

Vault Data Standard Introduction

Marco Mirandola

CEO @ coolOrange



Agenda

Features

CAD dialogs
(new/update)

CAD copy/replace

CAD Functional
Designs

Vault dialogs
(new/update)

Vault tabs

Installation

Setup

Folder-structure

Update/Upgrades

Logs

Configuration

Categories,
properties,
mapping

Numbering
schemes

CAD config files

Dynamic
properties config

Customizing

Category filtering

Numbering
scheme filtering

Bread Crumb

Menus, tabs,
dialogs

Data Validation

Custom
Numbering

Cascading Drop-
downs

Debugging

Tools

PowerShell
editing

Dialog design

Resources

Official
documentation

Forum

Blogs

Learning objectives

What is Vault Data Standard (VDS) and what can it do for you/me

How to install and configure VDS

How to customize VDS and according tools

Further resources

Required skills

For the configuration

Vault, AutoCAD, Inventor basic understanding

Vault configuration

XML file editing

For the customization

Basic coding skills

Basic UI skills

What is Data Standard

It's a free Vault add-on

It standardize data entry

It's an Autodesk product

It's supported by Autodesk

It comes with the Vault installation

Supported applications: Vault, Inventor,

AutoCAD, AutoCAD Mechanical

Supported Versions: all supported Vault
versions

Data Standard Features



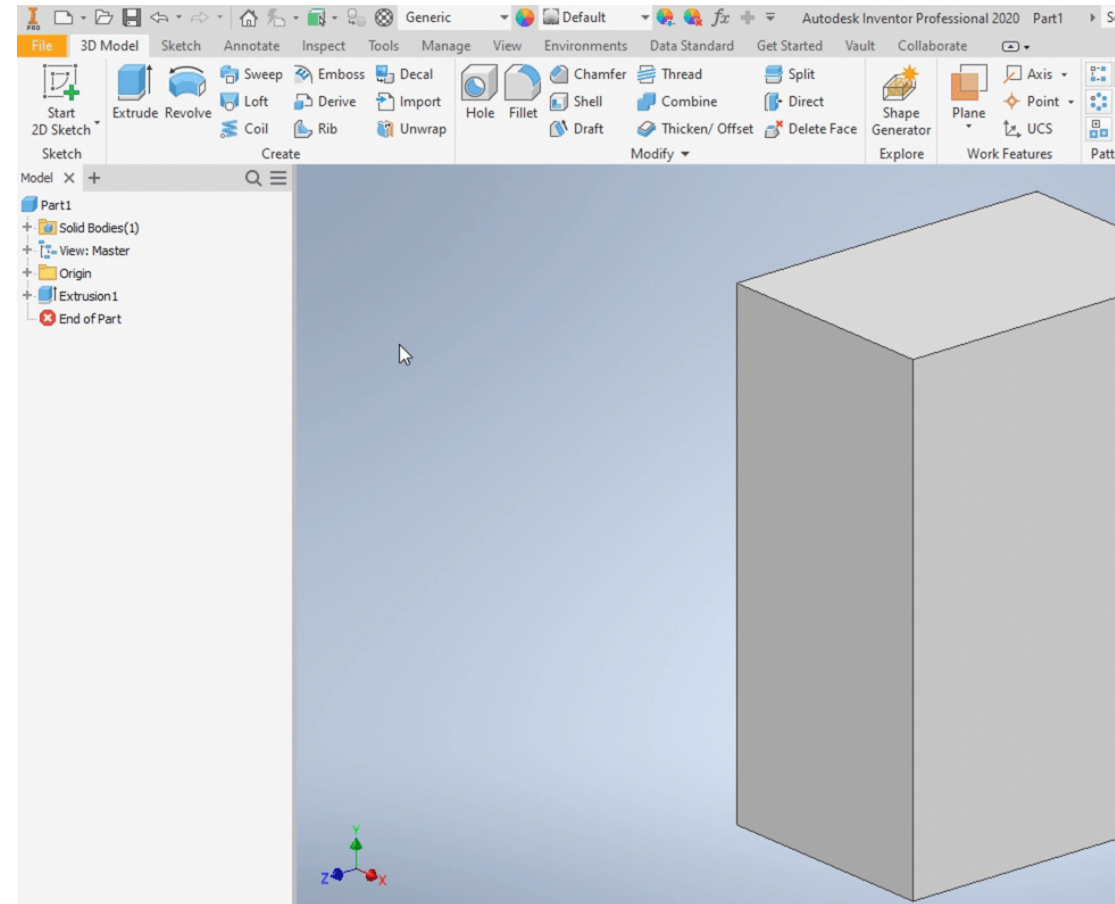
CAD initial save dialog

The Data Standard dialog appears on the initial save of an Inventor or AutoCAD file

It prompts the user to enter required values

Save the file just in the permitted folders

Drive category and numbering scheme

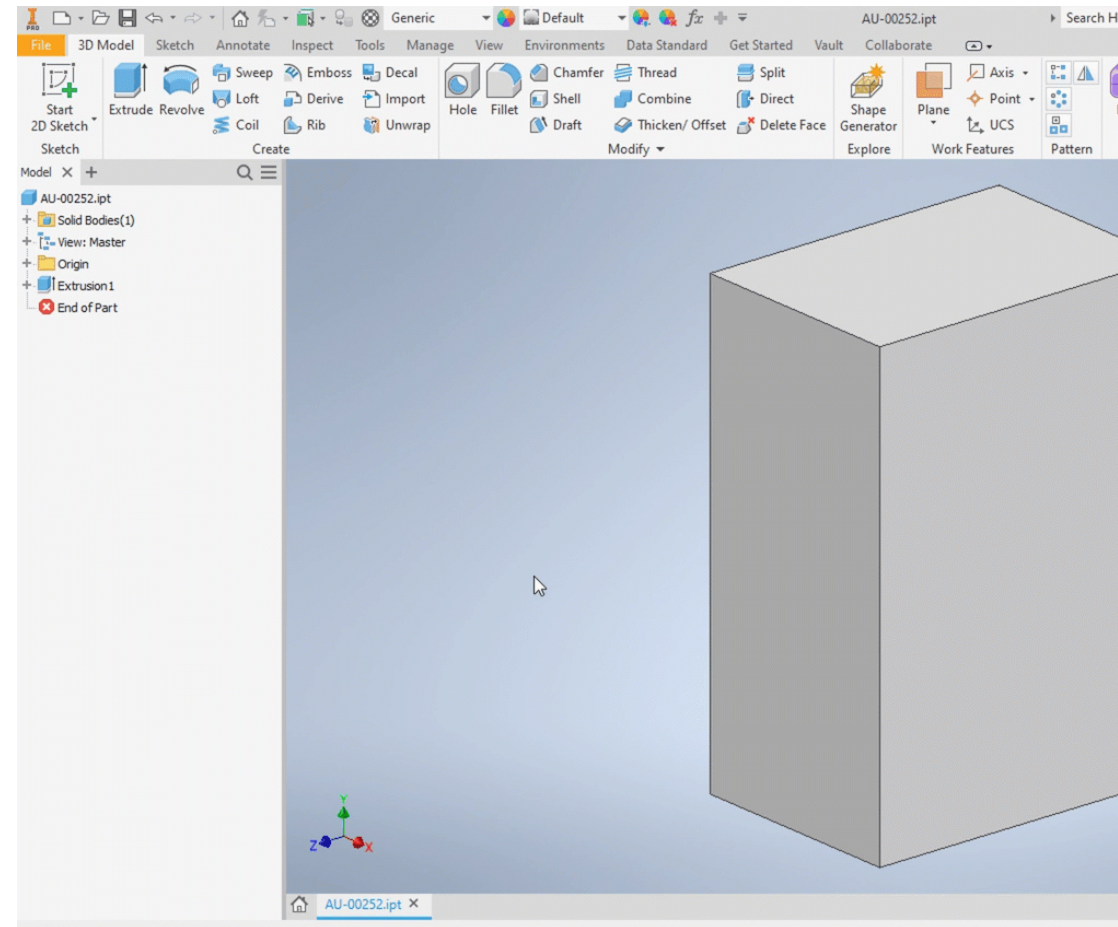


CAD edit properties dialog

After initial save, the properties can be edited at any time

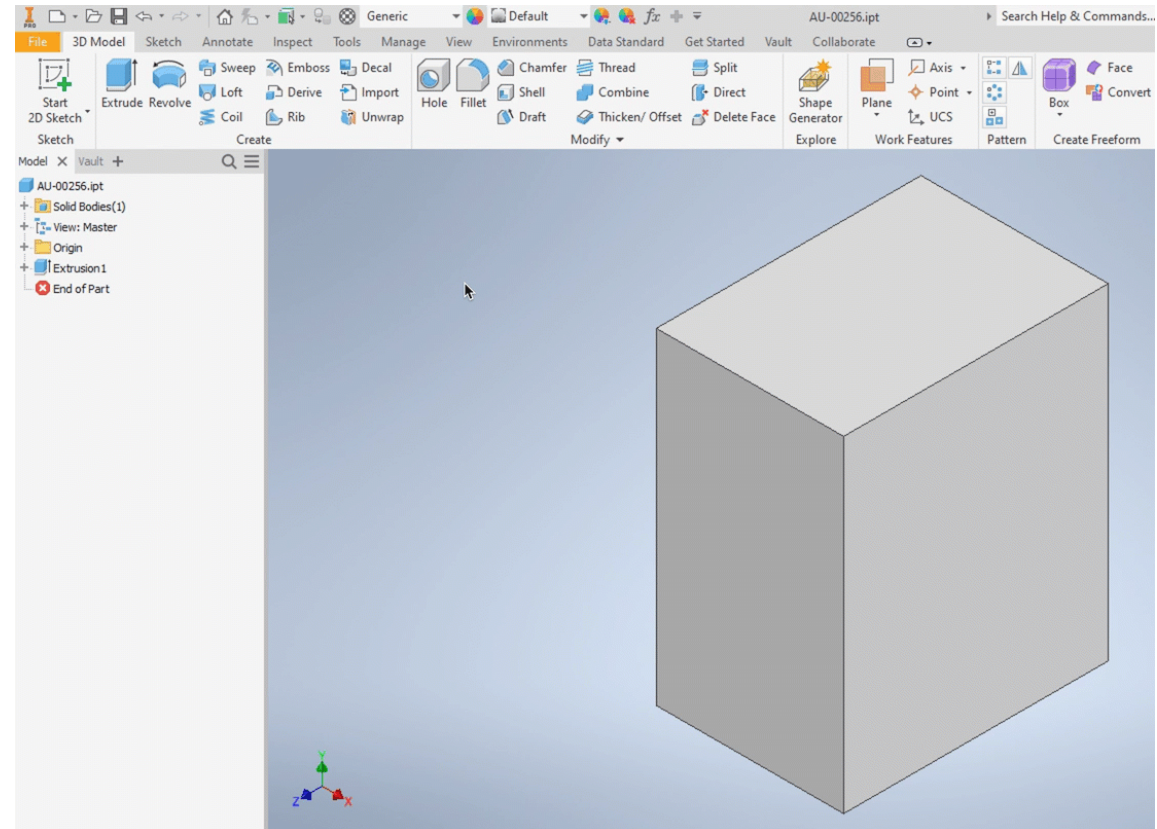
The property entry is subject to the configured property rules

