crystal decisions™

Crystal Reports

Using VB Variables with the Crystal VBX/OCX Formulas() Property (Crystal formulas)

Overview

This document describes how to use variables when setting a Crystal formula at runtime with the VBX/OCX control. Several methods are covered including string, numeric, and date data types.

Contents

Introduction

VB variables can be used with the Formulas() property if they are declared As String or converted to string (regardless of the actual data type) before being passed to Crystal Reports. The value of the variable passed to the report must be in the same format and look exactly as if entered directly into the Crystal Reports Formula Editor.

NOTE	The following code also works with the OCX but the Control name defaults to CrystalReport1 instead of Report1 (as it does with the VBX).

Important Notes

String (enclose in single quotes ‘This is a String’ or double quotes "This is a String")

Numbers (do not need quotes) 1234

Dates (must look like a Date(Date(YYYY,MM,DD) function)

The Formulas() property can only be used to change formulas that already exist in Crystal Reports, not to create new formulas. The Formulas() property has an array index that increments if more than one formula is modified in a Crystal Custom Control:

```
Report1.Formulas(0) = "Stringfield=" & CHARSTR
Report1.Formulas(1) = "Numberfield=" & NUM
Report1.Formulas(2) = "Datefield=" & WHOLEDATE
```

If only one formula is being modified per Control, then the parenthesis will contain a zero:

```
Report1.Formulas(0) = "Datefield=" & WHOLEDATE
```

It doesn’t matter how many formulas there are in a Crystal report or in what order they are entered on the report, the array index is affected only by the number of formulas modified in a Control. The total number and order of entry of formulas in a Crystal report has nothing to do with the Formulas() property array index.

Examples Overview

The following examples assume:

- A Crystal report has been set up with three formulas:

Formula 1 name: **@Stringfield " "**

// The quotes with a space serve as a place-holder to receive a string value *NOTE* Any formula that results in a string data type can be used instead of this formula

Formula 2 name: **@Numberfield 0**

// The zero serves as a place-holder to receive a numeric value *NOTE* Any formula that results in a numeric data type can be used instead of this formula

Formula 3 name: **@Datefield Date(1990,01,31)**

// The Date() with any date serves as a place-holder to receive a date *NOTE* Any formula that results in a date data type can be used instead of this formula

- Although @ is appended to the beginning of each formula name when a formula is inserted in Crystal Reports, the @ must not be included in the Formulas() statement in VB when passing Crystal the formula name
- The three formulas are all modified in one Custom Control
- The variable names used in VB are:

  CHARSTR

  NUM

  WHOLEDATE

  YEARDATE

  MONTHDATE

  DAYDATE

- Three TextBoxes have been placed on a VB form:

  StringEntry.Text - for the string examples

  NumberEntry.Text - for the number examples

  DateEntry.Text - for whole date examples.

- Three additional TextBoxes have been placed on the form as needed for the date examples:

  YearEntry.Text - for the year
  MonthEntry.Text - for the month
  DayEntry.Text - for the day

- Ampersand (&) versus Plus (+) in the Formulas() statement: The (+) works fine if the Variable is declared As String in VB. However if the Variable is declared As Integer, using the (+) will result in a type mismatch error in VB because the Crystal formula name is a string. A string cannot be concatenated to an integer. The Ampersand & actually forces a concatenation by converting the integer to a Variant of VarType 8 which is a string. The Ampersand & is used in most of the examples in this document as it works for both string and integer variable declarations.

- Single quotes are passed to Crystal Reports String formula fields although double quotes can be used.

Passing a variable from VB that results in a String data type in Crystal Reports

As both VB and Crystal Reports require string values to be surrounded by quotes, a variable that results in a string data type in Crystal Reports requires two sets of quotes; one set for VB and one set for Crystal. There are a number of ways to include the extra set of quotes needed by Crystal Reports.The first two examples show how to pass the variable with the extra quotes when the value of the variable is hard-coded.The third example shows how to pass the variable with the extra quotes when the value of the variable is entered via a TextBox at runtime.

