ABB MEASUREMENT & ANALYTICS | APPLICATION MIGRATION MANUAL

# Spirit$^{IT}$ eXLerate

## Measurement supervisory software

## Upgrade eXLerate 2003 applications to eXLerate 2016

**Measurement made easy**

**Spirit$^{IT}$ eXLerate Application Upgrade**

## Introduction

Welcome to the exciting world of Spirit$^{IT}$ eXLerate!

Using SpiritIT eXLerate, you can create your complete real-time HMI applications.

This manual guides you through the steps of upgrading an eXLerate 2003 appliction to eXLerate 2016.

## For more information

All publications of Spirit$^{IT}$ eXLerate are available for free download from:

| | Search for |
|---|---|
| eXLerate Installation manual | IM/eXL-EN |
| eXLerate Configuration manual | CM/eXL-EN |
| Flow-X function reference manual | CM/FlowX/FR-EN |
| eXLerate release notes | RN/eXL-EN |

# Contents

# 1. Migrating to eXLerate 2016

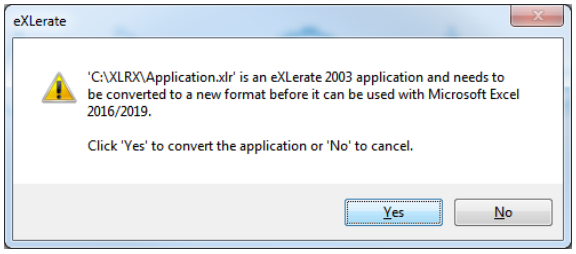
Spirit IT eXLerate 2016 can open and run Spirit IT eXLerate 2010 applications. Migrating Spirit IT eXLerate 2010 applications to eXLerate 2016 requires no conversion.

Migrating an eXLerate 2003 application to an eXLerate 2016 application requires conversion. Spirit IT eXLerate 2016 is used in combination with Microsoft Excel 365, 2019, or 2016. These Excel versions use an XML file structure. The eXLerate 2003 program was used in combination with Excel 2003, which uses a binary file format. Some steps are automatic and other are manual conversion steps.

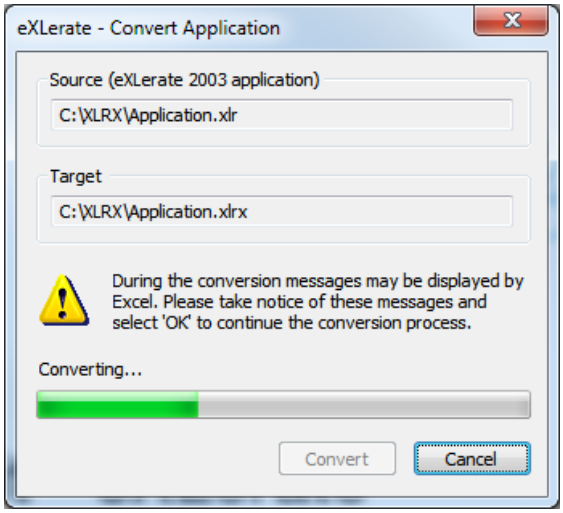
The manual steps are required as the recent Microsoft Excel versions need to convert and repair the old Excel file.

## 1.1. eXLerate conversion

When you open an eXLerate 2003 "XLR" file with Spirit IT eXLerate 2016, eXLerate will offer the possibility to convert the application.

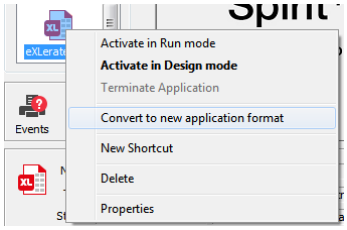



When you click "Yes" the Conversion Wizard is started. The wizard shows the source-file and the destination-file. To start the operation, click the "Convert" button.




During the conversion, Excel will restart several times. Excel might display warnings that it needs to make a change to the workbook in order to convert to the new file-format. For instance, the name “ZZ100” is valid in Excel 2003, but conflicts with a valid range in newer Excel versions. Note the warnings and change the name in your application afterwards so it doesn’t conflict with the newer Excel.