The rules are validated at each save



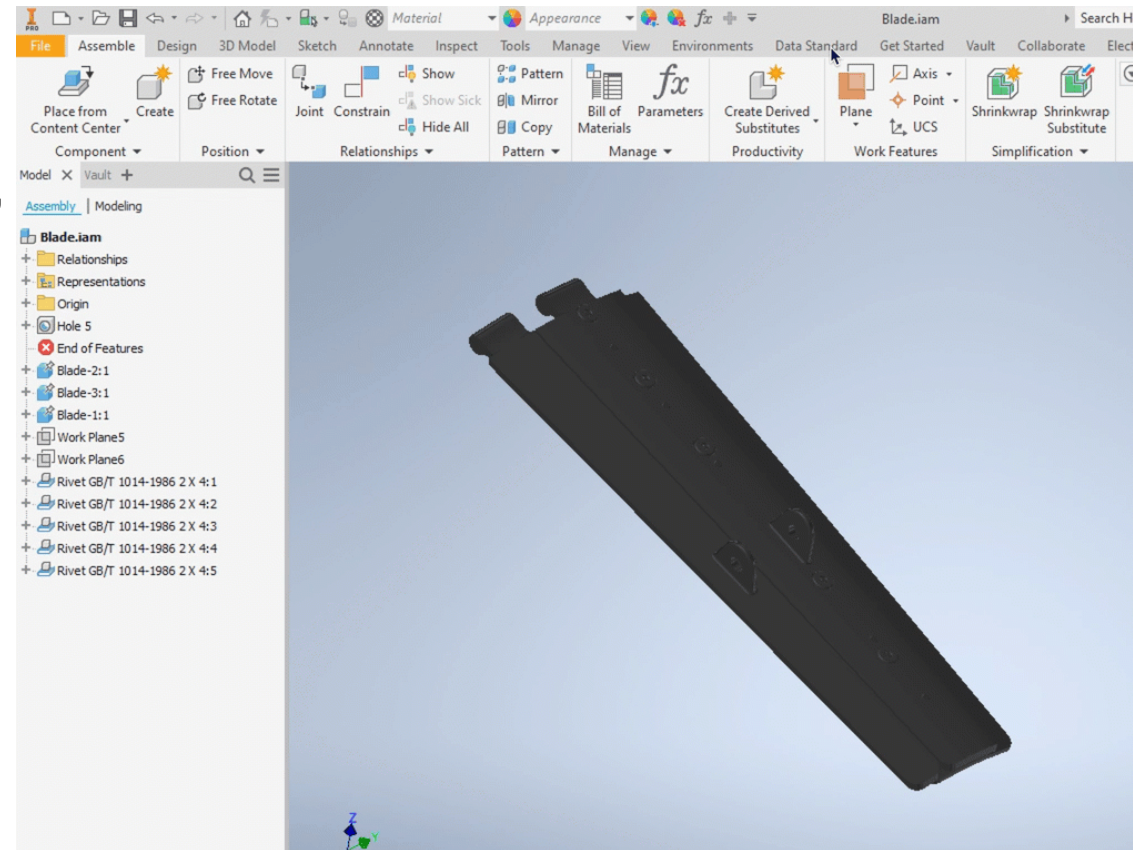
CAD copy w/o drawing

VDS offers a dedicated copy function, which takes care of related drawings



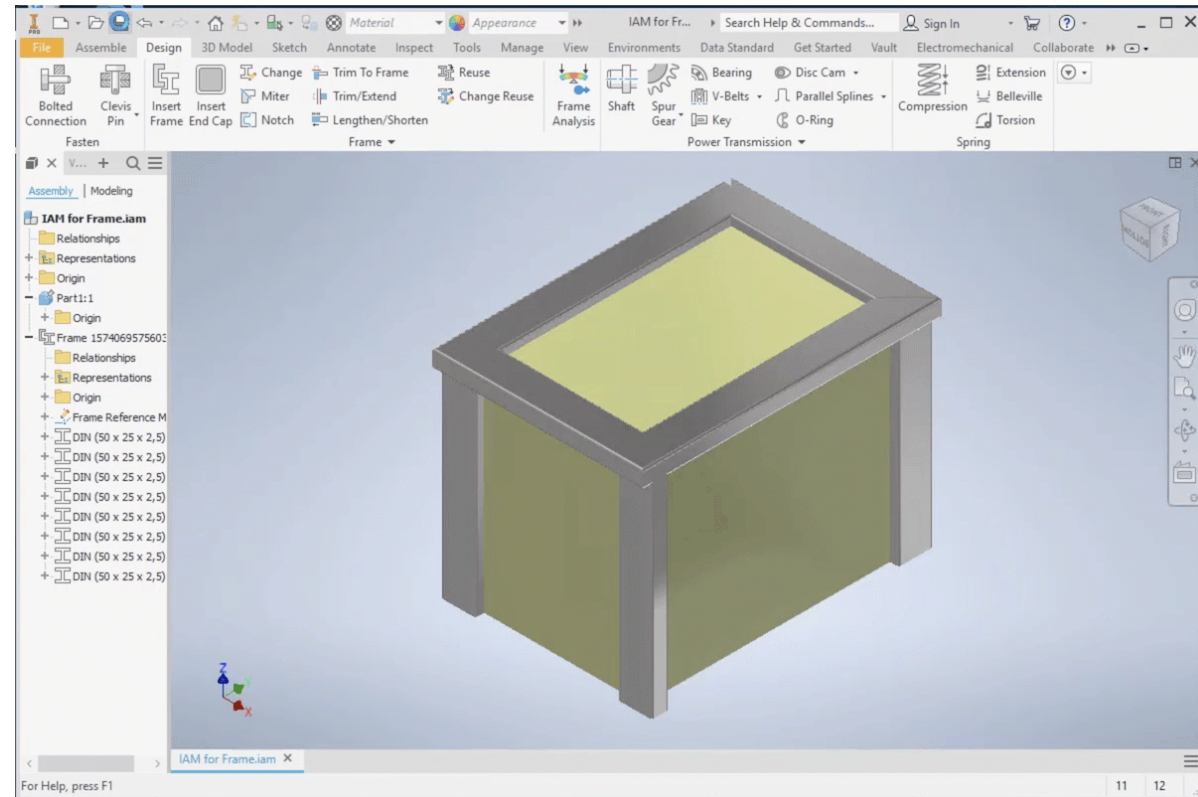
CAD replace by copy

VDS offers a dedicated replace function,
which takes care of related drawings



CAD functional designs

VDS offers a dedicated dialog for functional designs, where several components needs to be saved at once



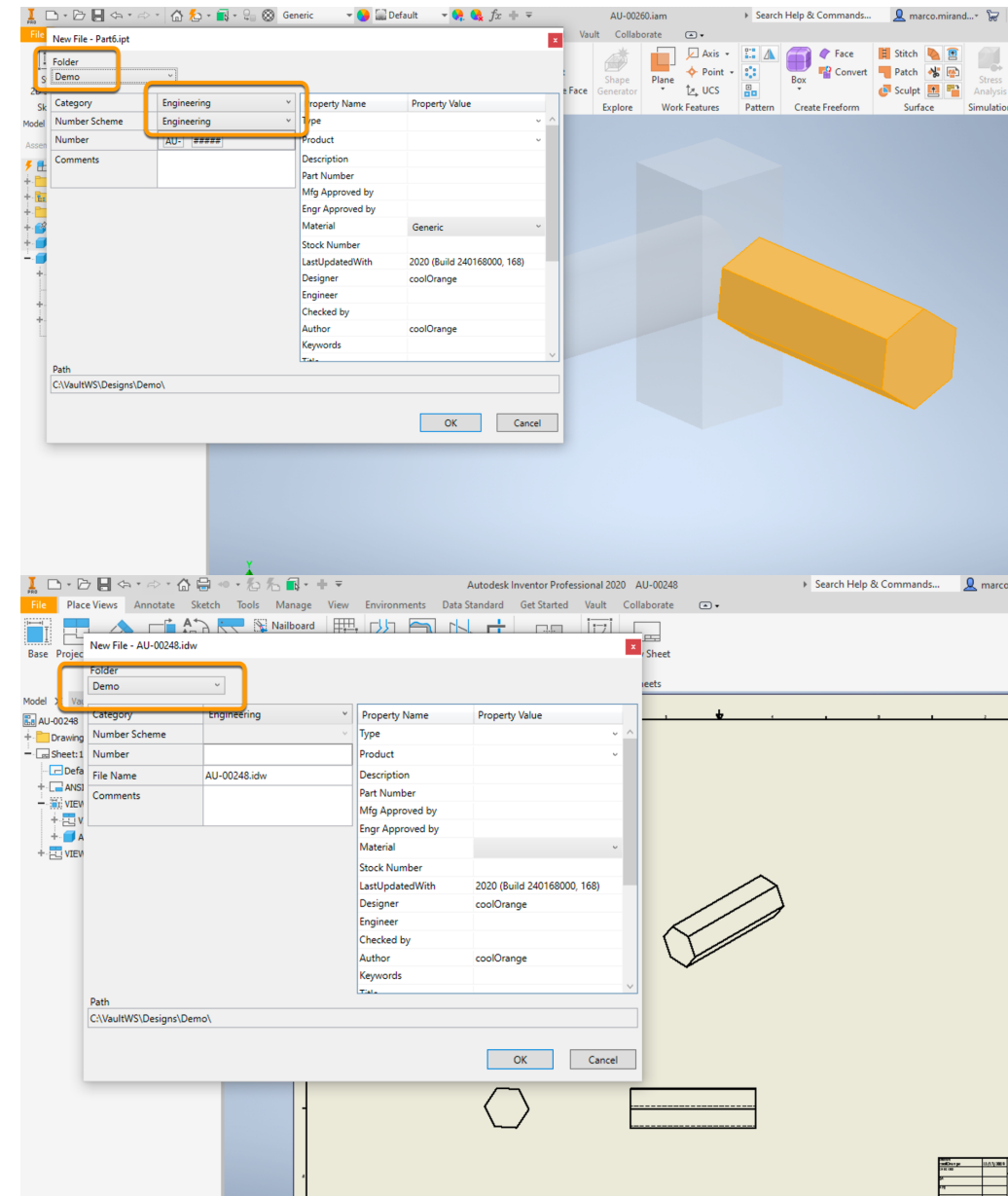
CAD VDS is smart

VDS has some built in behaviors to simplify usability

Once an assembly is saved, new added components will be suggested to be placed into the same folder

When saving a drawing, the suggested folder will be the one from the inserted model

If the numbering scheme name matches a category name, by selecting the category, the numbering scheme will follow

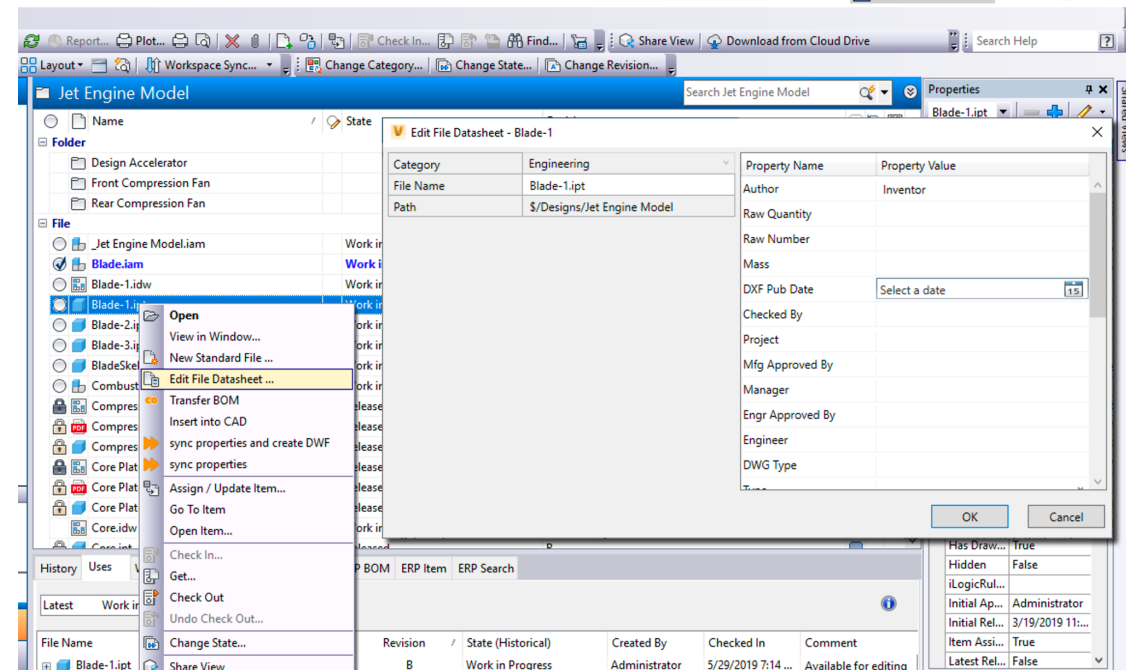
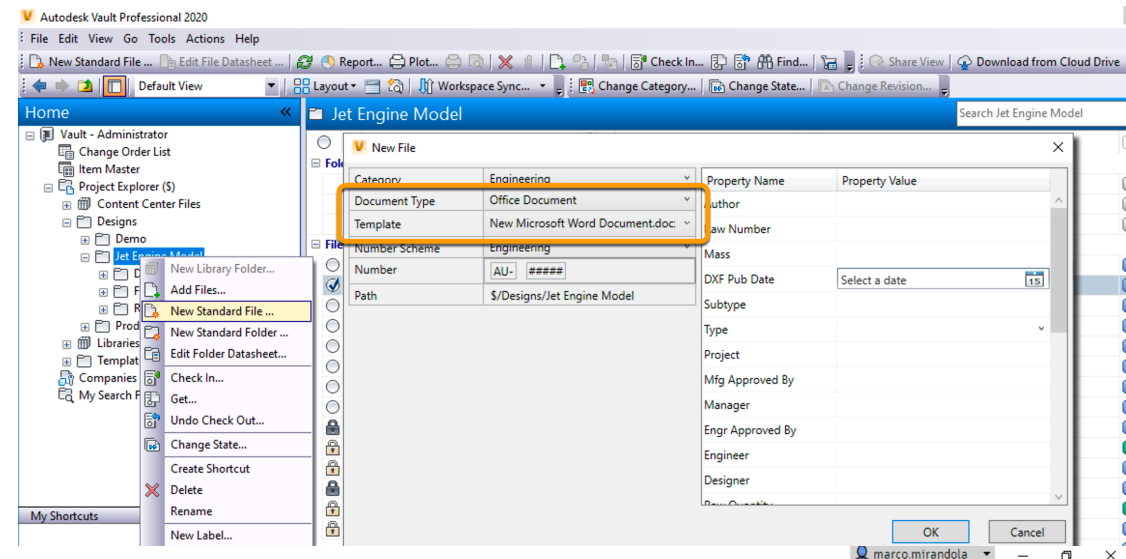


