrative functions, such as document security or statistical logging. If this macro exists in the Macros directory. It runs when the report or cube opens. If the macro returns False, the document is not opened. For example, to record the types of reports that your organization creates in a database, include scripting in the DocOpen.mac file.

Requirements

- The macro, DocOpen.mac, must contain a function called DocOpenMacro, which returns a True or False (-1 or 0 respectively) value. If the macro does not exist, the function name in the macro is incorrect, or the number of parameters are mismatched, the macro does not run and the report or cube opens normally.

An administrator should maintain this macro centrally in the Macros directory specified in the PowerPlay Preferences dialog box.

Parameters	Description
StartMode	Required. Specifies how to open the report. 1 = Normal startup 2 = OLE Type: Long
ReportName	Required. Specifies the report file name being opened. Type: String
CubeName	Required. Specifies the .mdc file name that the report uses. Type: String
CubePassword	Optional. Specifies logon information for protected cubes. Type: String

Return Type

Long

Example

This example logs all reports and cubes that were opened to a text file.

```
Function DocOpenMacro( StartMode As Long, ReportName
As String, CubeName As String, Unused1 As String, CubePassword As
String, Unused2 As String, Unused3 As String, Unused4 As String)
As Long
   Dim strLogFile As String
   strLogFile = "c:\log.txt"
   Open strLogFile For Append As #1
   Print #1, "Report: " & ReportName & " Cube:
" & CubeName
   Close #1
   DocOpenMacro = 1
End Function
```

Related Topics

- "After Doc Open Macro" on page 434
- "AppClose Macro" on page 435
- "AppOpen Macro" on page 436
- "Highlight Exceptions Macro"
- Chapter 6, "Administrative Macros," on page 433

Highlight Exceptions Macro

A macro that highlights exceptions in a report.

Syntax

Function ExceptionMacro *(DimensionLineSetting as String, RowLabel as String, ColumnLabel as String, LayerLabel as String, ExceptionName as String, ValueOfCell as Double) As Long ComparisonCode* **End Function**

Discussion

Use this macro to add extra administrative functions, such as document security or statistical logging. After you specify the use of this macro in the Custom Exceptions dialog box, IBM Cognos PowerPlay runs this macro function once for every cell in a report. For example, to highlight values in cells (with a particular exception style) that match a specified criteria, include the scripting in the ExceptionMacroName.mac file.

Requirements

- The macro must include a function called ExceptionMacro.
- The macro must be located in the Macro directory specified in the Preferences dialog box.
- You can use this macro for one or more valid report file names.
- PowerPlay runs this function once for every cell in the report, so do not include any code for the ComparisonCode parameter that shows message boxes because each message box will appear at least once for each cell. For example, an exception macro uses a message box to indicate that a cell is an exception. If the report has 2,400 cells whereby 1,834 cells meet the exception criteria, 1,834 messages will appear as each cell is tested.
- Each time there is a change to a category, such as drilling down, adding a row, or performing calculations, PowerPlay runs the function again for every cell in the report.
- An administrator should maintain this macro centrally in the Macros directory specified in the PowerPlay Preferences dialog box.

Parameters	**Description**
DimensionLineSetting	Required. Specifies the Dimension line settings that PowerPlay passes to the exception macro. Type: String
RowLabel	Required. Specifies the Row label that PowerPlay passes to the exception macro. Type: String
ColumnLabel	Required. Specifies the Column label that PowerPlay passes to the exception macro. Type: String
LayerLabel	Required. Specifies the Layer label that PowerPlay passes to the exception macro. Type: String

Parameters	Description
ExceptionName	Required. Specifies the name of the exception that PowerPlay passes to the exception macro. Type: String
ValueOfCell	Required. Specifies the value of the cell that PowerPlay passes to the macro. Type: Double
ComparisonCode	Required. Identifies whether the style associated with the exception is applied to a cell. The ExceptionFunction macro determines the value returned. 0 = do not apply the style 1 = apply the style Type: Integer

Return Type

Long

Example

This example applies the exception style to each cell where the value in the cell is less than zero. When the macro returns a value of 1, it applies the style associated with the specified exception to the cell. When the macro returns 0, it does not apply the exception style to the cell.

```
Function ExceptionMacro (PPDimLine as String, PPRow as
String, PPCol as String, PPLayer as String, PPExName as String,
PPCellV as Double) As Long
  if PPCellV < 0 then
    ExceptionMacro = 1
  else
    ExceptionMacro = 0
  end if
End Function
```

Related Topics

- "After Doc Open Macro" on page 434
- "AppClose Macro" on page 435
- "AppOpen Macro" on page 436
- "DocClose Macro" on page 437
- "DocOpen Macro" on page 438
- Chapter 6, "Administrative Macros," on page 433

Notices

This information was developed for products and services offered worldwide.

This material may be available from IBM in other languages. However, you may be required to own a copy of the product or product version in that language in order to access it.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service. This document may describe products, services, or features that are not included in the Program or license entitlement that you have purchased.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan Ltd.
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokyo 103-8510, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Software Group
Attention: Licensing
3755 Riverside Dr.
Ottawa, ON
K1V 1B7
Canada

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information is for planning purposes only. The information herein is subject to change before the products described become available.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

Depending upon the configurations deployed, this Software Offering may use session and persistent cookies that collect each user's

- name
- user name
- password

for purposes of

- session management
- authentication
- enhanced user usability
- single sign-on configuration
- usage tracking or functional purposes other than session management, authentication, enhanced user usability and single sign-on configuration

These cookies cannot be disabled.

If the configurations deployed for this Software Offering provide you as customer the ability to collect personally identifiable information from end users via cookies and other technologies, you should seek your own legal advice about any laws applicable to such data collection, including any requirements for notice and consent.

For more information about the use of various technologies, including cookies, for these purposes, see IBM's Privacy Policy at http://www.ibm.com/privacy and IBM's Online Privacy Statement at http://www.ibm.com/privacy/details in the section entitled "Cookies, Web Beacons and Other Technologies" and the "IBM Software Products and Software-as-a-Service Privacy Statement" at http://www.ibm.com/software/info/product-privacy.

Index

Numerics

A

B

C

D

E

F

G

H

I

K

L

M

N

O

P

Q

R

S

T

U

V

W

Z