How to set custom functions in VB.NET using ByteScout Spreadsheet SDK

This tutorial will show how to set custom functions in VB.NET

The sample source code below will teach you how to set custom functions in VB.NET. ByteScout Spreadsheet SDK is the SDK component for writing, reading, modifying and calculating Excel and CSV spreadsheets. Can calculate and reculculate formulas with Excel installed. You may import or export data to and from CSV, XML, JSON. Supports export to databases, arrays, streams and you can use it to set custom functions with VB.NET.

This rich sample source code in VB.NET for ByteScout Spreadsheet SDK includes the number of functions and options you should do calling the API to set custom functions. This VB.NET sample code is all you need for your app. Just copy and paste the code, add references (if needs to) and you are all set! This basic programming language sample code for VB.NET will do the whole work for you to set custom functions.

Trial version of ByteScout Spreadsheet SDK can be downloaded for free from our website. It also includes source code samples for VB.NET and other programming languages.

FOR MORE INFORMATION AND FREE TRIAL:

Download Free Trial SDK (on-premise version)

Read more about ByteScout Spreadsheet SDK

Explore API Documentation

Get Free Training for ByteScout Spreadsheet SDK

Get Free API key for Web API

visit www.ByteScout.com

Source Code Files:

Module1.vb

```vbnet
Imports Bytescout.Spreadsheet

Module Module1

    ' This example demonstrates the calculation of custom functions.
    Sub Main()

        ' Create Spreadsheet instance

        Dim spreadsheet As New Spreadsheet()
        spreadsheet.RegistrationName = "demo"
        spreadsheet.RegistrationKey = "demo"

        ' Load document
        spreadsheet.LoadFromFile("CustomFuncExample.xlsx")

        ' Add custom formula handler
        spreadsheet.CustomFunctionsCallback = AddressOf MyFunctions

        ' Calculate the first worksheet
        Dim worksheet = spreadsheet.Workbook.Worksheets(0)
        worksheet.Calculate()

        ' Save calculated values to neighbor cells to demonstrate custom functions are
        worksheet("C2").Value = worksheet("B2").Value
        worksheet("C3").Value = worksheet("B3").Value
        worksheet("C4").Value = worksheet("B4").Value

        ' Save modified spreadsheet
        spreadsheet.SaveAs("result.xlsx")

        ' Cleanup
        spreadsheet.Dispose()

        ' Open saved spreadsheet in associated application (for demo purpose)
        Process.Start("result.xlsx")

    End Sub

    Private Function MyFunctions(ByVal funcname As String, ByVal args As Object(), ByRe

        ' Handle "CUSTOMFUNC_FACTORIAL" function
        If String.Compare(funcname, "CUSTOMFUNC_FACTORIAL", StringComparison.OrdinalIg

            handled = True

            ' Compute factorial
            If args.Length > 0 Then
                Dim value As Integer = args(0)

                If value = 0 Or value = 1 Then
                    Return 1
                End If

                Dim f As Integer = 1
```

```vbnet
                For i As Integer = 1 To value
                    f = f * i
                Next

                Return f
            End If

            Return Nothing

        End If

        ' Handle "CUSTOMFUNC_SUM" function
        If String.Compare(funcname, "CUSTOMFUNC_SUM", StringComparison.OrdinalIgnoreCas

            handled = True

            ' Compute the sum of values
            If args.Length > 0 Then

                Dim sum As Double

                For Each o As Object In args
                    sum = sum + o
                Next

                Return sum

            End If

            Return Nothing

        End If

        Return Nothing

    End Function

End Module
```

VIDEO

ON-PREMISE OFFLINE SDK