***‘Mirrored Arrow’ repair reason***

***SdrObjCustomShape*** class has the ***SetMirroredY*** function.

*void SdrObjCustomShape::SetMirroredY( const sal\_Bool bMirrorY )*

*{*

 *SdrCustomShapeGeometryItem aGeometryItem( (SdrCustomShapeGeometryItem&)GetMergedItem( SDRATTR\_CUSTOMSHAPE\_GEOMETRY ) );*

 *const OUString sMirroredY( "MirroredY" );*

 *PropertyValue aPropVal;*

 *aPropVal.Name = sMirroredY;*

 *aPropVal.Value <<= bMirrorY;*

 *aGeometryItem.SetPropertyValue( aPropVal );*

 *SetMergedItem( aGeometryItem );*

*}*

When loading a document, this function is not called.

By this function is called, ***SdrObjCustomShape*** has ***MirroredY*** property, and the shape can be calculated with mirrored property.

In ***EnhancedCustomShape2d::EnhancedCustomShape2d()***, use “MirroredY” string and value.

This module is called from a parent class’s function.

And the parent function ***getTextBounds()*** uses an already created ***SdrObjCustomShape***.

Because this object has no “MirroredY” property, ***EnhancedCustomShape2d()*** function cannot parse “MirroredY” property.

awt::Rectangle SAL\_CALL EnhancedCustomShapeEngine::getTextBounds()

    throw ( RuntimeException )

{

    awt::Rectangle aTextRect;

    SdrObject\* pSdrObjCustomShape( GetSdrObjectFromXShape( mxShape ) );

    uno::Reference< document::XActionLockable > xLockable( mxShape, uno::UNO\_QUERY );

    if ( pSdrObjCustomShape && pSdrObjCustomShape->GetModel() && xLockable.is() && !xLockable->isActionLocked() )

    {

        if ( pSdrObjCustomShape )

        {

            EnhancedCustomShape2d aCustomShape2d( pSdrObjCustomShape );

            Rectangle aRect( aCustomShape2d.GetTextRect() );

            aTextRect.X = aRect.Left();

            aTextRect.Y = aRect.Top();

            aTextRect.Width = aRect.GetWidth();

            aTextRect.Height = aRect.GetHeight();

        }

    }

    return aTextRect;

}

Grab from ‘EnhancedCustomShape2d::EnhancedCustomShape2d’:

const OUString sType( "Type" );

const OUString sMirroredX( "MirroredX" );

const OUString sMirroredY( "MirroredY" );

OUString sShapeType;

SdrCustomShapeGeometryItem& rGeometryItem = (SdrCustomShapeGeometryItem&)(const SdrCustomShapeGeometryItem&)pCustomShapeObj->GetMergedItem( SDRATTR\_CUSTOMSHAPE\_GEOMETRY );

Any\* pAny = rGeometryItem.GetPropertyValueByName( sType );

if ( pAny ) {

    \*pAny >>= sShapeType;

    bOOXMLShape = ( sShapeType.startsWith("ooxml-") );

    OSL\_TRACE("shape type: %s %d", OUStringToOString( sShapeType, RTL\_TEXTENCODING\_ASCII\_US ).getStr(), bOOXMLShape);

}

eSpType = EnhancedCustomShapeTypeNames::Get( sShapeType );

pAny = rGeometryItem.GetPropertyValueByName( sMirroredX );

if ( pAny )

    \*pAny >>= bFlipH;

pAny = rGeometryItem.GetPropertyValueByName( sMirroredY );

if ( pAny )

    \*pAny >>= bFlipV;

if ( pCustomShapeObj->ISA( SdrObjCustomShape ) )    // should always be a SdrObjCustomShape, but you don't know

    nRotateAngle = (sal\_Int32)(((SdrObjCustomShape\*)pCustomShapeObj)->GetObjectRotation() \* 100.0);

else

        nRotateAngle = pCustomShapeObj->GetRotateAngle();

**This part has no value**

Meaning, there is no code that sets “MirroredY” property to ***SdrObjCustomShape***.

So, in ***SimpleShape::implConvertAndInsert()*** function, we set the ***MirroredY*** property
(in the patch):

// When flip has 'x' or 'y', the associated ShapeRect will be changed but direction change doesn't occur.

// It might occur internally in SdrObject of "sw" module, not here.

// The associated properties "PROP\_MirroredX" and "PROP\_MirroredY" have to be set here so that direction change will occur internally.

if (bFlipX || bFlipY)

{

    if (bFlipX)

        aPropertySet.setAnyProperty(PROP\_MirroredX, makeAny( bFlipX ) );

 if (bFlipY)

        aPropertySet.setAnyProperty(PROP\_MirroredY, makeAny( bFlipY ) );

    aPropertySet.setAnyProperty(PROP\_CustomShapeGeometry, makeAny( true ) );

}

And we also set this value to ***SdrObjCustomShape*** object in ***SvxCustomShape::setPropertyValue()*** :

if ( pObject && aPropertyName.equalsAsciiL( RTL\_CONSTASCII\_STRINGPARAM( "MirroredX" ) ) )

    ((SdrObjCustomShape\*)pObject)->SetMirroredX(true);

if ( pObject && aPropertyName.equalsAsciiL( RTL\_CONSTASCII\_STRINGPARAM( "MirroredY" ) ) )

    ((SdrObjCustomShape\*)pObject)->SetMirroredY(true);

You said “*You could avoid introducing a new UNO property by just adding the* ***MirroredX****/****Y*** *keys to the existing* ***CustomShapeGeometry***”.

I can understand what this means.

But ***SdrObjCustomShape*** already has a **SetMirroredY** function, so at this **setPropertyValue()** module, we directly set this property.

And we did not use SetMirroredY function twice.

This is here:

    if ( bCustomShapeGeometry )

    {

        ((SdrObjCustomShape\*)pObject)->MergeDefaultAttributes(0);

        Rectangle aRect( pObject->GetSnapRect() );

        // #i38892#

        bool bNeedsMirrorX = ((SdrObjCustomShape\*)pObject)->IsMirroredX() != bMirroredX;

        bool bNeedsMirrorY = ((SdrObjCustomShape\*)pObject)->IsMirroredY() != bMirroredY;

        boost::scoped\_ptr< SdrGluePointList > pListCopy;

        if( bNeedsMirrorX || bNeedsMirrorY )

        {

            const SdrGluePointList\* pList = pObject->GetGluePointList();

            if( pList )

                pListCopy.reset( new SdrGluePointList(\*pList) );

        }

        if ( bNeedsMirrorX )

        {

            Point aTop( ( aRect.Left() + aRect.Right() ) >> 1, aRect.Top() );

            Point aBottom( aTop.X(), aTop.Y() + 1000 );

            pObject->NbcMirror( aTop, aBottom );

            // NbcMirroring is flipping the current mirror state,

            // so we have to set the correct state again

            ((SdrObjCustomShape\*)pObject)->SetMirroredX( bMirroredX ? sal\_False : sal\_True );

        }

        if ( bNeedsMirrorY )

        {

Above is the original code before our change.

But the original code does not set the ***SetMirroredY*** value, so the part below has no sense.

Because there is no module that sets ***MirroredY*** property - we wrote the addition that sets it.

If you want declare new UNO property, we can repair this part.