The Conversion wizard is automatically started when necessary, but can also be started manually from the Control Center short-cut. Right-click on the shortcut and select "Convert to new application format" from the menu.

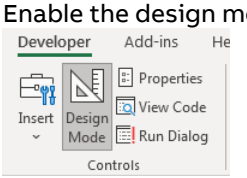



Upon success, the Shortcut in the Control Center is automatically updated to refer to the new eXLerate 2016 application file ("XLRX").If the conversion fails, perform the manual steps to convert the application.

## 1.2. Excel conversion

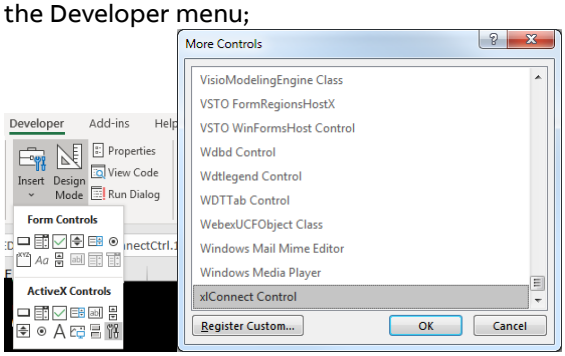
Due to some Excel version incompatibility related to ActiveX components, some manual actions may be required. If the converted application is giving errors, perform the following manual steps:

- Rename the file extension from "XLRX" into "XLSM";
- Start Excel and open the "XLSM" file;
- Excel might crash while opening and you might see an error message;
- If Excel crashed, re-open Excel without the file. Excel will perform an automatic repair;
- Enable the design mode from Developer menu;




- Remove the "xlConnect" controls ("xComms" sheet);

- Save the repaired file;
- Insert a new "xlConnect" Active X Control from the Developer menu;




- Save the file;
- Rename the file extension from "XLSM" back to "XLRX";
- Open the application with eXLerate;
- Update the application according the steps in the next sections;

# 2. Application upgrade

With the upgrade to eXLerate 2016, functionality has been depreciated and replaced with enhanced components. Once you have converted the application according the steps in the previous chapter, upgrade the application by replacing the obsolete functionality with the improved functionality.

## 2.1. Flow calculation library

With eXLerate 2003, xlMath was the library containing the flow calculation functions. In 2009 the Spirit^IT Flow-Xpert** library was as its successor. With Spirit^IT eXLerate 2016, the xlMath library will become obsolete. Therefore new applications should use the Flow-Xpert functions ("fx" prefix) instead of the xlMath functions ("xl" prefix).

The following table shows the alternative Flow-Xpert functions This table only shows the functions contained in xlMath. Spirit^IT Flow-Xpert itself contains more functions but these surpass the scope of this document.

| Function | Alternative |
|---|---|
| xlApi2540_Density | fxAPI_Dens15C_1980 |
| xlAPIDens_Table5 | fxAPI_Table5_1980 |
| xlAPIDens_Table6 | fxAPI_Table6_1980 |
| xlAPIDens_Table23 | fxAPI_Table23_1980 |
| xlAPIDens_Table24 | fxAPI_Table24_1980 |
| xlAPIDens_Table53 | fxAPI_Table53_1980 |
| xlAPIDens_Table54 | fxAPI_Table54_1980 |
| xlDensitySolartron7835 | fxSolartron_Gas_M |
| xlGpaTp25_Density | fxAPI_Dens15C_NGL_LPG |
| xlLiquidDensity | fxAPI_MPMS_11_3_3_2; fxEthylene_IUPAC_C; fxEthylene_IUPAC_M |
| xlMassFlow_AGA3 | fxAGA3_C |
| xlMassFlow_ISO5167_Orifice | fxISO5167_Orifice |
| xlMassFlow_ISO5167_ClassVent | fxISO5167_Venturi |
| xlMassFlow_ISO5167_VentNozzle | fxISO5167_VenturiNozzle |
| xlMassFlow_ISO5167_Nozzle | fxISO5167_ISA1932 |
| xlMassFlow_ISO5167_LongRadiusNozzle | fxISO5167_LongRadius |
| xlMolarMass_AGA8 | fxAGA8_C; fxAGA8_M |
| xlProp_AGA5 | fxAGA5_C |
| xlProp_AGA8 | fxAGA8_C; fxAGA8_M |
| xlProp_AGA10 | fxAGA10ex_M |
| xlProp_ISO6976_1995 | fxISO6976_1995_M |
| xlProp_ISO6976_1983 | fxISO6976_1983_M |
| xlProp_NX19 | fxNX19_M |
| xlProp_SGERG88 | fxSGERG_C; fxSGERG_M |
| xlVOS_GasUnie | fxAGA10_M |
| xlR | universal gas constant: 8.31451 J/mol K. |
| xlComponents_AGA8 | See below |
| xlLegend_AGA10 | See below |