Example 1 has the value hard-coded and includes the extra quotes while assigning the variable a value:

```
Dim CHARSTR As String
'Declare CHARSTR as a string variable.


CHARSTR = "'This is a string '"
'Assign a value to the variable. Note the use of two sets
of quotes.


Report1.Formulas(0) = "Stringfield=" & CHARSTR
'Concatenate the Crystal Reports formula name with the VB
'variable name.


Msgbox Report1.Formulas(0)
'Display the exact value that is passed to Crystal Reports
in a message box
'This message box will display:
'Stringfield= 'This is a string'
'The Crystal report will print: This is a string
```

Example 2 has the value hard-coded and includes the extra quotes in the Formulas() statement:

```
Dim CHARSTR As String
'Declare CHARSTR as a string variable.


 CHARSTR = "This is a string"
'Assign a value to the variable.Note the use of one set of
'quotes.
```

```
 Report1.Formulas(0) = "Stringfield= '" & CHARSTR & "'"
'Concatenate the Crystal Reports formula name with an open
'quote, the VB variable name, and a closed quote.


 Msgbox Report1.Formulas(0)
'Display the exact value that is passed to Crystal Reports
in a message box.This message box will display:
'Stringfield= 'This is a string'
'The Crystal report will print: This is a string
```

Example 3 shows how to pass a variable when the value is entered via a TextBox at runtime:

```
Dim CHARSTR As String
'Declare CHARSTR as a string variable.


 CHARSTR = StringEntry.Text
'Assign the contents of the StringEntry 'TextBox to the
CHARSTR variable.


 Report1.Formulas(0) = "Stringfield='" & CHARSTR & "'"
'Concatenate the Crystal Reports formula name with an open
quote, the VB variable 'name, and a closed quote.


 Msgbox Report1.Formulas(0)
'Display the exact value that is passed to Crystal Reports
in a message box. This message box will display the Crystal
formula name, an equal sign, and whatever was entered in
the StringEntry TextBox at runtime. The Crystal report will
print what was entered in the StringEntry TextBox at
runtime.
```

Passing a variable from VB that results in a Numeric data type in Crystal Reports

Because the Formulas() property passes only string data types, variables declared As Integer must be converted to a string before being passed to Crystal Reports. The first example shows a number in a variable declared As String. The second example shows that a variable declared As Integer can be converted to a string simply by using the Ampersand & in the Formulas() statement. The third example converts an integer to a string using the str$() and a Plus + in the Formulas() statement. The fourth example shows how to convert and pass a variable when the value of the variable is entered via a TextBox at runtime.

Example 1 has the value hard-coded and the variable declared As String:

```
Dim NUM As String
'Declare NUM as a string variable.

NUM = "1234"
'Assign a value to the variable. Note the use of one set of
quotes.

 Report1.Formulas(1) = "Numberfield=" & NUM
'Concatenate the Crystal Reports formula name with the VB
variable name. Plus (+) could be used instead of Ampersand
(&) as NUM is as a string and needs no conversion

 Msgbox Report1.Formulas(1)
'Display the exact value that is passed to Crystal Reports
in a message box.
'This message box will display: Numberfield= 1234
'The Crystal report will print: 1234 (a numeric data type)
```

Example 2 has the value hard-coded and the variable declared As Integer:

```
Dim NUM As Integer
'Declare NUM as an Integer variable.

NUM = 1234
'Assign a value to the variable.

Report1.Formulas(1) = "Numberfield=" & NUM
'Concatenate the Crystal Reports name formula with the VB
variable name. The Ampersand & converts NUM to a string.

 Msgbox Report1.Formulas(1)
'Display the exact value that is passed to Crystal Reports
in a message box.
'This message box will display: Numberfield= 1234
'The Crystal report will print: 1234 (a numeric data type)
```

