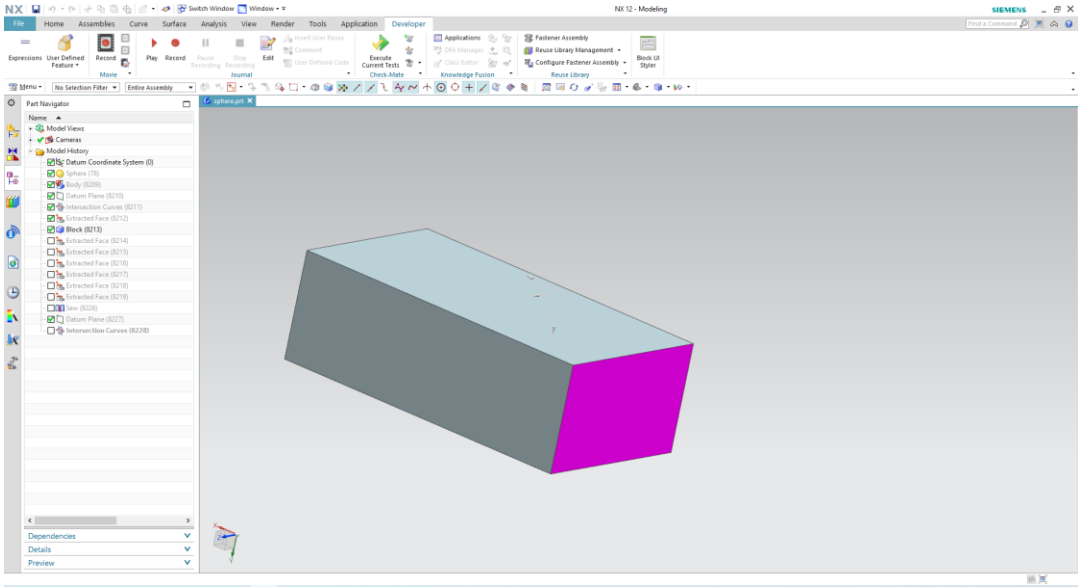
Introduction to NXOpen

Siemens NX

1. Basic structure of Journals
2. Basics
3. Code example
4. Tips
5. NXOpen.UF.Modl Package
6. Code example create DatumPlane
7. Sources

- NXOpen is an API, that can be used to write programs, customize or extend Siemens NX (much simpler and easier to understand is Snap (special licence is required))
- The „Developer“-tab can be used to record, play and edit „Journals“ (right click in the grey stripe and select „Developer“ to show the tab)
- most commonly used language is VisualBasic (a way to compile code with „Visual Studio Community“ is described in the official „NXOpen getting started“ guide)



Basic structure of Journals

Import packages

Module

- Sub
  - Code
- Functions

```
Imports System
Imports NXOpen, NXOpen.UF
imports snap,Snap.UI.Input,Snap.Create,snap.ui

Module Module1


    Sub Main()

        ' enter your code here

        '
    end sub



    function function_name()
        ' your function
        return '...
    end function


End Module
```

Basics

- Declaring variables

```
dim var_1 as integer = 1
```

- Implement lists

```
dim planes as datumplane() = {}
```

alternative:

```
dim planes = {}
```

- Expand lists (add elements)

```
Array.Resize(planes,planes.Length+1)
planes(planes.Length-1) = 'new_datumplane
```

- Try loop

```
try
    'expression, that could go wrong
catch ex1 as NXOpen.NXException
    continue for
end try
```

Code example

Let the user select a body and change the color

```
Imports System
Imports NXOpen, NXOpen.UF
imports snap,Snap.UI.Input,Snap.Create,snap.ui

Module Module1

    Dim nullNXOpen_Features_Feature As NXOpen.Features.Feature = Nothing
    Dim theSession As Session = Session.GetSession()
    Dim theUfSession As UFSession = UFSession.GetUFSession()
    Dim lw As ListingWindow = theSession.ListingWindow
    Dim workPart As Part = theSession.Parts.Work

    Sub Main()

        dim sel_body as NXopen.body = func_SelectBody("Select a body","body")
        sel_body.color = 181
        sel_body.redisplayobject

    end sub


    function func_SelectBody(prompt As String, sel_type as string)
      dim selObj as nxobject
      Dim theUI As UI = UI.GetUI
       Dim title As String = prompt
       Dim includeFeatures As Boolean = False
       Dim keepHighlighted As Boolean = False
       Dim selAction As nxopen.Selection.SelectionAction = nxopen.Selection.SelectionAction.ClearAndEnableSpecific
       Dim cursor As Point3d
       Dim scope As nxopen.Selection.SelectionScope = nxopen.Selection.SelectionScope.WorkPart
       Dim selectionMask_array(1) As nxopen.Selection.MaskTriple
        if sel_type = "body" then
            With selectionMask_array(0)
            .Type = UFConstants.UF_solid_type
            .Subtype = NXOpen.UF.UFConstants.UF_solid_body_subtype
            .SolidBodySubtype = 0
            End With
        end if

       Dim resp As nxopen.Selection.Response = theUI.SelectionManager.SelectObject( _
           prompt, title, scope, selAction, _
           includeFeatures, keepHighlighted, selectionMask_array, selobj, cursor)
      return selObj
    end function


End Module
```