Vault file property editing

File creation dialog for Vault

Create files from a template

Edit properties



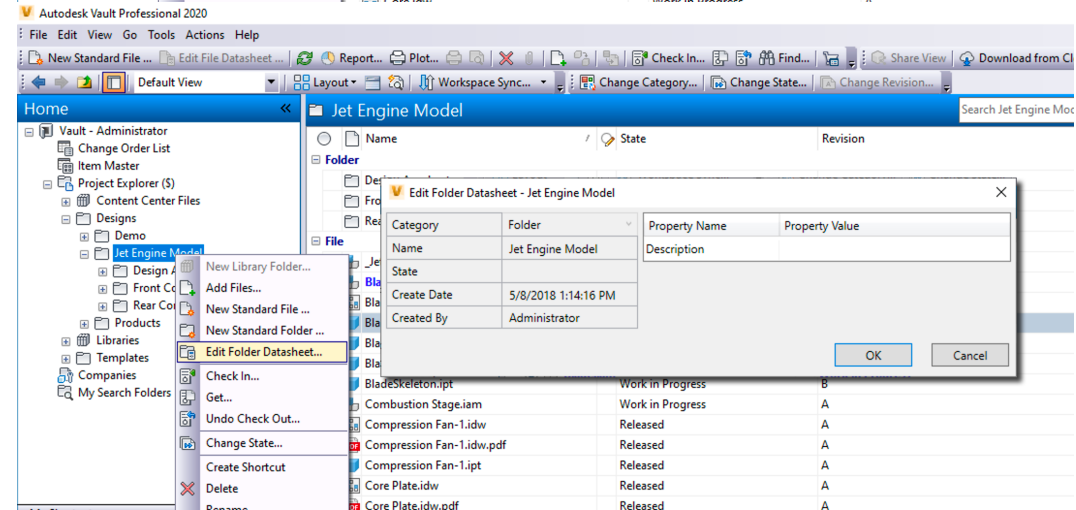
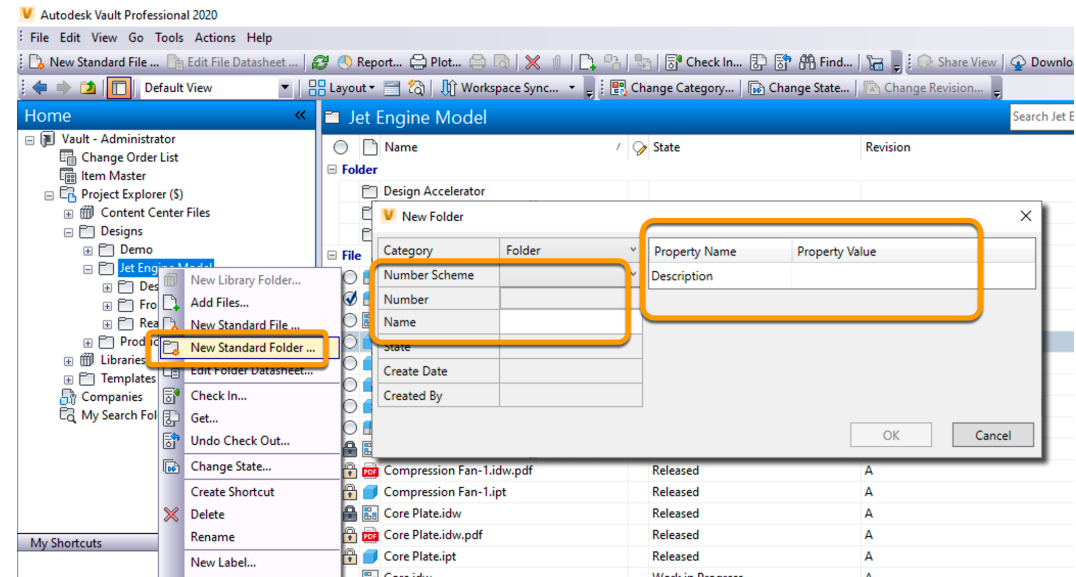
Vault folder creation

Vault folder property editing

Folder creation dialog for Vault

Numbering for folders

Edit properties



Vault tabs

Datasheet for files, folders, items, change order, custom objects

CAD BOM preview for files

Related files dialog for items

The image displays three overlapping screenshots of the SolidWorks Vault application interface, highlighting different functional tabs.

Left Screenshot (Jet Engine Model): Shows the 'Datasheet' tab for a file named 'Blade.iam'. The interface includes a tree view on the left with folders like 'Design Accelerator' and 'Front Compression Fan'. The main area shows a table with columns for 'Category', 'Folder', 'Property Name', and 'Property Value'. The 'Blade.iam' file is selected, and its state is 'Work in Progress'.

Middle Screenshot (Rear Compression Fan): Shows the 'CAD BOM' tab for a file named 'Blade.iam'. The interface displays a table with columns for 'Thumbnail', 'Position', 'Part Number', 'Quantity', 'Component Type', 'Material', 'Title', and 'Description'. The 'Blade.iam' file is selected, and its state is 'Work in Progress'.

Right Screenshot (Item Master): Shows the 'Item Master' tab for a file named 'Blade.iam'. The interface displays a table with columns for 'Number', 'Revision', and 'State'. The 'Blade.iam' file is selected, and its state is 'Work in Progress'. The 'Associated Files' and 'Datasheet' tabs are highlighted in the bottom right corner.

Data Standard Installation/Upgrade

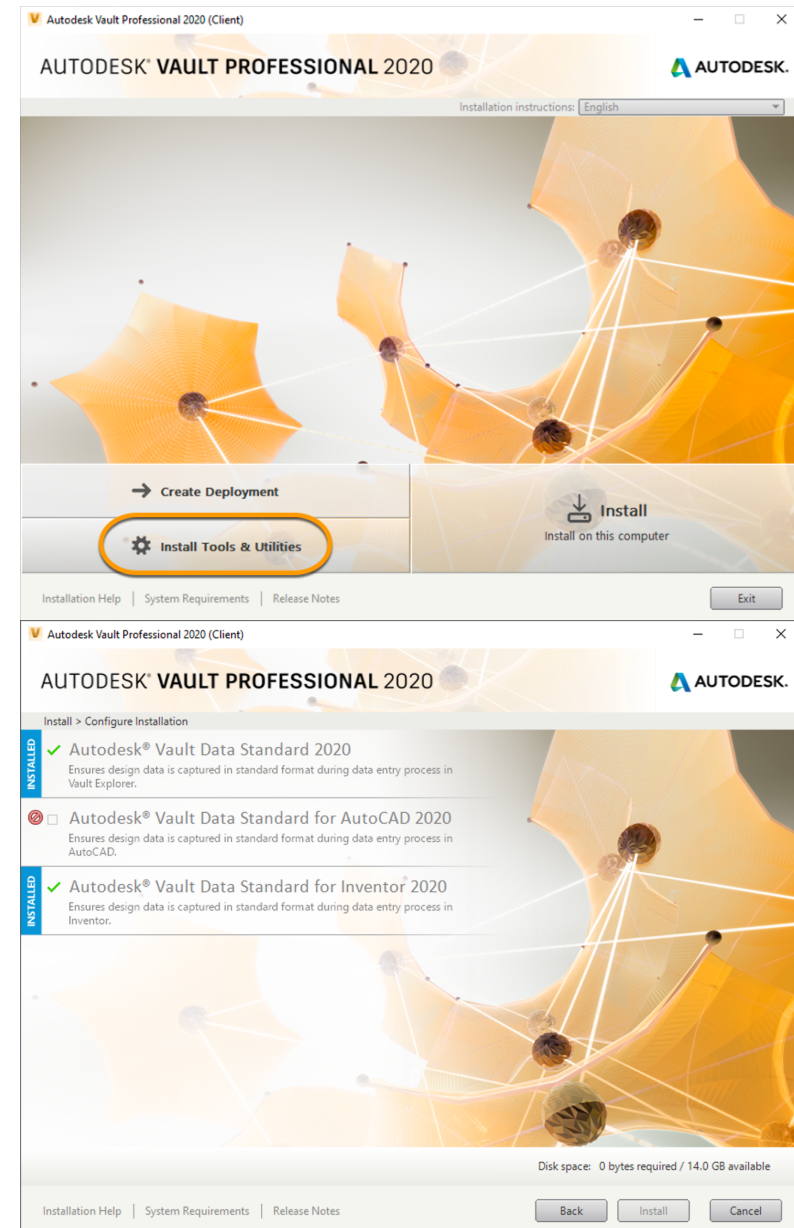


VDS installation

VDS is part of the Vault client setup since
Vault 2019

3 separate installer for Vault, AutoCAD,
Inventor

<https://knowledge.autodesk.com/support/vault-products/troubleshooting/caas/CloudHelp/cloudhelp/2019/ENU/Vault-Install/files/GUID-EF14D3BA-EC42-4388-A0A5-BCEB3BC3A411-htm.html>



VDS Folder Structure

C:\ProgramData\Autodesk\Vault
<Verion>\Extensions\DataStandard

Folder for CAD and for Vault

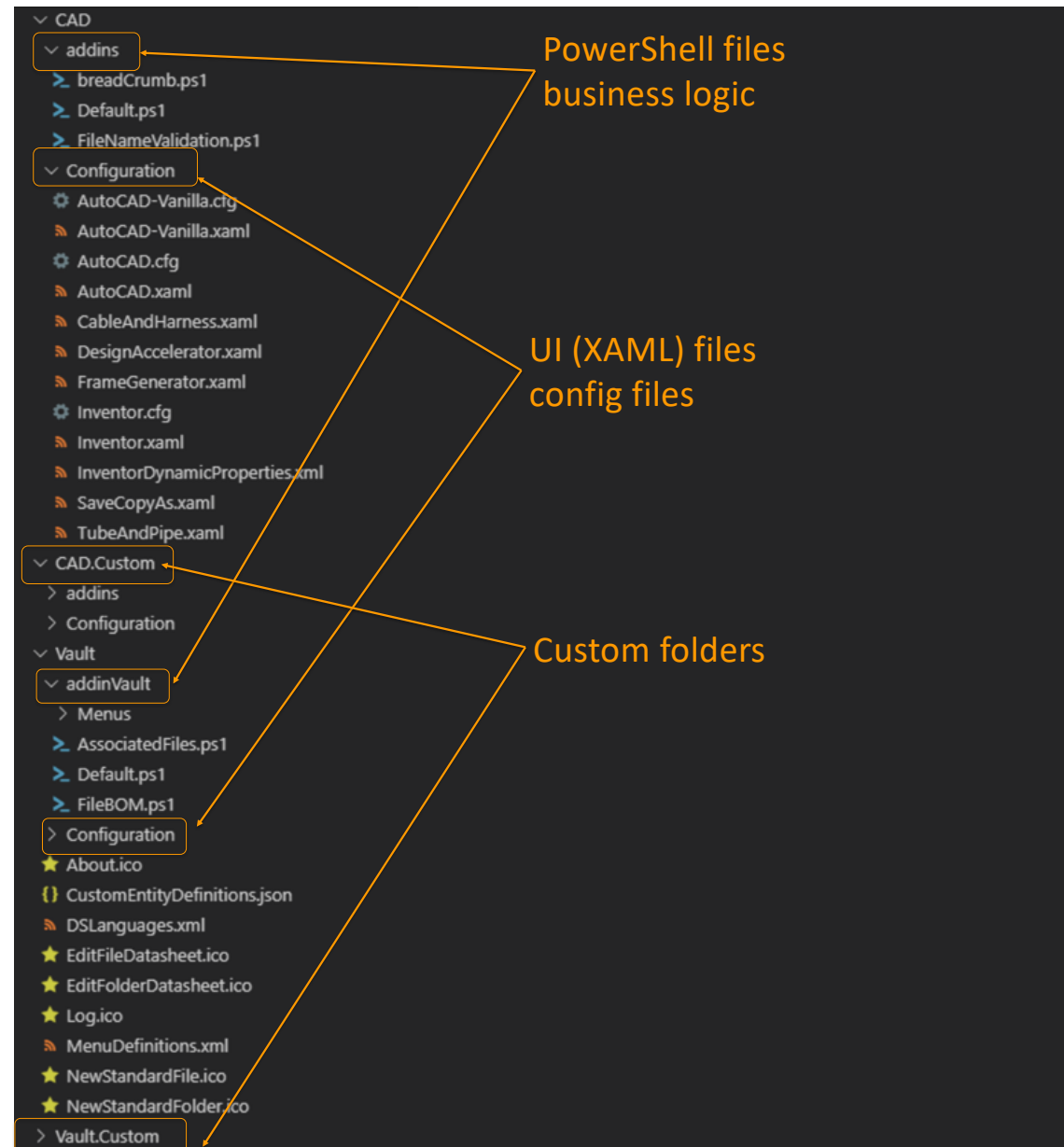
Split between default and custom

Folder contains a

code area (addins)

config area (Configuration)

Demo



VDS Logs

All logs file are under C:\temp\

Change log behavior

C:\ProgramData\Autodesk\Inventor 2019\Addins\Data
Standard\dataStandard.InvAddin.dll.log4net

C:\ProgramData\Autodesk\ApplicationPlugins\Autodesk
DataStandard
2019.bundle\Contents\dataStandard.ACadAddIn.dll.l
og4net

C:\ProgramData\Autodesk\Vault
2019\Extensions\DataStandard\dataStandard4Vault.d
ll.log4net

Local Disk (C:) > Temp

Name

DataStandardInventorlog.txt
DataStandardInventorlog.txt.1
DataStandardInventorlog.txt.2
DataStandardInventorlog.txt.3
dataStandardVaultlog.txt
dataStandardVaultlog.txt.1
dataStandardVaultlog.txt.2