Example 3 has the value hard-coded and converts the variable with str$() in the Formulas() statement:

```
Dim NUM As Integer
'Declare NUM as a integer variable.

NUM = 1234
```

```
'Assign a value to the variable.


 Report1.Formulas(1) = "Numberfield=" + Str$( NUM)

'Convert variable and concatenate the Crystal Reports
formula 'namewith the VB variable name. Plus + is used
because the Str$() function converts NUM to string


 Msgbox Report1.Formulas(1)

'Display the exact value that is passed to Crystal Reports
in a message box.

'Message box will display: Numberfield= 1234

'Crystal report will print: 1234 (a numeric data type)
```

Example 4 shows how to pass a variable when the value is entered via a TextBox at runtime:

```
Dim NUM As Integer

'Declare NUM as an integer variable.


NUM = NumberEntry.Text

'Assign the contents of the NumberEntry TextBox to the NUM
'variable.


Report1.Formulas(1) = "Numberfield=" & NUM

'Concatenate the Crystal Reports formula name with the VB
'variable name.


Msgbox Report1.Formulas(1)

'Display the exact value that is passed to Crystal Reports
in a message box. This message box will display the Crystal
formula name, an equal sign, and whatever was entered in
the NumberEntry TextBox at runtime.The Crystal report will
print what was entered in the NumberEntry TextBox at
runtime.
```

Passing a variable from VB that results in a Date data type in Crystal Reports

Crystal Reports accepts dates only as a string data type in the Date(yyyy,mm,dd) format. The characters that make up Date(yyyy,mm,dd) must be concatenated with the Crystal Reports formula name and the value of the date variable.

The first example shows the date hard-coded and assigned to a single variable as a numeric string. The second example shows the date hard-coded and assigned to three variables as numeric strings. The third example shows how to format when the date is entered via three TextBoxes at runtime. The fourth example uses the VB CVDate() and Format() functions to convert and format a

hard-coded short or long date. The fifth example uses the VB CVDate() and Format() functions to convert and format a short or long date entered via a TextBox at runtime.

Example 1 shows the date hard-coded and assigned to a single variableas a numeric string Formatting is done in the Formulas() statement:

```
Dim WHOLEDATE

'Declare WHOLEDATE as a variable


 WHOLEDATE = "1995, 01, 31"

'Assign a value to the variable. Note that the year is 4
digits, the month andday are 2 digits. The year, month and
day are in the order Crystal expects and are delimited by
commas, and the entire string is surrounded by quotes.


Report1.Formulas(2) = "Datefield= Date("& WHOLEDATE & ")"

'Concatenate the Crystal Reports formula name with the word
Date, an open parenthesis, the VB variable name, and a
closed parenthesis.


MsgBox Report1.Formulas(2)

'Display the exact value that is passed to Crystal Reports
in a message box. This message box will display:

'Datefield= Date(1995,01,31)

'The Crystal report will print the Windows default (unless
otherwise 'specified in Crystal): 95/1/31
```

Example 2 shows the date hard-coded and assigned to three variables as strings. Formatting is done in the Formulas() statement:

```
Dim YEARDATE, MONTHDATE, DAYDATE

'Declare YEARDATE, MONTHDATE and DAYDATE as variables.


YEARDATE = "1995"

'Assign a value to the YEARDATE variable. Note that the
year is 4 digits.


MONTHDATE = "01"

'Assign a value to the MONTHDATE variable. Note that the
month is 2 digits.


DAYDATE = "31"

'Assign a value to the DAYDATE variable. Note that the day
is 2 digits.
```

```
 Report1.Formulas(2) = "Datefield= Date(" & YEARDATE & "," 
& MONTHDATE & "," & DAYDATE & ")"

'The previous 2 lines should be on 1 line. Concatenate the
Crystal Reports formula name with the word Date, an open
parenthesis, the YEARDATE variable, a comma, the MONTHDATE
variable, a comma, the DAYDATE variable, a comma, and a
closed parenthesis.


 MsgBox Report1.Formulas(2)

'Display the exact value that is passed to Crystal Reports
in a 'message box.

'This message box will display: Datefield= Date(1995,01,31)

'The Crystal report will print the Windows default (unless
'otherwise specified in Crystal): 95/1/31
```