| Function | Alternative |
|---|---|
| xlPseudoComp_AGA8 | No alternative available. |
| xlThermoProp | No alternative available. |
| xlThermoPropNames | No alternative available. |
| xlGravity | No alternative available. |

Table 1 Flow-Xpert alternatives for xlMath

The xlMath functions **xlComponents_AGA8** and **xlLegend_AGA10** are so-called meta functions. These functions returns name information about a AGA8 and AGA10 components respectively. There are no alternative for these functions in Spirit^IT eXLerate 2016. Spirit^IT Flow-Xpert contains extensive documentation on components instead.

The tables below list the content of these functions and can be used for migration purposes.

| No. | Component name | Formula |
|---|---|---|
| 1 | Methane | C1 |
| 2 | Nitrogen | N2 |
| 3 | Carbon Dioxide | CO2 |
| 4 | Ethane | C2 |
| 5 | Propane | C3 |
| 6 | Water | H2O |
| 7 | Hydrogen Sulphide | H2S |
| 8 | Hydrogen | H2 |
| 9 | Carbon Monoxide | CO |
| 10 | Oxygen | O2 |
| 11 | i-Butane | iC4 |
| 12 | n-Butane | nC4 |
| 13 | i-Pentane | iC5 |
| 14 | n-Pentane | nC5 |
| 15 | n-Hexane | nC6 |
| 16 | n-Heptane | nC7 |
| 17 | n-Octane | nC8 |
| 18 | n-Nonane | nC9 |
| 19 | n-Decane | nC10 |
| 20 | Helium | He |
| 21 | Argon | Ar |

Table 2 xlComponents_AGA8

| No. | Component | Formula | Units |
|---|---|---|---|
| 1 | Molecular weight | Mw | kg/kmol |
| 2 | Molar density at base conditions | Rhob | mol/m3 |
| 3 | Molar density at flowing conditions | Rhof | mol/m3 |
| 4 | Mass density at base conditions | Rhob | kg/m3 |
| 5 | Mass density at flowing conditions | Rhof | kg/m3 |
| 6 | Ideal gas relative density | iRD | - |
| 7 | Real gas relative density | rRD | - |
| 8 | Velocity of sound | w | m/s |
| 9 | Velocity of sound | w | ft/s |
| 10 | Compressibility at base conditions | Zb | - |
| 11 | Compressibility at flow conditions | Zf | - |
| 12 | Supercompressibility | Fpv | - |
| 13 | Ideal gas specific enthalpy | H0 | kJ/kg |

| No. | Component | Formula | Units |
|---|---|---|---|
| 14 | Real gas specific enthalpy | H | kJ/kg |
| 15 | Real gas specific entropy | S | kJ/kg/K |
| 16 | Ideal gas isobaric heat capacity | Cp0 | kJ/kg/K |
| 17 | Real gas isobaric heat capacity | Cp | kJ/kg/K |
| 18 | Real gas isochoric heat capacity | Cv | kJ/kg/K |
| 19 | Ideal gas isobaric heat capacity | Cp0 | kJ/kmol/K |
| 20 | Real gas isobaric heat capacity | Cp | kJ/kmol/K |
| 21 | Real gas isochoric heat capacity | Cv | kJ/kmol/K |
| 22 | Ratio of specific heats | Gamma | - |
| 23 | Isentropic exponent | Kappa | - |
| 24 | Critical flow factor | C* | - |
| 25 | Ideal gas specific enthalpy | H0 | kJ/kmol |
| 26 | Real gas specific enthalpy | H | kJ/kmol |
| 27 | Isentropic ideal gas critical flow factor | C*i | - |
| 28 | Isentropic real gas critical flow factor | CRi | - |
| 29 | Calculation time | t | mS |