DataStandardInventorlog.txt - Notepad

File Edit Format View Help

```
2019-11-13 12:28:15,353 [1] INFO dataStandard.InvAddIn.myViewAddInServer - Loading Data Standard For Inventor 24.0.49.0.
2019-11-13 12:29:05,731 [1] INFO dataStandard.InvAddIn.InventorHelpers.NameValueMapExtensions - Couldn't find FileDialog in the context.
2019-11-13 12:29:05,774 [1] INFO dataStandard.InvAddIn.InventorProperties - PropertySets : Summary Information
2019-11-13 12:29:05,777 [1] INFO dataStandard.InvAddIn.InventorProperties - Inventor property "Title="
2019-11-13 12:29:05,783 [1] INFO dataStandard.InvAddIn.InventorProperties - Inventor property "Subjects="
2019-11-13 12:29:05,783 [1] INFO dataStandard.InvAddIn.InventorProperties - Inventor property "Author=coolOrange"
2019-11-13 12:29:05,784 [1] INFO dataStandard.InvAddIn.InventorProperties - Inventor property "Keywords="
2019-11-13 12:29:05,784 [1] INFO dataStandard.InvAddIn.InventorProperties - Inventor property "Comments="
2019-11-13 12:29:05,784 [1] INFO dataStandard.InvAddIn.InventorProperties - Inventor property "Last Saved By="
2019-11-13 12:29:05,784 [1] INFO dataStandard.InvAddIn.InventorProperties - Inventor property "Revision Number="
2019-11-13 12:29:05,784 [1] INFO dataStandard.InvAddIn.InventorProperties - Inventor property "Thumbnail="
2019-11-13 12:29:05,784 [1] INFO dataStandard.InvAddIn.InventorProperties - PropertySets : Document Summary Information
2019-11-13 12:29:05,784 [1] INFO dataStandard.InvAddIn.InventorProperties - Inventor property "Category="
2019-11-13 12:29:05,784 [1] INFO dataStandard.InvAddIn.InventorProperties - Inventor property "Managers="
2019-11-13 12:29:05,785 [1] INFO dataStandard.InvAddIn.InventorProperties - Inventor property "Company="
2019-11-13 12:29:05,785 [1] INFO dataStandard.InvAddIn.InventorProperties - PropertySets : Design Tracking Properties
2019-11-13 12:29:05,785 [1] INFO dataStandard.InvAddIn.InventorProperties - Inventor property "Creation Time=11/13/2019 1:28:49 PM"
2019-11-13 12:29:05,785 [1] INFO dataStandard.InvAddIn.InventorProperties - Inventor property "Part Number="
2019-11-13 12:29:05,785 [1] INFO dataStandard.InvAddIn.InventorProperties - Inventor property "Projects="
2019-11-13 12:29:05,786 [1] INFO dataStandard.InvAddIn.InventorProperties - Inventor property "Cost Center="
2019-11-13 12:29:05,786 [1] INFO dataStandard.InvAddIn.InventorProperties - Inventor property "Checked By="
2019-11-13 12:29:05,786 [1] INFO dataStandard.InvAddIn.InventorProperties - Inventor property "Date Checked=1/1/1601 12:00:00 AM"
2019-11-13 12:29:05,786 [1] INFO dataStandard.InvAddIn.InventorProperties - Inventor property "Engr Approved By="
2019-11-13 12:29:05,786 [1] INFO dataStandard.InvAddIn.InventorProperties - Inventor property "Engr Date Approved=1/1/1601 12:00:00 AM"
2019-11-13 12:29:05,786 [1] INFO dataStandard.InvAddIn.InventorProperties - Inventor property "User Status="
2019-11-13 12:29:05,786 [1] INFO dataStandard.InvAddIn.InventorProperties - Inventor property "Material=Generic"
2019-11-13 12:29:05,788 [1] INFO dataStandard.InvAddIn.InventorProperties - Inventor property "Part Property Revision Id={6B0442EF-C08E-4D46-8D0"
2019-11-13 12:29:05,788 [1] INFO dataStandard.InvAddIn.InventorProperties - Inventor property "Catalog Web Link="
2019-11-13 12:29:05,788 [1] INFO dataStandard.InvAddIn.InventorProperties - Inventor property "Part Icons="
2019-11-13 12:29:05,788 [1] INFO dataStandard.InvAddIn.InventorProperties - Inventor property "Description="
2019-11-13 12:29:05,788 [1] INFO dataStandard.InvAddIn.InventorProperties - Inventor property "Vendor="
2019-11-13 12:29:05,788 [1] INFO dataStandard.InvAddIn.InventorProperties - Inventor property "Document SubType={4D298490-49B2-11D0-93C3-7E07066"
2019-11-13 12:29:05,788 [1] INFO dataStandard.InvAddIn.InventorProperties - Inventor property "Document SubType Name=Modeling"
2019-11-13 12:29:05,788 [1] INFO dataStandard.InvAddIn.InventorProperties - Inventor property "Proxy Refresh Date=1/1/1601 12:00:00 AM"
2019-11-13 12:29:05,788 [1] INFO dataStandard.InvAddIn.InventorProperties - Inventor property "Mfg Approved By="
2019-11-13 12:29:05,789 [1] INFO dataStandard.InvAddIn.InventorProperties - Inventor property "Mfg Date Approved=1/1/1601 12:00:00 AM"
2019-11-13 12:29:05,789 [1] INFO dataStandard.InvAddIn.InventorProperties - Inventor property "Cost=0.0000"
2019-11-13 12:29:05,789 [1] INFO dataStandard.InvAddIn.InventorProperties - Inventor property "Standards="
2019-11-13 12:29:05,789 [1] INFO dataStandard.InvAddIn.InventorProperties - Inventor property "Design Status=1"
2019-11-13 12:29:05,789 [1] INFO dataStandard.InvAddIn.InventorProperties - Inventor property "Designer=coolOrange"
2019-11-13 12:29:05,789 [1] INFO dataStandard.InvAddIn.InventorProperties - Inventor property "Engineer="
2019-11-13 12:29:05,789 [1] INFO dataStandard.InvAddIn.InventorProperties - Inventor property "Authority="
2019-11-13 12:29:05,789 [1] INFO dataStandard.InvAddIn.InventorProperties - Inventor property "Parameterized Template=False"
2019-11-13 12:29:05,789 [1] INFO dataStandard.InvAddIn.InventorProperties - Inventor property "Template Rows="
```

VDS Update

Since Vault 2018.1 all customizations shall be in the .Custom folders

Run the setup for an update

VDS Upgrade

Run the setup for an upgrade

Copy the .Custom folder to the new location

Compare your .Custom with the default folder and apply the modifications

- Menu file
- XAML
- PowerShell

Data Standard Configuration



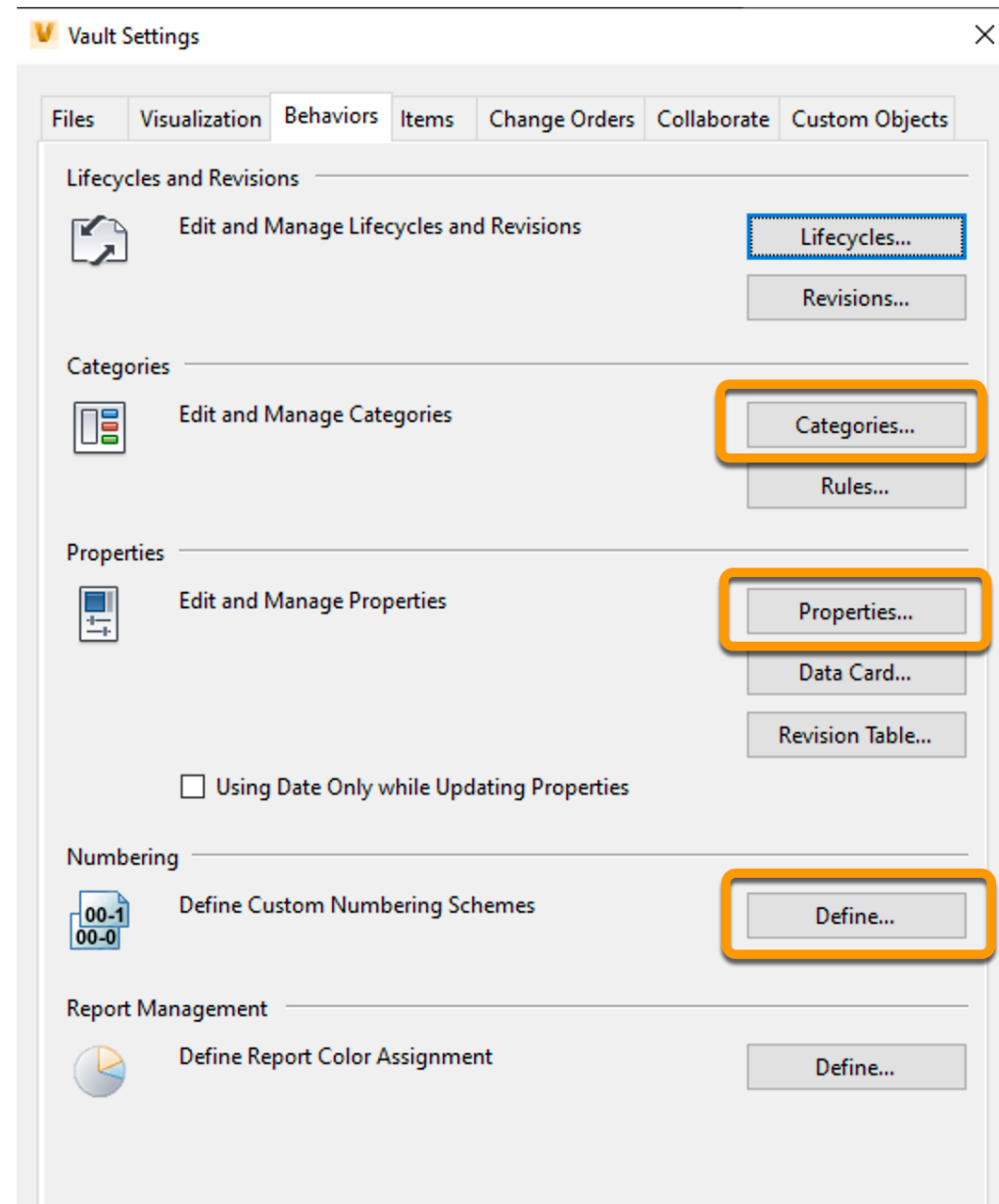
Configure your Vault!!!

Categories

Properties

Mapping (CAD)

Numbering schemes



Demo

Dynamic properties

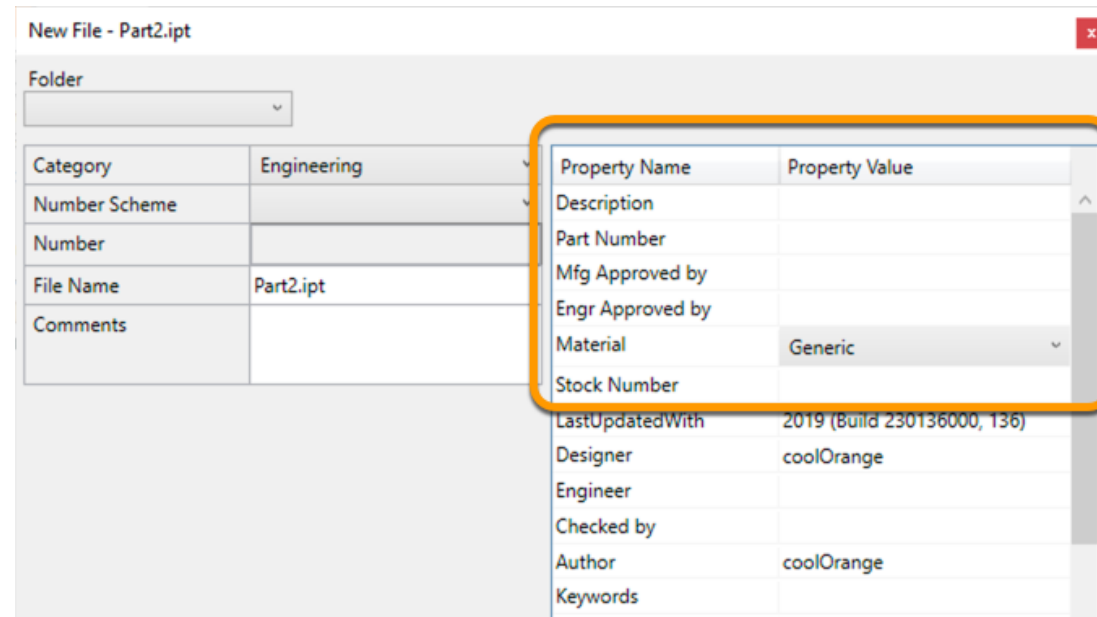
Create a file in all required categories in order to generate the following XML files

Define property order and visibility

- InventorDynamicProperties.xml
- AutoCadDynamicProperties.xml
- VaultDynamicProperties.xml

<https://blog.coolorange.com/2015/08/21/data-standard-dynamic-properties/>

Demo



Property Name	Property Value
Description	
Part Number	
Mfg Approved by	
Engr Approved by	
Material	Generic
Stock Number	
LastUpdatedWith	2019 (Build 230136000, 136)
Designer	coolOrange
Engineer	
Checked by	
Author	coolOrange
Keywords	

```
CAD.Custom > Configuration > InventorDynamicProperties.xml
1  <?xml version="1.0" encoding="utf-8"?>
2  <DynamicProperties>
3    <DynamicPropertiesCategory Category="Engineering">
4      <DynamicProperty>Description</DynamicProperty>
5      <DynamicProperty>Part Number</DynamicProperty>
6      <DynamicProperty>Mfg Approved by</DynamicProperty>
7      <DynamicProperty>Engr Approved by</DynamicProperty>
8      <DynamicProperty>Material</DynamicProperty>
9      <DynamicProperty Hidden="True">Stock</DynamicProperty>
10     <DynamicProperty>Stock Number</DynamicProperty>
11     <DynamicProperty>LastUpdatedWith</DynamicProperty>
12     <DynamicProperty>Designer</DynamicProperty>
13     <DynamicProperty>Engineer</DynamicProperty>
```

File numbering

Configure your Vault

- Create numbering schemes which match category names

New File - Part1.ipt

Folder

Category	Engineering
Number Scheme	Engineering
Number	AU- #####
Comments	

Property Name	Property Value
Raw_Number	
Raw_Quantity	0.00
Description	
Part Number	
Mfg Approved by	
Engr Approved by	
Material	Generic
Stock Number	
LastUpdatedWith	2019 (Build 230136000, 136)
Designer	coolOrange
Engineer	
Checked by	
Author	coolOrange