Tips

- There are 3 important NX-datatypes:

  -Objects (e.g. NXOpen.line , NXOpen.block , NXOpen.datumplane)

  -NXOpen.Features („a feature consists of multiple objects“)

  -NXOpen.UF (can be used to get special infos like distances or intersection infos)

```
theUfSession.Modl.AskBoundingBoxExact(obj.tag, csys, min_corner, directions,distances)
```

- If you want to use a specific NX function, simply record a „journal“ and extract the important code lines

Convert a journal to useful code

During the convertion of the recorded journal into useful code:

change datatype: `directcast(feature1, Features.DatumPlaneFeature)`

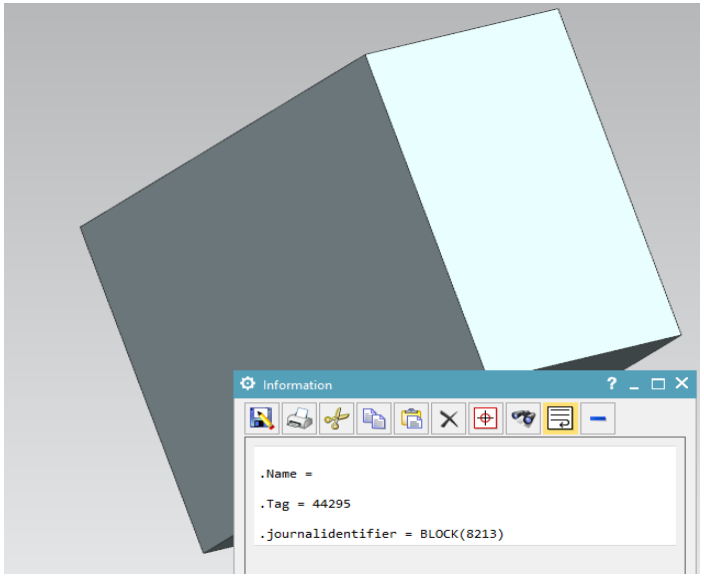
find objects: `workPart.Features.FindObject(own_feature.journalidentifier)`

Every NXOpen.features.[…] Class has 3 properties:

.Name (custom name)

.Tag (id)

.Journalidentifier (system name)



NXOpen.UF.Modl Package

- This package provides many classes the user can use in many ways. It gives the programmer the ability to compute distances, extract intersection information or create Structures

- You have to pass the OUTput paramter to the function (e.g. the „CreateBsurf“ function creates a B-Spline-Surface (type = nxopen.body) and returns the „tag“)

```
theUFSession.Modl.CreateBsurf(nv,nu-free_space,Gradu+1,Gradv+1,knoten_v,knoten_u,points_double,bsurftag,0,0)
```

Code example create DatumPlane

```
Dim datumPlaneBuilder1 As Features.DatumPlaneBuilder = workPart.Features.CreateDatumPlaneBuilder(nullNXOpen_Features_Feature)
dim plane1 As nxopen.Plane = datumPlaneBuilder1.GetPlane()
plane1.SetUpdateOption(NXOpen.SmartObject.UpdateOption.WithinModeling)
plane1.SetMethod(PlaneTypes.MethodType.Angle)
Dim geom1(1) As NXObject
geom1(0) = datumPlane
geom1(1) = line
plane1.SetGeometry(geom1)
plane1.SetExpression(angle)
plane1.SetAlternate(NXOpen.PlaneTypes.AlternateType.One)
plane1.Evaluate()
dim feature1 As Features.Feature = datumPlaneBuilder1.CommitFeature()
Dim datumPlaneFeature1 As Features.DatumPlaneFeature = directcast(feature1, Features.DatumPlaneFeature)
dim datumPlane1 as DatumPlane = datumPlaneFeature1.DatumPlane
datumPlane1.blank
datumPlaneBuilder1.Destroy()
```

Sources

Sources:

- https://docs.plm.automation.siemens.com/data_services/resources/nx/1872/nx_api/common/en_US/graphics/fileLibrary/nx/nxopen/NXOpen_Getting_Started.pdf
- https://docs.plm.automation.siemens.com/data_services/resources/nx/11/nx_api/custom/en_US/nxopen_python_ref/index.html
- https://docs.plm.automation.siemens.com/tdoc/nx/10/nx_api#uid:index_xid969099:xid961278

Communities/Forums:

- https://nxjournaling.com/
- https://community.sw.siemens.com/
- https://ww3.cad.de/
- https://www.eng-tips.com/