Table 3 xlLegend_AGA10

## 2.2. Miscellaneous functions

The **xlMath** library also contained various general purpose functions. The list below shows the alternatives for these functions in Spirit^IT eXLerate 2016.

| Function | Alternative |
|---|---|
| xlTime | exNow |
| xlBitTest | exBitTest |
| xlBits2Num | exBits2Num |
| xlNum2Num | exNum2Num |
| xlNumBytes | Obsolete |
| xlCRC16 | exCRC32 |
| xlEGU | (Flow-Xpert) fxConvertUnit |
| xlFitValue | exFitValue |
| xlFitUser | exFitUser |
| xlFitLin | exFitLin |

Table 4 xlMath alternatives

xlMath contained a set of basic trending functions, intended for use in standalone Excel applications without eXLerate. These functions have become obsolete. As an alternative, the eXLerate Trending functionality should be used.

| Function |
|---|
| xlTrendValue |
| xlTrendExtremes |
| xlTrendTime |
| xlTrendAverage |

Table 5 Obsolete trend functions

## 2.3. Trend controls

Trend controls were introduced and added to Spirit^IT eXLerate in 2006. These controls supersede the old trending based on Excel Charts. With Spirit^IT eXLerate 2016, the old (legacy) trending has become obsolete. The new trend controls can be easily added to sheets and forms.

Applications that are already built with the new trend controls (i.e. uses exTrendChart controls) do not require any migration. Applications that are built using the legacy trending module based on Excel Charts, need to be migrated to the new trending module.

To migrate, follow these steps:

- Remove legacy trending worksheet(s) (Right-click the worksheet and select "Delete")
- Remove legacy trending functions from button-table
- Run the Button-wizard (this removes the legacy trending related functions from modEvents)
- Remove legacy trending 'AutoUpdate' VBA function
- Remove legacy trending worksheet functions
  - exTrendUpdate
  - exTrendData
  - exTrendDataEx
  - exTrendPenInfo
  - exZoomFactor
- Remove legacy trending VBA functions
  - exAutoMoveToEnd
  - exBigZoomIn
  - exBigZoomOut
  - exHistZoomFactor
  - exMoveBack
  - exMoveFastBack
  - exMoveFastForward
  - exMoveForward
  - exMoveToBegin
  - exMoveToEnd
  - exTrendOptions
  - exZoomIn
  - exZoomOut
- Insert and configure new trend controls on worksheet(s) or user form(s)

## 2.4. Databases

A new set of generic database functions was introduced in 2006. These functions have the prefix "exSQL" supersede the old-style "exMySQL" worksheet functions. The "exSQL" functions make it possible to communicate with the then newly introduced embedded and external databases. The old-style "exMySQL" functions have become

obsolete. The table below shows the obsolete functions and their alternatives.

| | |
|---|---|
| exMySQLConnect | exSQLConfigureDatabase |
| exMySQLCreateQuery | exSQLCreateQuery |
| exMySQLExecQuery | exSQLExecQuery |
| exMySQLExecRangeQuery | exSQLExecQuery |
| exMySQLExecRecordQuery | exSQLExecQuery |
| exMySQLLastError | exSQLLastError |
| exMySQLInfo | exSQLDiagnosticalValue |
| exMySQLPing | exSQLDiagnosticalValue |
| exMySQLStatus | exSQLDiagnosticalValue |
| exSQLExecRangeQuery | exSQLExecQuery |
| exSQLExecRecordQuery | exSQLExecQuery |

Table 6 Database functions

## 2.5. Shapes

In Excel, shapes have undergone a big metamorphosis since Excel 2003. The shapes look nicer and contain a lot more configurable properties.