Demo

CAD config files

Copy the config file into the
CAD.Custom\Configuration folder

Configure your CAD VDS

- Supported file types (SupportedFileTypes)
- Skip special files (SkipForProperties)
- Check-in after save (ShowCheckInDialog)

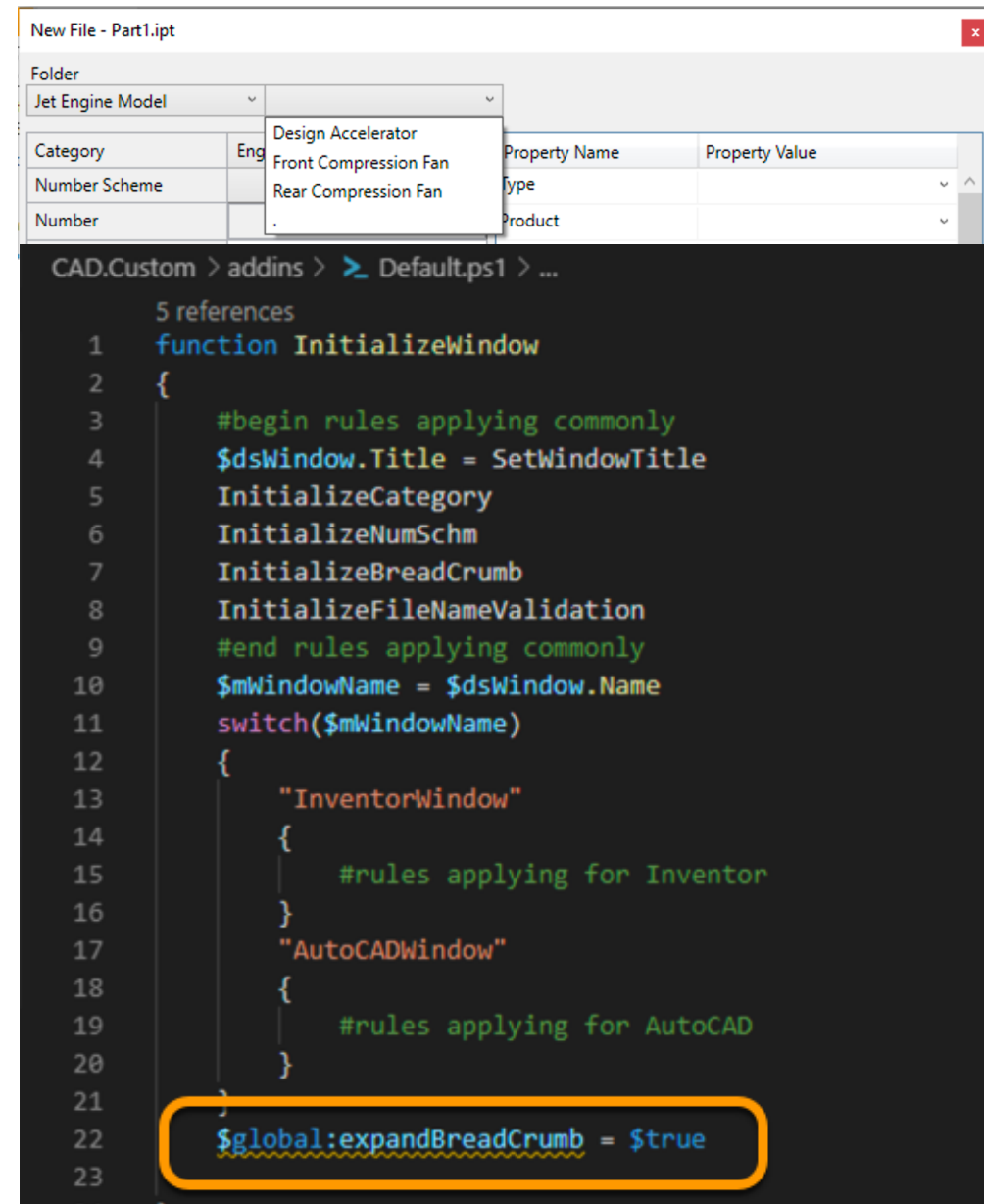
```
> Extensions > Datastandard > CAD.Custom > Configuration > Inventor.cfg
1  <?xml version="1.0" encoding="utf-8"?>
2  <Configuration>
3    <PathDefinition>{Workspace}\{Prop[Folder].Value}</PathDefinition>
4    <FileNameDefinition>{Prop[DocNumber].Value}</FileNameDefinition>
5  > <PropertyDefinitions>...
12 </PropertyDefinitions>
13 <SupportedFileTypes>IDW,IAM,IPT,IPN,DWG</SupportedFileTypes>
14 <ShowCheckInDialog>True</ShowCheckInDialog>
15 <SkipForProperties></SkipForProperties>
16 </Configuration>
```

```
> Extensions > Datastandard > CAD.Custom > Configuration > AutoCAD.cfg
1  <?xml version="1.0" encoding="utf-8"?>
2  <Configuration>
3    <PathDefinition>{Workspace}\{Prop[Folder].Value}</PathDefinition>
4    <FileNameDefinition>{Prop[GEN-TITLE-DWG].Value}</FileNameDefinition>
5  > <PropertyDefinitions>...
11 </PropertyDefinitions>
12 <ShowCheckInDialog>True</ShowCheckInDialog>
13 <TitleBlock>ISO_TITLEA,ISO_TITLEB,DIN_TITLE</TitleBlock>
14 </Configuration>
```


CAD bread crumb behavior

Configure whether the bread crumb shall be automatically expanded or not

1. Create a new Default.ps1 in the CAD.Custom\addins folder
2. Copy the function InitializeWindow from the standard Default.ps1 to the custom
3. Modify the variable `$global:expandBreadCrumb` to either `$true` or `$false`



The screenshot shows a CAD software interface with a custom Default.ps1 file open. The file path is `CAD.Custom > addins > Default.ps1 > ...`. The function `InitializeWindow` is defined with 5 references. The variable `$global:expandBreadCrumb` is set to `$true`, which is highlighted with a yellow box. The function also includes rules for Inventor and AutoCAD windows.

```
5 references
1 function InitializeWindow
2 {
3     #begin rules applying commonly
4     $dsWindow.Title = SetWindowTitle
5     InitializeCategory
6     InitializeNumSchm
7     InitializeBreadCrumb
8     InitializeFileNameValidation
9     #end rules applying commonly
10    $mWindowName = $dsWindow.Name
11    switch($mWindowName)
12    {
13        "InventorWindow"
14        {
15            #rules applying for Inventor
16        }
17        "AutoCADWindow"
18        {
19            #rules applying for AutoCAD
20        }
21    }
22    $global:expandBreadCrumb = $true
23 }
```

Vault templates configuration

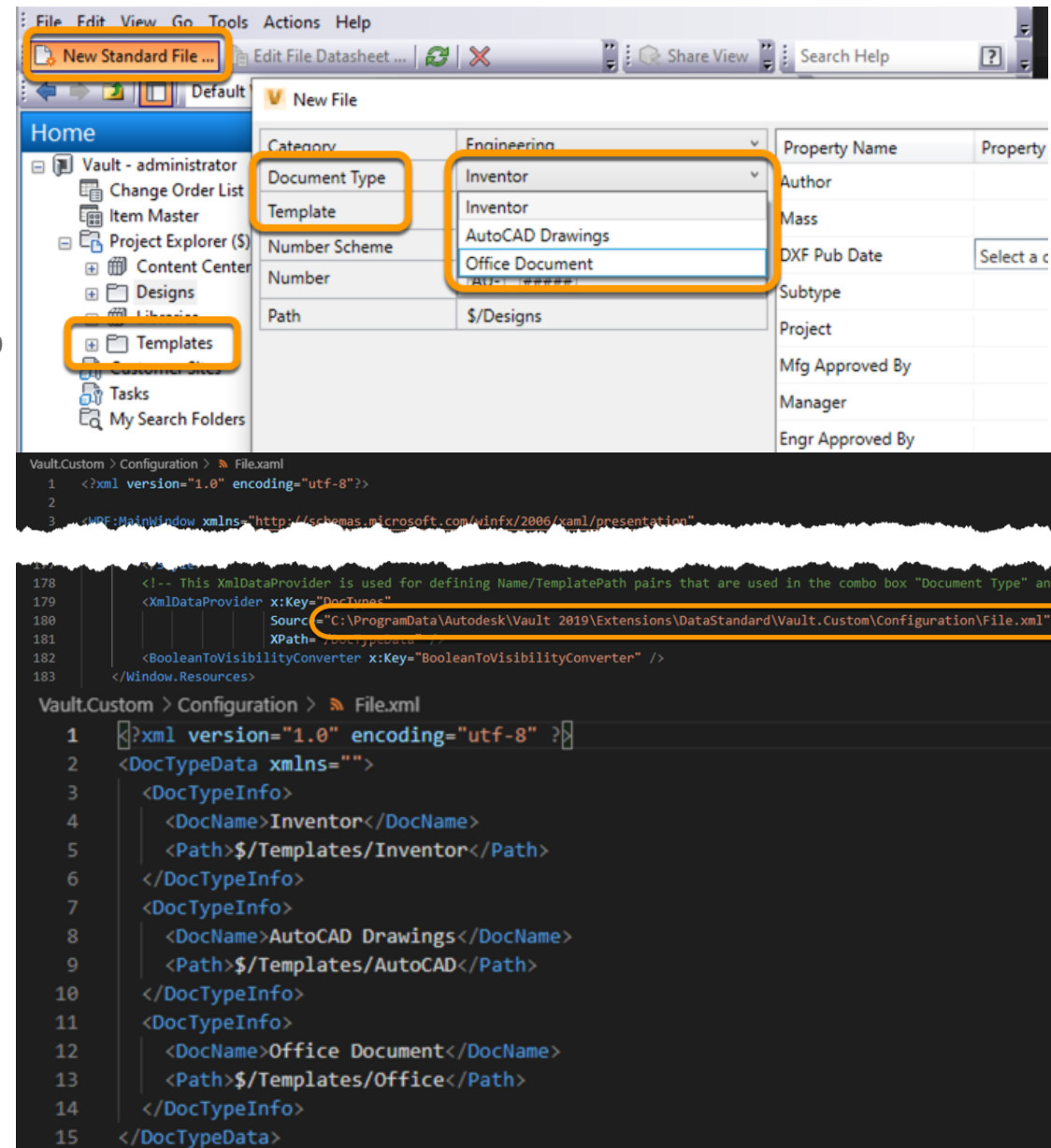
Copy the file.xml and file.xaml into the .Custom folder

Modify the path to the file.xml in the file.xml

Configure the file.xml

Create the folders inside Vault and add the template files

<http://help.autodesk.com/view/VAULT/2020/ENU/?guid=GUID-DBDC927B-269D-4552-AECF-37CEDE0AE326>



Custom objects

Create a custom object

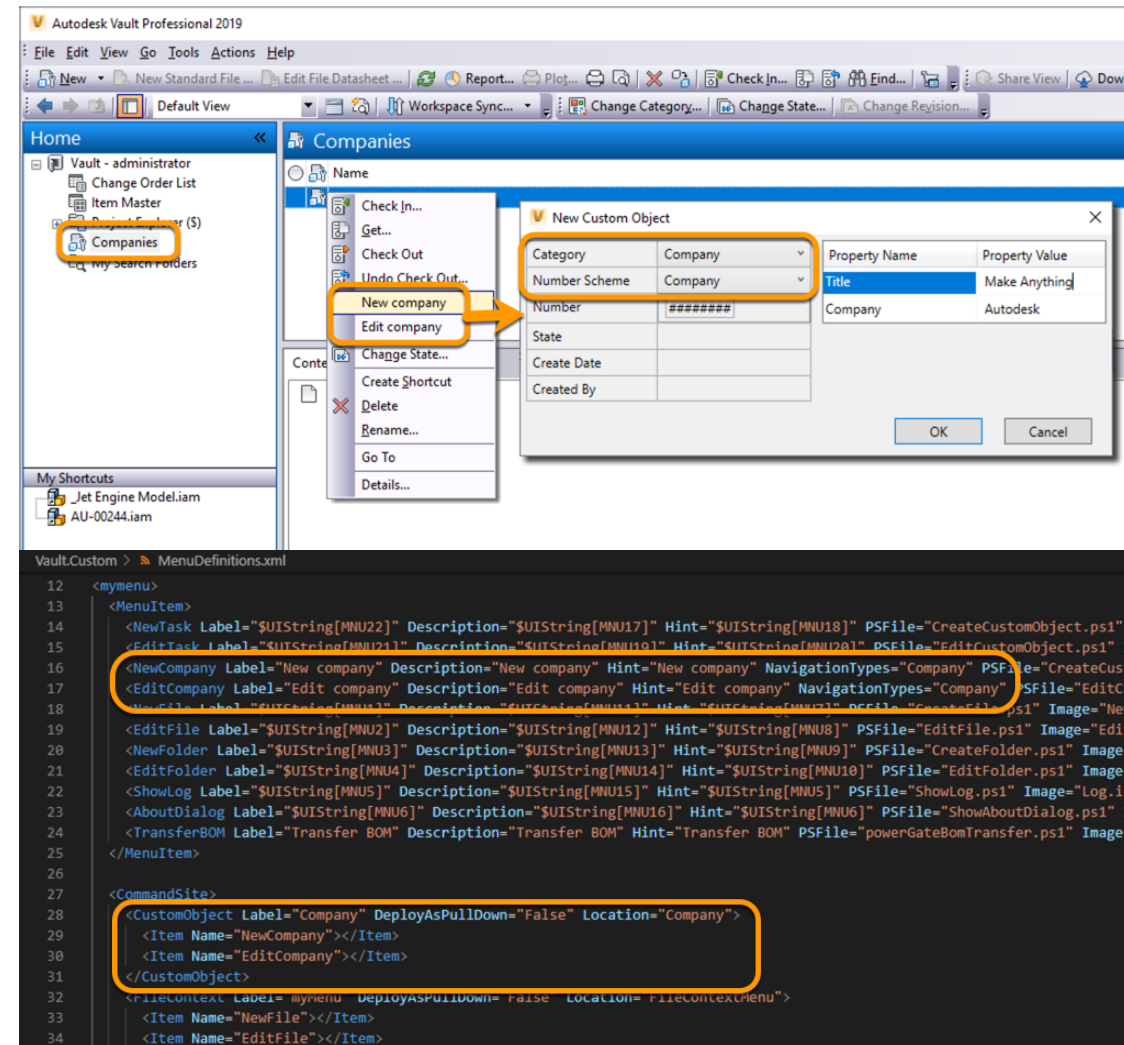
Define a category with same name

Create a numbering scheme with same name

Assign properties to the category

Create in the MenuDefinition.XML
new menu item entries

<http://help.autodesk.com/view/VAULT/2020/ENU/?guid=GUID-E853B96F-4B6C-425A-BA32-DE858E1AA6DF>



