© 1995-2018 Dassault Systèmes. All rights reserved.

Get and Set Sensor Example (VBA)

This example shows how to get a Measurement (dimension) sensor, resets its value to set off an alert, and fire notifications before resetting value.

- Module
- Class module

Module

```
'----------------------------------------------------

' Preconditions:

' 1. Open a part document that has a dimension

'    of 2.5 inches and a corresponding Measurement

'    (dimension) sensor that has an alert set to

'    go off if the value of the dimension is

'    reset to > 3 inches.

' 2. Select the Measurement (dimension) sensor

'    of 2.5 inches in the Sensors folder in the

'    FeatureManager design tree.

' 3. Click the Run Sub/UserForm button on the

'    toolbar in the IDE.

' 4. Click Run.

'

' Postconditions:

' 1. The Measurement (dimension) sensor alert

'    is enabled, if it wasn't previously enabled.

' 2. The value of the dimension is set to 3.5

'    inches.

' 3. The Measurement (dimension) sensor alert

'    is triggered and an event is fired

'    whenever the sensor is updated.

'----------------------------------------------------

Option Explicit


Sub main()


    Dim swApp                   As SldWorks.SldWorks
Dim swPart                  As SldWorks.PartDoc
Dim swModel                 As SldWorks.ModelDoc2
Dim swSelMgr                As SldWorks.SelectionMgr
Dim swFeat                  As SldWorks.Feature
Dim swSensor                As SldWorks.Sensor
Dim swDimSensor             As SldWorks.DimensionSensorData
Dim swDisplayDim            As SldWorks.DisplayDimension
Dim swDim                   As SldWorks.Dimension
Dim alertValue1             As Double
Dim alertvalue2             As Double
Dim sensorValue             As Double
Dim retVal                  As Long
Dim swPartEvents            As Class1

Set swApp = Application.SldWorks
Set swModel = swApp.ActiveDoc

' Event notification
Set swPart = swModel
Set swPartEvents = New Class1
Set swPartEvents.swPart = swApp.ActiveDoc

Set swSelMgr = swModel.SelectionManager

' Get the selected Measurement (dimension) sensor
' in Sensors folder in FeatureManager design tree
Set swFeat = swSelMgr.GetSelectedObject6(1, -1)
Set swSensor = swFeat.GetSpecificFeature2

' Get name of sensor
Debug.Print "Sensor name = " & swFeat.Name

' Make sure that the selected sensor is a Measurement
' (dimension) sensor (as of SOLIDWORKS 2009 SP2, only
' Measurement (dimension) sensors supported);
' if it's not, then exit the macro
If swSensor Is Nothing Then
    Debug.Print "Selected sensor is not a Measurement (dimension) sensor. Exiting macro."
```

```
        Exit Sub

    End If



    ' Get type of sensor

    Select Case swSensor.SensorType

        Case swSensorSimulation

            Debug.Print "Sensor type = Simulation"

        Case swSensorMassProperty

            Debug.Print "Sensor type = Mass Property"

        Case swSensorDimension

            Debug.Print "Sensor type = Measurement (dimension)"

        Case swSensorInterfaceDetection

            Debug.Print "Sensor type = Interference Detection"

        End Select



    ' Get whether the sensor is in an alerted state

    Debug.Print "Is an alert currently triggered for this sensor ? " & swSensor.SensorAlertState



    'Enable sensor's alert

    swSensor.SensorAlertEnabled = True

    Debug.Print "Is an alert enabled for this sensor? " & swSensor.SensorAlertEnabled



    ' Get sensor's alert state

    If swSensor.SensorAlertState Then

        Select Case swSensor.SensorAlertType

            Case swSensorAlert_GreaterThan

                Debug.Print "Sensor alert type = Greater than"

            Case swSensorAlert_LessThan

                Debug.Print "Sensor alert type = Less than"

            Case swSensorAlert_Exactly

                Debug.Print "Sensor alert type = Exactly"

            Case swSensorAlert_NotGreaterThan

                Debug.Print "Sensor alert type = Not greater than"

            Case swSensorAlert_NotLessThan

                Debug.Print "Sensor alert type = Not less than"

            Case swSensorAlert_NotExactly

                Debug.Print "Sensor alert type = Not exactly"

            Case swSensorAlert_Between
```

```
                Debug.Print "Sensor alert type = Between"
            Case swSensorAlert_NotBetween
                Debug.Print "Sensor alert type = Not between"
            Case swSensorAlert_True
                Debug.Print "Sensor alert type = True"
            Case swSensorAlert_False
                Debug.Print "Sensor alert type = False"
        End Select

        ' Get sensor's alert values
        alertValue1 = swSensor.SensorAlertValue1
        ' ISensor::SensorAlertValue2 is only valid if sensor
        ' alert type is swSensorAlert_Between
        alertvalue2 = swSensor.SensorAlertValue2
        Debug.Print "  Alert value 1 = " & alertValue1
        Debug.Print "  Alert value 2 = " & alertvalue2
    End If

    ' Set sensor to a different sensor type
    swSensor.SensorType = swSensorSimulation
    Select Case swSensor.SensorType
        Case swSensorSimulation
            Debug.Print "Set sensor type to = Simulation"
        Case swSensorMassProperty
            Debug.Print "Set sensor type to = Mass Property"
        Case swSensorDimension
            Debug.Print "Set sensor type to = Measurement (dimension)"
        Case swSensorInterfaceDetection
            Debug.Print "Set sensor type to = Interference Detection"
        End Select
    ' Update and evaluate sensor
    swSensor.UpdateSensor
    ' Set sensor type back to original type
    swSensor.SensorType = swSensorDimension
    ' Update and evaluate sensor again
    swSensor.UpdateSensor
    ' Print updated sensor type
    Select Case swSensor.SensorType
```

```
        Case swSensorSimulation

            Debug.Print "Sensor updated back to type = Simulation"

        Case swSensorMassProperty

            Debug.Print "Sensor updated back to type = Mass Property"

        Case swSensorDimension

            Debug.Print "Sensor updated back to type = Measurement (dimension)"

        Case swSensorInterfaceDetection

            Debug.Print "Sensor updated back to type = Interference Detection"

    End Select

    ' Because sensor is a Measurement (dimension) sensor,

    ' get the sensor's feature data, object, configuration name, and value

    If TypeOf swSensor Is SldWorks.DimensionSensorData Then

        Set swDimSensor = swSensor.GetSensorFeatureData


        ' Get Measurement (dimension) sensor value

        sensorValue = swDimSensor.sensorValue

        ' Convert meters to inches

        Debug.Print "Sensor value: " & (sensorValue * 39.37) & " inches"


        ' Get the actual dimension and update it

        ' to a value that sets off the alert

        Set swDisplayDim = swDimSensor.GetDisplayDimension

        Set swDim = swDisplayDim.GetDimension2(1)

        retVal = swDim.SetValue3(3.5, swSetValue_UseCurrentSetting, Nothing)

        swSensor.UpdateSensor

        swModel.ForceRebuild3 (True)


        ' Get Measurement (dimension) sensor value again

        sensorValue = swDimSensor.sensorValue

        ' Convert meters to inches

        Debug.Print "New sensor value: " & (sensorValue * 39.37) & " inches"



    End If

End Sub
```

Class module

```
Option Explicit


Public WithEvents swPart As SldWorks.PartDoc


Private Function swPart_SensorAlertPreNotify(ByVal SensorIn As Object, ByVal SensorAlertType As Long) As
Long

    MsgBox "The value of the sensor deviates from its limits."

End Function
```