### 2.5.1. Gradient fill

Latest Excel versions have more gradient capabilities than Excel 2003 and gradient settings are represented in a different way. Excel 2003 supports one color gradients. Using this feature, you could create a gradient using only one color by setting the darkness of the second gradient color. Gradients in the latest versions are represented as a set of gradient stops which can be added, removed, moved and changed.

When converting an eXLerate 2003 application, Excel automatically converts one color gradients to multi-stop gradients fill. The color of the gradient stops will be set to a fixed color - the color it had at the moment of the conversion. As SpiritIT eXLerate 2016 only animates the first color only, which is the same as in eXLerate 2003, the gradient for the second stop that appears when animating the first color (e.g. from red to blue) may be incorrect. This is because the color of the second gradient stop will not change.

To resolve this issue, set the second gradient stop to a neutral color, for instance: white, black or gray. This will ensure that when animating the first color, the gradient appears correctly.

eXLerate has no automatic gradient fill correction, but below is an example VBA code that you can use to automate the gradients on shapes.

```
'------------------------------------------------------------
--------
'Module:        modConvert
'Date:          21/11/14
'Author:        Bram Wijnen
'
'Description
'  This module can be used to automatically change the color
and
'  position of gradient stops of shapes
'  with 2 or 3 gradient stops. This is necessary for the
conversion
'  of eXLerate 2003 applications.
'  The constant lBorderColor is the color used for the
gradient stops
'  that are not controlled by animations.
'------------------------------------------------------------
--------
Option Explicit
Option Base 0

Public lBorderColor As Long
'------------------------------------------------------------
--------
' This procedure convert all shapes with 2 or 3 gradient
stops
' (also grouped shapes).
' Called after conversion to change the gray color of the
gradient
' stops that are not controlled by the animation sheet.
'------------------------------------------------------------
--------
Sub CheckShapes()
    Dim oShape As Shape
    Dim oChild As Shape
    Dim oSheet As Worksheet
    Dim vExceptions() As Variant

    On Error Resume Next

    vExceptions = Array("Freeform 998", "Group 6", "Group
17", _
        "ButtonFrame", "AlmPrt.SI.Status",
        "AlmPrt.CCR.Status", "Rectangle 646", "Rectangle 51")

    lBorderColor = RGB(120, 120, 120)

    For Each oSheet In Application.Worksheets
        For Each oShape In oSheet.Shapes
            If Not ExactMatch(vExceptions, oShape.Name) Then
                If oShape.Type = msoGroup Then
                    For Each oChild In oShape.GroupItems
                        If InStr(oShape.Name, "Meter") > 0
And _
                            InStr(oShape.Name, "Pipe") = 0
Then
                            If oChild.Type = msoGroup Then
                                oChild.Ungroup
                            Else
                                SetGradient oChild, True
                            End If
                        Else
                            SetGradient oChild
                        End If
                    Next oChild
                ElseIf oShape.Type = msoAutoShape Then
                    SetGradient oShape
                End If
            End If
        Next oShape
    Next oSheet
End Sub