Data Standard Customization



Default category

Define default category

1. Copy the function

InitializeCategory into a
custom PowerShell script

2. Change the business logic

Demo

New File - Part1.ipt

Folder	
Category	Engineering
Number Scheme	Engineering
Number	Office
File Name	Part1.ipt
Comments	

Property Name	Property Value
Rw_Number	
Rw_Quantity	0.00
Description	
Part Number	
Mfg Approved by	
Engr Approved by	
Material	Generic
Stock Number	
LastUpdatedWith	2019 (Build 230136000, 136)
Designer	coolOrange

```
CAD.Custom > addins > ➤ Default.ps1 > GetNumSchms {}  
function InitializeCategory()  
{  
    if ($Prop["_CreateMode"].Value)  
    {  
        if (-not $Prop["_SaveCopyAsMode"].Value)  
        {  
            if($prop["_FileExt"].Value -eq '.idw') {  
                $Prop["_Category"].Value = "Drawings"  
            }  
            else{  
                $Prop["_Category"].Value = "Engineering"  
            }  
        }  
    }  
}
```

Custom category list

Filter the list of categories

1. Copy the function `GetCategories` into a custom PowerShell script
2. Change the business logic

The screenshot displays a CAD application window titled "New File - Part1.ipt". The interface includes a "Folder" dropdown, a "Category" dropdown menu, and a "Number Scheme" dropdown. The "Category" dropdown is open, showing a list of categories: "Engineering" (selected), "Engineering", and "Office". The "Number Scheme" dropdown is also open, showing "Engineering" and "Office". The "File Name" field is set to "Part1.ipt". The "Comments" field is empty. To the right of the dropdowns is a table with two columns: "Property Name" and "Property Value". The table contains the following data:

Property Name	Property Value
Rw_Number	
Rw_Quantity	0.00
Description	
Part Number	
Mfg Approved by	
Engr Approved by	
Material	Generic
Stock Number	
LastUpdatedWith	2019 (Build 230136000, 136)
Designer	coolOrange
Engineer	

Below the CAD application window, a PowerShell script is shown in a dark-themed editor. The script is titled "CAD.Custom > addins > Default.ps1 > ...". The script defines a function `GetCategories` that filters a list of categories based on the file extension. The script is as follows:

```
1 function GetCategories
2 {
3     $categories = $vault.CategoryService.GetCategoriesByEntityClassId("FILE", $true)
4     $allowedCategories = @('Engineering', 'Office')
5     if($prop["_FileExt"].Value -eq '.idw') {
6         $allowedCategories = @('Drawings')
7     }
8     $myCategoryList = @()
9     foreach($category in $categories){
10         if($allowedCategories -contains $category.Name){
11             $myCategoryList += $category
12         }
13     }
14     return $myCategoryList
15 }
16
17
18
19
```

Custom numbering scheme list

Filter the list of numbering schemes

1. Copy the function GetNumSchms into a custom PowerShell script
2. Change the business logic

The screenshot shows a CAD application window titled "New File - Part1.ipt". It features a "Folder" dropdown and a table with columns "Category", "Number Scheme", "Number", "File Name", and "Comments". The "Number Scheme" column is highlighted with an orange box, showing a list of schemes: "Office" and "Engineering". The "Engineering" scheme is selected and highlighted with a blue dashed border.

Below the table, there is a PowerShell script editor showing the function `GetNumSchms`. The script is designed to filter numbering schemes based on the selected category. The following code segments are highlighted with orange boxes:

```
function GetNumSchms
{
    $specialFiles = @(".DWG", ".IDW", ".IPN")
    if ($specialFiles -contains $Prop["_FileExt"].Value -and !$Prop["_GenerateFileNumber4SpecialFiles"].Value)
    {
        return $null
    }
    if (-Not $Prop["_EditMode"].Value)
    {
        [System.Collections.ArrayList]$numSchemes = @($vault.DocumentService.GetNumberingSchemesByType('Active'))
        if ($numSchemes.Count -gt 1)
        {
            $numSchemes = $numSchemes | Sort-Object -Property IsDflt -Descending
        }

        $allowedNumSchemes = @('Engineering', 'Office')
        $myNumSchemesList = @()
        foreach($numScheme in $numSchemes){
            if($allowedNumSchemes -contains $numScheme.Name){
                $myNumSchemesList += $numScheme
            }
        }

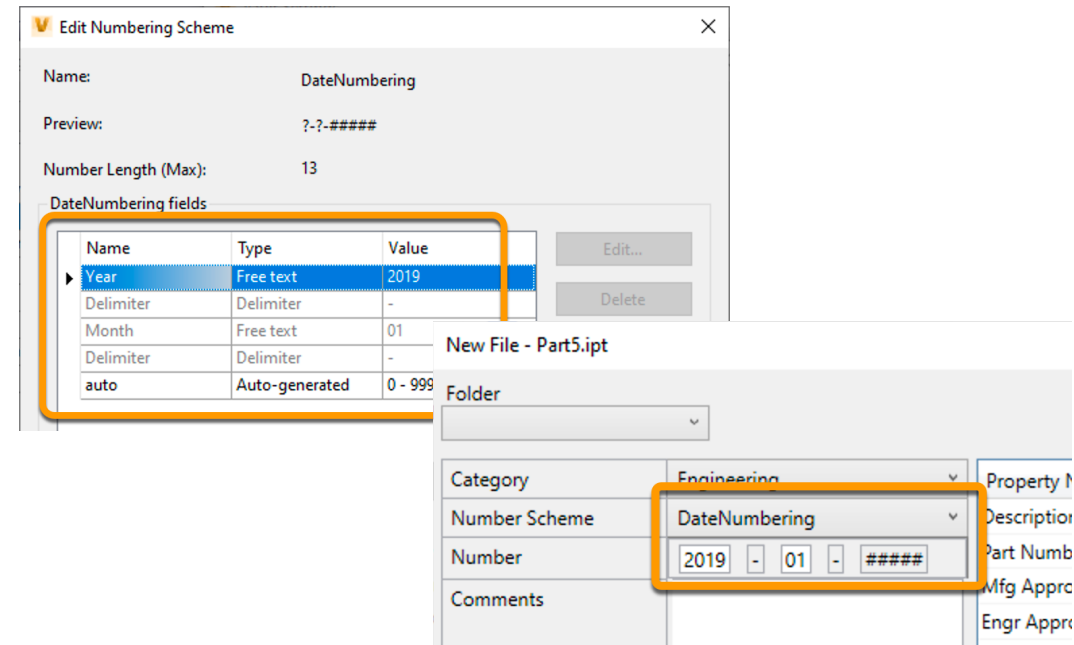
        if ($Prop["_SaveCopyAsMode"].Value)
        {
            $noneNumSchm = New-Object 'Autodesk.Connectivity.WebServices.NumSchm'
            $noneNumSchm.Name = $UIString["LBL77"]
            $numSchemes.Add($noneNumSchm) | Out-Null
        }

        return $myNumSchemesList
    }
}
```


Custom numbering

Numbering scheme with dynamic values

- i.e. Numbering scheme with Year and Month



```
CAD.Custom > addins > Default.ps1 > OnPostCloseDialog {}  
function OnPostCloseDialog  
{  
    $mWindowName = $dswindow.Name  
    switch($mWindowName)  
    {  
        "InventorWindow"  
        {  
            if($prop["_NumSchm"].Value -eq "DateNumbering"){  
                $numSchms = $vault.DocumentService.GetNumberingSchemesByType("Activated")  
                $dateNumberingSchm = $numSchms | Where-Object { $_.Name -eq "DateNumbering" }  
                $currentYear = Get-Date -Format "yyyy"  
                $currentMonth = Get-Date -Format "MM"  
                $newFileName = $vault.DocumentService.GenerateFileName($dateNumberingSchm.SchmID,@($currentYear,$currentMonth))  
                $Prop["DocNumber"].Value = $newFileName  
            }  
        }  
        "AutoCADWindow"  
        {  
            #rules applying for AutoCAD
```


Custom numbering

Numbering scheme with dynamic values

- i.e. Numbering scheme by Product and Type

New File - Part1.ipt

Folder:

Category	Engineering
Number Scheme	ProductNumbering
Number	MX253 - 01 - ####
Comments	

Property Name	Property Value
Type	12
Product	PX015
Description	MX021
Part Number	MX025
Mfg Approved by	PX015
Engr Approved by	PX033
Material	LX555
Stock Number	LX666

```
CAD.Custom > addons > Default.ps1 > OnPostCloseDialog {}  
function OnPostCloseDialog  
{  
    $mWindowName = $dsWindow.Name  
    switch($mWindowName)  
    {  
        "InventorWindow"  
        {  
            if($prop["_NumSchm"].Value -eq "ProductNumbering"){  
                $numSchms = $vault.DocumentService.GetNumberingSchemesByType("Activated")  
                $dateNumberingSchm = $numSchms | Where-Object { $_.Name -eq "ProductNumbering" }  
                $product = $prop["Product"].Value  
                $type = $prop["Type"].Value  
                show-inspector product  
                $newFileName = $vault.DocumentService.GenerateFileNumber($dateNumberingSchm.SchmID,@($product,$type))  
                $Prop["DocNumber"].Value = $newFileName  
            }  
        }  
    }  
}
```

Change property behavior

Change dynamically/programmatically
the property behaviors

- i.e. make a property obligatory
depending on the selected
numbering scheme

Property Name	Property Value
Type	
Description	
Part Number	
Mfg Approved by	
Engr Approved by	
Material	Generic

```
CAD.Custom > addins > Default.ps1 > InitializeWindow {}  
function InitializeWindow  
{  
    #begin rules applying commonly  
    OnNumSchmChanged  
    $pswindow.Title = SetWindowTitle  
    InitializeCategory  
    InitializeNumSchm
```

```
CAD.Custom > addins > Default.ps1 > OnNumSchmChanged {}  
function OnNumSchmChanged  
{  
    $Prop["_NumSchm"].add_PropertyChanged({  
        if($prop["_NumSchm"].Value -eq "ProductNumbering"){  
            $prop["Type"].IsObligatory = $true  
        }  
        else{  
            $prop["Type"].IsObligatory = $null  
        }  
    })  
}
```

Set the bread crumb

Change dynamically/programmatically
the property behaviors

- i.e. make a property obligatory
depending on the selected
numbering scheme

The screenshot displays a software interface for creating a new file. The top section, titled 'New File - Part4.ipt', contains a 'Folder' dropdown and a table for property initialization. The table has two columns: 'Category' and 'Property Name', and two rows: 'Number Scheme' and 'Number'. The 'Category' dropdown is set to 'Engineering', and the 'Number Scheme' dropdown is set to 'Type'. The 'File Name' is 'Part4.ipt'. Below the table, there is a 'Comments' field. The bottom section shows a PowerShell script in a dark-themed editor. The script is titled 'CAD.Custom > addins > Default.ps1 > InitializeWindow {}'. The script defines a function 'InitializeWindow' which calls several initialization functions, including 'SetBreadCrumb'. The 'SetBreadCrumb' function is highlighted with a yellow box. The script also defines a function 'SetBreadCrumb' which sets the text of the 'cmbBreadCrumb_0' and 'cmbBreadCrumb_1' controls based on the 'Product' and 'Type' properties.