Example 3 shows the date entered via three TextBoxes at runtime. The formatting is included while assigning the variable its value:

```
Dim YEARDATE, MONTHDATE, DAYDATE, WHOLEDATE

'Declare YEARDATE, MONTHDATE, DAYDATE, and WHOLEDATE as
variables.


 YEARDATE = YearEntry.Text

'Assign the YearEntry TextBox value to the YEARDATE
variable.


 MONTHDATE = MonthEntry.Text

'Assign the MonthEntry TextBox value to the MONTHDATE
variable.


DAYDATE = DayEntry.Text

'Assign the DayEntry TextBox value to the DAYDATE variable.


WHOLEDATE = "Date(" & YEARDATE & "," & MONTHDATE & "," &
DAYDATE & ")"

'Concatenate the word Date, an open parenthesis, the
YEARDATE variable, a comma, the MONTHDATE variable, a
comma, the DAYDATE variable, a comma, and a closed
parenthesis and assign the value to the WHOLEDATE variable.


 Report1.Formulas(2) = "Datefield=" & WHOLEDATE

'Concatenate the Crystal formula name with the WHOLEDATE
variable.


 MsgBox Report1.Formulas(2)

'Display the exact value that is passed to Crystal Reports
in a message box. This message box will display the Crystal
```

```
formula name, an equal sign, the date formatting and
whatever was entered in the YearEntry, MonthEntry, and
DayEntry TextBoxes at runtime. The Crystal report will
print the date in the Windows default (unless otherwise
specified in Crystal).
```

Example 4 uses the VB CVDate() and Format() functions to convert and format a hard-coded short or long date. The formatting is included in the Formulas() statement:

```
Dim WHOLEDATE

'Declare WHOLEDATE as a variable.


WHOLEDATE = CVDate("January 31, 1995")

'Use the VB CVDate() function to convert a long or short
date (no day of week)and assign it to the WHOLEDATE
variable. Note that the date is quoted.


 Report1.Formulas(2) = "Datefield= " & Format(WHOLEDATE, "
\D\a\t\e\(yyyy\,mm\,dd\)")

'The previous 2 lines should be on 1 line. Concatenate the
Crystal formula name with the WHOLEDATE variable and use
the VB Format() function to insert Date(, ,) as needed by
Crystal.


MsgBox Report1.Formulas(2)

'Display the exact value that is passed to Crystal Reports
in a 'message box.

'This message box will display: Datefield= Date(1995,01,31)

'The Crystal report will print the Windows default (unless
otherwise specified in Crystal): 95/1/31
```

Example 5 uses the VB CVDate() and Format() functions to convert and format a short or long date entered via a TextBox at runtime. The formatting is included while assigning the variable its value:

```
Dim WHOLEDATE

'Declare WHOLEDATE as a variable.


 WHOLEDATE= Format(CVDate(DateEntry.Text), "
\D\a\t\e\(yyyy\,mm\,dd\)")

'Assign the (long or short day, no day of week) contents of
the DateEntry TextBox to the WHOLEDATE variable and use the
VB CVDate() and Format() functions to convert it and insert
Date(, ,) as needed by Crystal.


Report1.Formulas(3) = "Datefield= " & WHOLEDATE
```

```
'Concatenate the Crystal formula name with the WHOLEDATE
variable.


 MsgBox Report1.Formulas(3)

'Display the exact value that is passed to Crystal Reports
in a message box. This message box will display the Crystal
formula name, an equal sign, the date formatting and
whatever was entered in the DateEntry TextBox at runtime.
The Crystal report will print the date in the Windows
default (unless otherwise 'specified in Crystal).
```

Contacting Crystal Decisions for Technical Support

We recommend that you refer to the product documentation and that you visit our Technical Support web site for more resources.

Self-serve Support:

http://support.crystaldecisions.com/

Email Support:

http://support.crystaldecisions.com/support/answers.asp

Telephone Support:

http://www.crystaldecisions.com/contact/support.asp