Pressu reDrop Conversions in DLL file

1 Introduction:

The conversion DLL file of unit Pressure Drop is written in C/C++ programming language style, and do not require any extra code in using this DLL file. This DLL file includes 2 functions to handle the Pressure Drop conversions as follow:

```
double LP_UnitConversionsPressureDrop_GetLeftValue (char* LeftUnit, char* RightUnit, double RightValue)  ;
double LP_UnitConversionsPressureDrop_GetRightValue(char* LeftUnit, char* RightUnit, double LeftValue )  ;
```

In Visual Basis, you can identify these functions with the code:

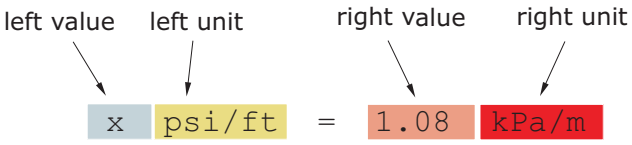
```
Declare Function LP_UnitConversionsPressureDrop_GetLeftValue Lib "LP_UnitConversionsPressureDrop.dll" _
(ByVal LeftUnit As String, ByVal RightUnit As String, ByVal RightValue As Double) As Double

Declare Function LP_UnitConversionsPressureDrop_GetRightValue Lib "LP_UnitConversionsPressureDrop.dll" _
(ByVal LeftUnit As String, ByVal RightUnit As String, ByVal RightValue As Double) As Double
```

2 Problems in Pressure Drop conversion

The two functions in DLL file are used to handle all Pressure Drop conversions in two problems.

Problem 1 The unknown value is on the **left hand side** of equation

This problem in conversion is described in the figure:



The value x is obtained by either one of two methods:

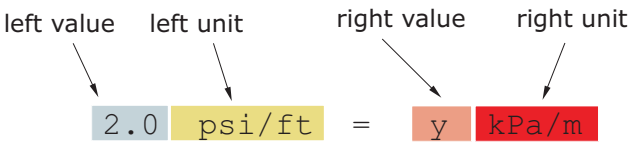
- Method A : The code is :

```
Dim x As Double
x = LP_UnitConversionsPressureDrop_GetLeftValue("PsiPerFoot", "KilopascalPerMeter", 1.08)
```

- Method B : The code is :

```
Dim LeftUnit, RightUnit As String
LeftUnit = "PsiPerFoot"
RightUnit = "KilopascalPerMeter"

Dim RightValue As Double
RightValue = 1.08

Dim x As Double
x = LP_UnitConversionsPressureDrop_GetLeftValue(LeftUnit, RightUnit, RightValue)
```

Problem 2 The unknown value is on the **right hand side** of equation
This problem in conversions is described in the figure:



The value y is obtained by either one of two methods:

- Method A : The code is :

```
Dim y As Double
y = LP_UnitConversionsPressureDrop_GetRightValue("PsiPerFoot", "KilopascalPerMeter", 2.0)
```

- Method B : The code is :

```
Dim LeftUnit, RightUnit As String
LeftUnit = "PsiPerFoot"
RightUnit = "KilopascalPerMeter"

Dim LeftValue As Double
LeftValue = 2.0

Dim y As Double
y = LP_UnitConversionsPressureDrop_GetRightValue(LeftUnit, RightUnit, LeftValue)
```

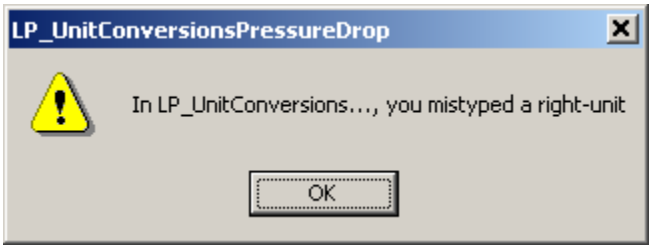
3 Unit names in Pressure Drop conversions

You can choose the unit name (case sensitive) in the following table for parameters, LeftUnit and/or RightUnit

MillimeterWaterPerMillimeter	KilopascalPerMeter
MillimeterWaterPerCentimeter	KilopascalPerKilometer
MillimeterWaterPerMeter	MegapascalPerMeter
CentimeterWaterPerMillimeter	MegapascalPerKilometer
CentimeterWaterPerCentimeter	KgfPerSquareCentimeterPerMeter
CentimeterWaterPerMeter	KgfPerSquareMeterPerMeter
MeterWaterPerMillimeter	BarPerMeter
MeterWaterPerCentimeter	PsiPerInch
MeterWaterPerMeter	PsiPerFoot
MillimeterHgPerMillimeter	PsiPerYard
MillimeterHgPerCentimeter	PsfPerInch
MillimeterHgPerMeter	PsfPerFoot
CentimeterHgPerMillimeter	PsfPerYard
CentimeterHgPerCentimeter	PsfPerMile
CentimeterHgPerMeter	InchWaterPerInch
PascalPerMillimeter	InchWaterPerFoot
PascalPerCentimeter	FootWaterPerInch
PascalPerMeter	FootWaterPerFoot

When your unit name is not in this table, the returns of functions are -1 and the error message will issue as shown in the following figures:

or



4 Tip

1. The parameters in two functions have the same order of unit name (LeftUnit is first and RightUnit is second) and the last parameter is the known value.
2. Use function LP_UnitConversionsPressureDrop_GetLeftValue(..) if your **unknown** value is on the **left hand side** of the equation.
3. Use function LP_UnitConversionsPressureDrop_GetRightValue(..) if your **unknown** value is on the **right hand side** of the equation.