Category	Engineering	Property Name	Property Value
Number Scheme	Type	Type	
Number		Product	
File Name	Part4.ipt	Description	
Comments		Part Number	
		Mfg Approved by	
		Engr Approved by	

```
CAD.Custom > addins > Default.ps1 > InitializeWindow {}  
3 function InitializeWindow  
4 {  
5     #begin rules applying commonly  
6     SetBreadCrumb  
7     $dsWindow.Title = SetWindowTitle  
8     InitializeCategory  
9     InitializeNumSchm  
10    InitializeBreadCrumb  
11    InitializeFileNameValidation  
12 }  
CAD.Custom > addins > Default.ps1 > SetBreadCrumb {}  
function SetBreadCrumb  
{  
    $Prop["Product"].add_PropertyChanged({  
        $dsWindow.FindName("cmbBreadCrumb_0").Text = "Products"  
        if($null -ne $dsWindow.FindName("cmbBreadCrumb_1")){  
            $dsWindow.FindName("cmbBreadCrumb_1").Text = $Prop["Product"].Value  
        }  
    })  
  
    $Prop["Type"].add_PropertyChanged({  
        if($null -ne $dsWindow.FindName("cmbBreadCrumb_2")){  
            $dsWindow.FindName("cmbBreadCrumb_2").Text = $Prop["Type"].Value  
        }  
    })  
}
```

Introduction in XAML

```
<ComboBox x:Name="NumSchms" Grid.Column="1" Grid.Row="1"
    DisplayMemberPath="Name"
    SelectedValuePath="Name"
    SelectedValue="{Binding Prop[_NumSchm].Value}"
    ItemsSource="{Binding PsList[GetNumSchms]}"
    IsEnabled="{Binding HasItems, RelativeSource={RelativeSource Self}}"
    Visibility="{Binding NotEditMode, Converter={StaticResource BooleanToVisibilityConverter}}"/>
```

```
<Label Content="{Binding UIString[LBL6], FallbackValue=File Name}"
    Grid.Row="3"
    Grid.Column="0"
    Visibility="{Binding Visibility, ElementName=FILENAME}" />
```

```
<TextBox Grid.Row="3"
    Grid.Column="1"
    x:Name="FILENAME"
    Text="{WPFF:ValidatedBinding Prop[DocNumber].Value}"
    IsReadOnly="{Binding Prop[_EditMode].Value}"
    Visibility="{Binding NumSchmFieldsEmpty, Converter={StaticResource BooleanToVisibilityConverter}}"/>
```

New File - Part1.ipt

Folder:

Category	Engineering	Property Name	Property Value
Number Scheme		Type	
Number		Product	
File Name	Part1.ipt	Description	
Comments		Part Number	
		Mfg Approved by	
		Engr Approved by	
		Material	Generic
		Stock Number	
		LastUpdatedWith	2020 (Build 240168000, 168)
		Designer	coolOrange
		Engineer	
		Checked by	
		Author	coolOrange
		Keywords	
		Tags	

Path: C:\VaultWS\Designs\

OK Cancel

```
<Button Command="{Binding CloseWindowCommand, ElementName=Inventor}"
    IsEnabled="{Binding Properties.IsValid}" Grid.Column="2"
    Width="80" HorizontalAlignment="Right" VerticalAlignment="Top"
    ToolTipService.ShowOnDisabled="True" ToolTip="{Binding UIString[BTN1], FallbackValue=OK}" Height="30"/>
```

Introduction in XAML

```
<Grid x:Name="grdMain" Margin="5">
  <Grid.RowDefinitions>
    <RowDefinition Height="Auto" />
    <RowDefinition Height="Auto" />
    <RowDefinition Height="Auto" />
    <RowDefinition Height="Auto" />
  </Grid.RowDefinitions>
  <Grid.ColumnDefinitions>
    <ColumnDefinition Width="Auto" />
    <ColumnDefinition Width="Auto" />
  </Grid.ColumnDefinitions>
</Grid>
```

```
<GroupBox x:Name="GroupFolder" Grid.ColumnSpan="2" Margin="-6,0,0,5" Border="1">
  <WrapPanel x:Name="BreadCrumb" ItemWidth="165" Margin="1,0,0,0">
    <WrapPanel>
  </WrapPanel>
</GroupBox>
```

```
<Grid x:Name="grdGeneralProps" Grid.Row="1" Grid.Column="1">
  <Grid.RowDefinitions> ...
</Grid.RowDefinitions>
  <Grid.ColumnDefinitions> ...
</Grid.ColumnDefinitions>
  <Label Grid.Row="0" Grid.Column="0" Content="Category">
    <ComboBox x:Name="Categories" Grid.Row="0" Grid.Column="0">
      <Label Grid.Row="1" Grid.Column="0" Content="Number Scheme">
        <ComboBox x:Name="NumSchms" Grid.Column="1">
          <Label Grid.Row="2" Grid.Column="0" Content="File Name">
            <WPFDNumSchemeCtrl x:Name="DSNumSchmsCtrl">
              <Label Content="{Binding UIString[LBL6], Fallschirm}">
                <TextBox Grid.Row="3" Grid.Column="1" x:Name="txtComments">
                  <Label x:Name="lblComments" Content="{Binding UIString[LBL7], Fallschirm}">
                    <TextBox x:Name="txtComments" Grid.Column="1">

```

New File - Part1.ipt

Folder

Category	Engineering	Property Name	Property Value
Number Scheme		Type	
Number		Product	
File Name	Part1.ipt	Description	
Comments		Part Number	
		Mfg Approved by	
		Engr Approved by	
		Material	Generic
		Stock Number	
		LastUpdatedWith	2020 (Build 240168000, 168)
		Designer	coolOrange
		Engineer	
		Checked by	
		Author	coolOrange
		Keywords	

Path

C:\VaultWS\Designs\

OK Cancel

```
<GroupBox x:Name="GroupPath" Header="{Binding UIString[LBL5], Fallschirm}">
  <TextBox Text="{Binding PathAndFileNameHandler.Path}" IsReadOnly="true">
</GroupBox>
```

```
<Grid x:Name="ButtonGrid" VerticalAlignment="Bottom">
  <Grid.ColumnDefinitions>
    <ColumnDefinition Width="140" />
    <ColumnDefinition Width="*" />
    <ColumnDefinition Width="80" />
    <ColumnDefinition Width="20" />
    <ColumnDefinition Width="80" />
  </Grid.ColumnDefinitions>
  <Button Command="{Binding CloseWindowCommand, ElementName=grdMain}">
  </Button>
  <Button Grid.Column="4" Width="80" HorizontalAlignment="Right">
</Grid>
```

Customize the UI

Move a property from the dynamic property grid to the main area

Copy the XAML file to the custom folder

Add Label and TextBox to the according Grid

Hide the Property from the Dynamic Property area

Demo

Property Name	Property Value
Type	
Product	
Description	
Part Number	
Mfg Approved by	
Mgr Approved by	

```
CAD.Custom > Configuration > Inventor.xml
239 IsReadOnly="{Binding Prop[_EditMode].Value}"
240 Visibility="{Binding NumSchmFieldsEmpty, Converter={StaticResource BooleanToVisibilityConverter}, ElementName=
241 <Label x:Name="lblComments" Content="{Binding UIString[LBL7], FallbackValue=Comments}" Grid.Row="4" BorderThickness="1"
242 <TextBox x:Name="txtComments" Grid.Column="1" Grid.Row="4" TextWrapping="Wrap" Text="{Binding Prop[Comments].Value}" Bor
243 VerticalContentAlignment="Top" AccentsReturn="True" VerticalScrollBarVisibility="Auto" Height="50" MaxWidth="18
244 <Label Content="Title" Grid.Row="5" BorderThickness="1" Height="Auto"/>
245 <TextBox Grid.Column="1" Grid.Row="5" Text="{WPFF:ValidatedBinding Prop[Title].Value}" BorderThickness="0,1,1,1"/>
246
```

```
CAD.Custom > Configuration > InventorDynamicProperties.xml
1 <?xml version="1.0" encoding="utf-8"?>
2 <DynamicProperties>
3   <DynamicPropertiesCategory Category="Engineering">
4     <DynamicProperty>Type</DynamicProperty>
5     <DynamicProperty>Product</DynamicProperty>
6     <DynamicProperty Hidden="True">Title</DynamicProperty>
7     <DynamicProperty>Description</DynamicProperty>
8     <DynamicProperty>Part Number</DynamicProperty>
9     <DynamicProperty>Mfg Approved by</DynamicProperty>
```


Customize dialogs

Extend the dialog's capabilities

Example "Select from custom objects"

New File - Part1.ipt

Folder

Category	Engineering	Property Name	Property Value
Number Scheme		Author	coolOrange
Number		Checked by	
File Name	Part1.ipt	Comments	
Comments		Company	Apple
Company	Apple	Description	
		Designer	coolOrange
		Engineer	
		Engr Approved by	
		Keywords	
		LastUpdatedWith	2020 (Build 240168000, 168)
		Manager	
		Mass	0.00

```
<Label Content="Company" Grid.Row="5" BorderThickness="1" Height="Auto"/>
<ComboBox Grid.Column="1" Grid.Row="5" SelectedValue="{Binding Prop[Company].Value}" ItemsSource="{Binding PsList[GetCompanies]}" Border
```

```
CAD.Custom > addons > > Default.ps1 > ...
31 function GetCompanies
32 {
33     $searchConds = @()
34     $searchCond += New-Object Autodesk.Connectivity.WebServices.SrchCond
35     $searchCond.PropDefId = 0
36     $searchCond.SrchOper = 1
37     $searchCond.SrchTxt = ""
38     $searchCond.PropTyp = "AllProperties"
39     $searchCond.SrchRule = "Must"
40     $searchConds += $searchCond
41     $bookmark = ""
42     $srchStatus = New-Object Autodesk.Connectivity.WebServices.SrchStatus
43     $entities = $vault.CustomEntityService.FindCustomEntitiesBySearchConditions($searchConds,$null,[ref]$bookmark,[ref]$srchStatus)
44     $companies = @()
45     foreach($entity in $entities)
46     {
47         $companies += $entity.Name
48     }
49     return $companies
50 }
```

Customize dialogs

Extend the dialog's capabilities

Example "Cascading drop down"

```
<Label Content="Products" Grid.Row="5" BorderThickness="1" Height="Auto"/>
<ComboBox Grid.Column="1" Grid.Row="5" SelectedValue="{Binding Prop[Product].Value}" ItemsSource="{Binding PsList[GetProducts]}" Border
<Label Content="Type" Grid.Row="6" BorderThickness="1" Height="Auto"/>
<ComboBox Name="ProductTypes" Grid.Column="1" Grid.Row="6" SelectedValue="{Binding Prop[Type].Value}" BorderThickness="0,1,1,1" MaxW
```

```
CAD.Custom > addons > Default.ps1 > InitializeWindow {}
3 function InitializeWindow
4 {
5     #begin rules applying commonly
6     OnProductChange
7     $dsWindow.Title = SetWindowTitle
8     InitializeCategory
```

New File - Part1.ipt

Folder

Category	Engineering	Property Name	Property Value
Number Scheme		Type	
Number		Product	MX012
File Name	Part1.ipt	Description	
Comments		Part Number	
		Mfg Approved by	
Products	MX012	Engr Approved by	
Type		Material	Generic
		Stock Number	
		LastUpdatedWith	2020 (Build 240168000, 168)
		Designer	coolOrange
		Engineer	
		Checked by	

```
function OnProductChange
{
    $prop["Product"].Add_PropertyChanged({
        GetTypesByProduct
    })
}

function GetProducts
{
    $products = @("MX012", "MX455", "PX111", "PX332")
    return $products
}

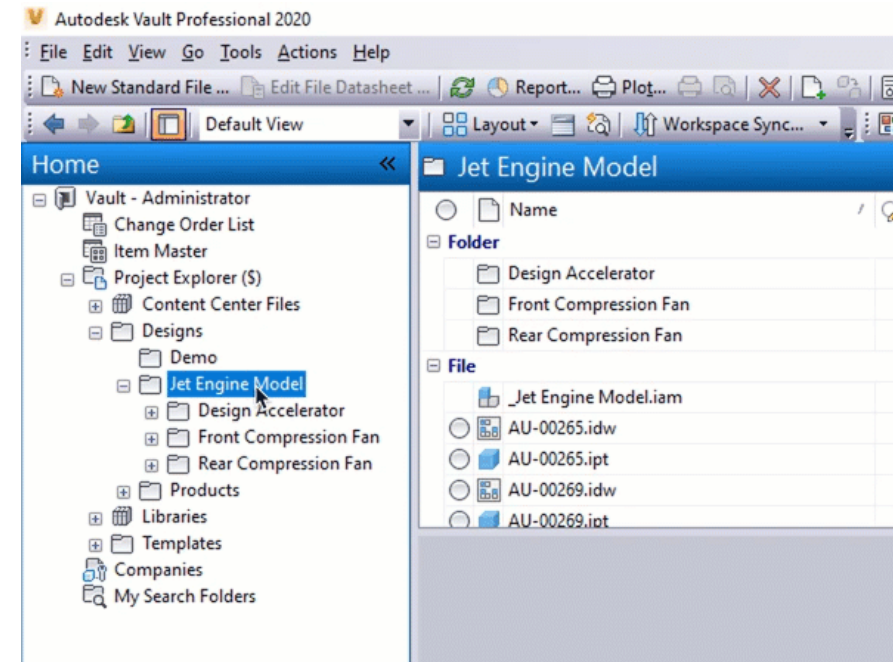
function GetTypesByProduct
{
    if($prop["Product"].Value -eq "MX012"){
        $products = @("12-A", "12-B", "12-C")
    }
    if($prop["Product"].Value -eq "MX455"){
        $products = @("455S", "455L", "455XL")
    }
    if($prop["Product"].Value -eq "PX111"){
        $products = @("auto", "manual")
    }
    if($prop["Product"].Value -eq "PX332"){
        $products = @("blue", "red", "orange", "yellow")
    }
    $dsWindow.FindName("ProductTypes").ItemsSource = $products
}
```

Custom menu items

Example “Copy folder structure”