'------------------------------------------------------------
--------
' This procedure converts 1 shape. It does the following:
' - Check if the shape has a gradient
' - If there are 2 gradient stops, it changes the color of
the stop
'   that is not controlled by the animation sheet to a gray
color.
' - If there are 3 gradient stops, it switches the positions
of the
'   1st and 2nd gradient stop, set the 2nd gradient stop with
the
'   original shape color and set the 1st and 3rd stop with
gray color
'------------------------------------------------------------
--------
Sub SetGradient(oShape As Shape, Optional bException As
Boolean)
    Dim lFillColor As Long

    On Error Resume Next
    With oShape.Fill
        If bException Then
            .TwoColorGradient msoGradientHorizontal, 1
            .GradientStops.Insert lBorderColor, 1
            .GradientStops(1).Position = 0.5
            .GradientStops(2).Position = 0
            .GradientStops(1).Color = RGB(255, 255, 255)
            .GradientStops(2).Color = lBorderColor
            .GradientStops(3).Color = lBorderColor
            Exit Sub
        End If

        If .GradientColorType = msoGradientColorMixed Then
Exit Sub
        lFillColor = .GradientStops(2).Color
```

```
        If .GradientStyle > 0 Then
            If .GradientStops.Count = 3 And _
                .GradientStops(2).Position > _
                .GradientStops(1).Position Then
                .GradientStops(1).Position = 0.5
                .GradientStops(1).Color = lFillColor
                .GradientStops(2).Position = 0
                .GradientStops(2).Color = lBorderColor
                .GradientStops(3).Color = lBorderColor
            ElseIf .GradientStops.Count = 3 And _
                .GradientStops(2).Position <= _
                .GradientStops(1).Position Then
                .GradientStops(2).Color = lBorderColor
                .GradientStops(3).Color = lBorderColor
            ElseIf .GradientStops.Count = 2 And _
             .GradientStops(1).Color <>
.GradientStops(2).Color Then
                If .GradientStops(1).Color = RGB(255, 255,
255) Then
                    .GradientStops(2).Color = lBorderColor
                Else
                    .Solid
                End If
            End If
        End If
    End With
End Sub
'-------------------------------------------------------------
--------
'Find an exact match in an array
'-------------------------------------------------------------
--------
Function ExactMatch(vItems() As Variant, strSearch As String)
As Boolean
    Dim aFilter() As String
    Dim lngUpper As Long
    Dim lngLower As Long
    Dim lngIndex As Long
    Dim lngCount As Long

    On Error Resume Next

    aFilter = Filter(vItems, strSearch)

    If Not ArrayEmpty(aFilter) Then
         lngUpper = UBound(aFilter)
         lngLower = LBound(aFilter)

         For lngIndex = lngLower To lngUpper
            If aFilter(lngIndex) = strSearch Then
               lngCount = lngCount + 1
            End If
         Next lngIndex

         ExactMatch = lngCount <> 0
    Else
         ExactMatch = False
    End If

End Function

'-------------------------------------------------------------
--------
'Function to check if array is empty
'Arguments:
'   vArray: Array to check
'-------------------------------------------------------------
--------
Function ArrayEmpty(vArray As Variant) As Boolean
    On Error Resume Next

    If Not IsArray(vArray) Then ArrayEmpty = True
    If UBound(vArray) < LBound(vArray) Then ArrayEmpty = True
End Function
```

## 2.5.2. 3D effects

Excel provides the possibility to add effects to shapes. As it is known that shapes with 3D effects may result in memory leaks on runtime-systems, it is recommended to remove all effects from all shapes. eXLerate has no automatic 3D effects removal, but below is an example VBA code that you can use to automate the removal of effects on shapes.

```
Option Explicit
Option Base 0
Option Compare Text
Private Enum msoShapeEffect
    msoShapeEffectNone = 0
    msoShapeShadow = (2 ^ 0)
    msoShapeReflection = (2 ^ 1)
    msoShapeGlow = (2 ^ 2)
    msoShapeSoftEdges = (2 ^ 3)
    msoShapeThreeD = (2 ^ 4)
    msoShapeEffectAll = (2 ^ 5) - 1
End Enum

' Loop through all shapes on all sheets and remove the 3D
Sub Remove3d()
    Dim oSheet As Worksheet
    Dim oShape As Shape
    For Each oSheet In ActiveWorkbook.Worksheets
        For Each oShape In oSheet.Shapes
            parseShape oShape, oSheet.Name & "!"
        Next oShape
    Next oSheet
End Sub
' Remove the 3D for the selected shape.
Sub parseShape(oShape As Shape, location As String)
    On Error Resume Next
    Select Case oShape.Type
        Case msoGroup           'For groups, parse all child
shapes
            Dim oSubShape As Shape
            For Each oSubShape In oShape.GroupItems
                parseShape oSubShape, location & oShape.Name
& "."
            Next oSubShape
            'All childs done, no need to do anything for this
group
            Exit Sub
       Case msoOLEControlObject, msoFormControl,
msoEmbeddedOLEObject
            ' These types have nothing to reset
       Case msoComment          'Only Shadow to reset
            removeShapeEffects oShape, msoShapeShadow
       Case msoChart            'Chart has no Reflection to
reset
            removeShapeEffects oShape, _
                msoShapeEffectAll - msoShapeReflection
       Case msoAutoShape, msoCallout, msoFreeform, msoLine, _
            msoPicture, msoTextBox
            removeShapeEffects oShape, msoShapeEffectAll
       Case Else
            ' The rest is not checked as it is not (commonly)
used
            removeShapeEffects oShape, msoShapeEffectAll
    End Select
End Sub

' Remove the 3D for the selected shape
Private Sub removeShapeEffects(oShape As Shape, _
                  eEffect As msoShapeEffect)
    On Error Resume Next
    If eEffect And msoShapeSoftEdges Then
        oShape.SoftEdge.Type = msoSoftEdgeTypeNone
    End If
    If eEffect And msoShapeGlow Then
        oShape.Glow.Radius = 0
    End If
    If eEffect And msoShapeReflection Then
        oShape.Reflection.Type = msoReflectionTypeNone
    End If
    If eEffect And msoShapeShadow Then
        oShape.Shadow.Visible = msoFalse
    End If
    If eEffect And msoShapeThreeD Then
        oShape.ThreeD.Visible = msoFalse

        'Resetting 3D still might leave the "Bevel" and
"Material" set
        'for some 'Button' shape styles
        'So if bevel is available, set shape style to
gradient and
        'mimic the bevel with similar color
        If oShape.ThreeD.BevelTopType <> msoBevelNone Or _
            oShape.ThreeD.BevelBottomType <> msoBevelNone
Then
            Dim lColor As Long
            lColor = oShape.Fill.ForeColor
            oShape.ShapeStyle = msoShapeStylePreset22
            With oShape.Fill.GradientStops
                .Item(1).Position = 0.1
                .Item(1).Color = lColor
                .Item(2).Color.RGB = RGB(25, 25, 25)
                .Item(2).Position = 0
                .Item(3).Color.RGB = RGB(230, 230, 230)
                .Insert .Item(1).Color, 0.9
            End With
        End If

    End If
End Sub
```

# 3. Document revisions

**Revision A November 2011**

– Initial release for eXLerate 2010 Migration Manual.

**Revision B December 2016**

– Update to eXLerate 2016.
– Update to ABB lay-out.
– New document code: IN/eXL2016-EN.

**Revision C July 2018**

– New document code: IN/eXL-EN.
– Reintroduce revisions chapter.
– Provisional support for MS Excel 2019 added.
– Windows 8 removed from software requirements.

**Revision D February 2019**

– Support for MS Excel 2019 added.
– Added references to Function Reference manual.
– Added ‘Required settings’ documentation.
– Added information for system administrators for operation system settings for kiosk mode.

**Revision E May 2020**

– Updated with the manual Excel actions required when conversion fails
– Updated the Application upgrade section
– Updated contact address and added reference to installation manual.

**Revision F December 2020**

– Layout updates

**ABB B.V.**
**Measurement & Analytics**
Achtseweg Zuid 151A =
5651 GW Eindhoven
The Netherlands
Phone: +31 40 236 9445
Mail: nl-spiritit-sales@abb.com

**ABB Malaysia Sdn Bhd.**
**Measurement & Analytics**
Lot 608, Jalan SS 13/1K
47500 Subang Jaya
Selangor Darul Ehsan, Malaysia
Phone: +60 3 5628 4888

**abb.com/midstream**

**ABB Inc.**
**Measurement & Analytics**
7051 Industrial Boulevard
Bartlesville OK 74006
United States of America
Phone: +1 800 442 3097

**ABB Limited**
**Measurement & Analytics**
Oldends Lane, Stonehouse
Gloucestershire, GL10 3TA
United Kingdom
Phone: +44 7730 019 180



IN/eXL-EN Rev. F 12.2020