Copy the MenuDefinitions.xml into the
Custom folder and apply changes

```
Vault.Custom > MenuDefinitions.xml
1 <?xml version="1.0" encoding="utf-8"?>
2 <myMenu>
3   <MenuItem>
4     <CopyFolder Label="Copy folder" Description="Copy folder" Hint="Copy folder" PSFile="CopyFolder.ps1"
5       Image="NewStandardFolder.ico" ToolbarPaintStyle="TextAndGlyph" NavigationTypes="Folder"
6       MultiSelectEnabled="False" />
7     <PasteFolder Label="Paste folders" Description="Paste folders" Hint="Paste folders" PSFile="PasteFolder.ps1"
8       Image="NewStandardFolder.ico" ToolbarPaintStyle="TextAndGlyph" NavigationTypes="Folder"
9       MultiSelectEnabled="False" />
10   <NewTask Label="$UIString[MNU22]" Description="$UIString[MNU17]" Hint="$UIString[MNU18]" ...
11   <EditTask Label="$UIString[MNU21]" Description="$UIString[MNU19]" Hint="$UIString[MNU20]" ...
12   <NewFile Label="$UIString[MNU1]" Description="$UIString[MNU11]" Hint="$UIString[MNU7]" PSFile="CreateFile.ps1" ...
13   <EditFile Label="$UIString[MNU2]" Description="$UIString[MNU12]" Hint="$UIString[MNU8]" PSFile="EditFile.ps1" ...
14   <NewFolder Label="$UIString[MNU3]" Description="$UIString[MNU13]" Hint="$UIString[MNU9]" PSFile="CreateFolder.ps1" ...
15   <EditFolder Label="$UIString[MNU4]" Description="$UIString[MNU14]" Hint="$UIString[MNU10]" PSFile="EditFolder.ps1" ...
16   <ShowLog Label="$UIString[MNU5]" Description="$UIString[MNU15]" Hint="$UIString[MNU5]" PSFile="ShowLog.ps1" ...
17   <AboutDialog Label="$UIString[MNU6]" Description="$UIString[MNU16]" Hint="$UIString[MNU6]" ...
18 </MenuItem>
19
20 <CommandSite>
21   <FolderContext Label="myMenu" DeployAsPullDown="False" Location="FolderContextMenu">
22     <Item Name="NewFile"></Item>
23     <Item Name="NewFolder"></Item>
24     <Item Name="EditFolder"></Item>
25     <Item Name="CopyFolder"></Item>
26     <Item Name="PasteFolder"></Item>
27   </FolderContext>
28 </CommandSite>
```



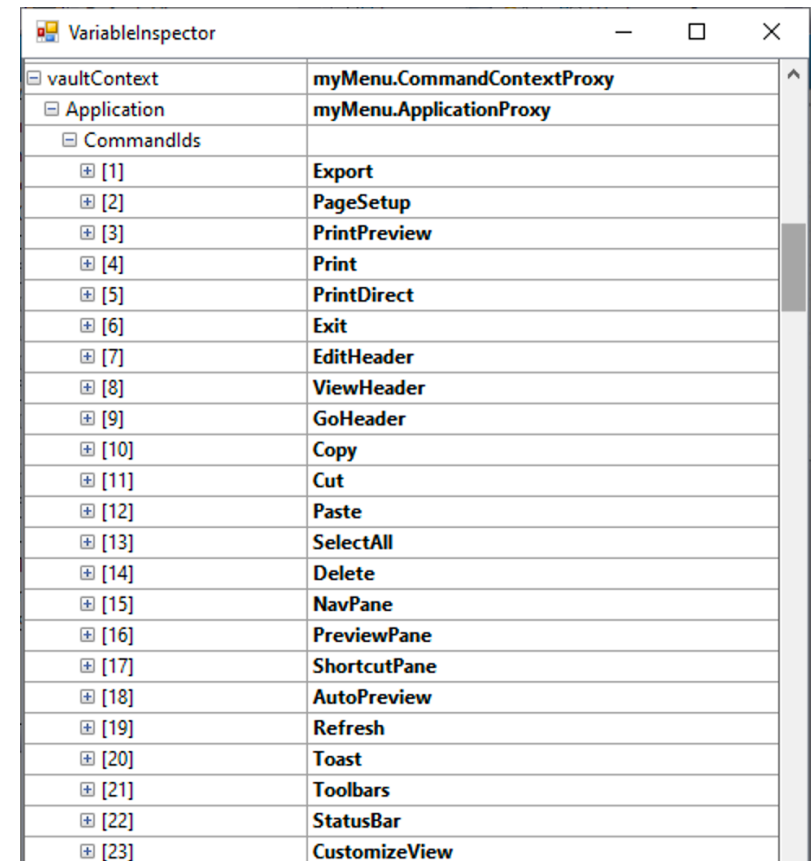
```
Vault.Custom > addinVault > Menus > CopyFolder.ps1 > ...
1
2 $folderId=$vaultContext.CurrentSelectionSet[0].Id
3
4 [AppDomain]::CurrentDomain.SetData("FolderToCopy",$folderId)
```

```
Vault.Custom > addinVault > Menus > PasteFolder.ps1 > ...
1 $targetId=$vaultContext.CurrentSelectionSet[0].Id
2 $vaultContext.ForceRefresh = $true
3
4 $sourceId = [AppDomain]::CurrentDomain.GetData("FolderToCopy")
5
6 2 references
7 function BuildFolders($sourceId,$targetId){
8   $sourceFolders = $vault.DocumentService.GetFoldersByParentId($sourceId,$false)
9   foreach($sourceFolder in $sourceFolders)
10   {
11     $targetFolder = $vault.DocumentService.AddFolder($sourceFolder.Name,$targetId,$sourceFolder.IsLib)
12     BuildFolders -sourceFolder $sourceFolder.Id -targetId $targetFolder.Id
13   }
14 }
15 BuildFolders -sourceId $sourceId -targetId $targetId
16
```

Suppress menus

At the end of the `MenuDefinition.xml`, the section `SuppressMenuItems` holds the list of menus to be removed

The list of all menus is hold in the variable `vaultContext.Application.CommandIds`



The VariableInspector window displays the following structure:

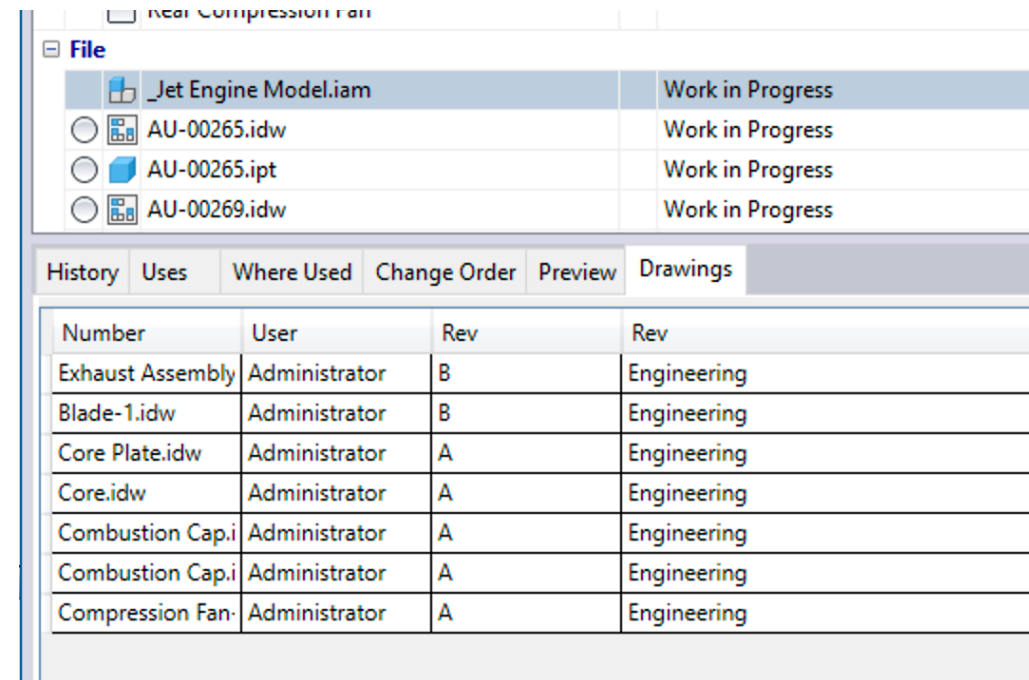
vaultContext	myMenu.CommandContextProxy
Application	myMenu.ApplicationProxy
CommandIds	
[1]	Export
[2]	PageSetup
[3]	PrintPreview
[4]	Print
[5]	PrintDirect
[6]	Exit
[7]	EditHeader
[8]	ViewHeader
[9]	GoHeader
[10]	Copy
[11]	Cut
[12]	Paste
[13]	SelectAll
[14]	Delete
[15]	NavPane
[16]	PreviewPane
[17]	ShortcutPane
[18]	AutoPreview
[19]	Refresh
[20]	Toast
[21]	Toolbars
[22]	StatusBar
[23]	CustomizeView

```
Vault.Custom > MenuDefinitions.xml
69      Folder.Rename
70      -->
71      <SuppressMenuItems>
72      NewFolder,NewFolderGroupMenu,NewFolderGroupButton,NewFolderForContextMenus
73      </SuppressMenuItems>
74    </mymenu>
```

Create new tabs

Copy the Datasheet.XAML to your custom folder

Rename the file and apply changes



The screenshot shows a software interface with a file list and a table of drawing data. The file list is titled 'File' and contains the following items:

File	Status
Jet Engine Model.iam	Work in Progress
AU-00265.idw	Work in Progress
AU-00265.ipt	Work in Progress
AU-00269.idw	Work in Progress

Below the file list is a table with the following columns: Number, User, Rev, and Rev. The table contains the following data:

Number	User	Rev	Rev
Exhaust Assembly	Administrator	B	Engineering
Blade-1.idw	Administrator	B	Engineering
Core Plate.idw	Administrator	A	Engineering
Core.idw	Administrator	A	Engineering
Combustion Cap.i	Administrator	A	Engineering
Combustion Cap.i	Administrator	A	Engineering
Compression Fan	Administrator	A	Engineering

Vault.Custom > addinVault > Default.ps1 > OnTabContextChanged {}

```
1
2 references
3 function OnTabContextChanged
4 {
5     $xamlFile = [System.IO.Path]::GetFileName($VaultContext.UserControl.XamlFile)
6
7     if ($VaultContext.SelectedObject.TypeId.SelectionContext -eq "FileMaster" -and $xamlFile -eq "[
8         $selectedFileId = $VaultContext.SelectedObject.Id
9         #File = $Vault.DocumentService.GetLatestFileByMasterId($selectedFileId)
10        $fileAssocs = $Vault.DocumentService.GetLatestFileAssociationsByMasterIds(@( $selectedFileId
11        $drawings = @()
12        if($fileAssocs[0].FileAssocs -ne $null){
13            foreach($assoc in $fileAssocs[0].FileAssocs){
14                if($assoc.ParFile.Name.EndsWith(".idw")){
15                    $drawings += $assoc.ParFile
16                }
17            }
18        }
19        $dsWindow.FindName("ListOfDrawings").ItemsSource = $drawings
20    }
```

Vault.Custom > Configuration > File > Drawings.xaml

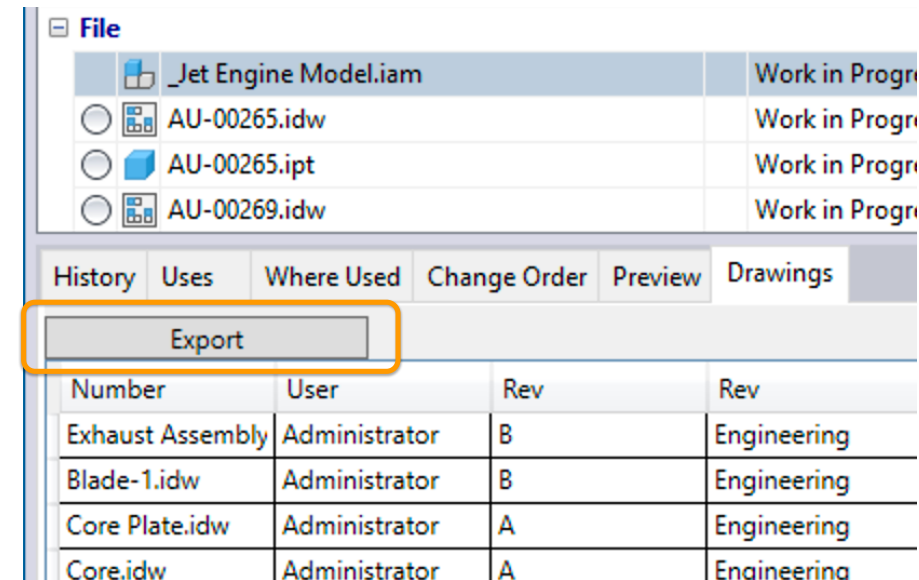
```
<UserControl xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
xmlns:behaviours="clr-namespace:Common.Wpf;assembly=Common"
xmlns:d="http://schemas.microsoft.com/expression/blend/2008" xmlns:mc="http://schemas.openxml
xmlns:WPF="clr-namespace:CreateObject.WPF;assembly=CreateObject"
x:Name="MainWindow"
behaviours:TabTitleBehavior.TabTitle="Drawings">
<UserControl.Resources> ...
</UserControl.Resources>
<Grid Margin="0.5,5,0">
<DataGrid Name="ListOfDrawings">
    <DataGrid.Columns>
        <DataGridTextColumn Binding="{Binding Name}" Header="Number" Width="100" IsReadOnly="True"
        <DataGridTextColumn Binding="{Binding CreateUserName}" Header="User" Width="100" IsReadOnly="True"
        <DataGridTextColumn Binding="{Binding FileRev.Label}" Header="Rev" Width="100" IsReadOnly="True"
        <DataGridTextColumn Binding="{Binding Cat.CatName}" Header="Rev" Width="100" IsReadOnly="True"
    </DataGrid.Columns>
</DataGrid>
</Grid>
</UserControl>
```


Buttons

Create a Button and bind your Command to a PowerShell function

Command="{Binding PsCmd[YourFunction]}"

Write your PowerShell function



```
<Grid Margin="0,5,5,0">
  <Grid.RowDefinitions>
    <RowDefinition Height="Auto" />
    <RowDefinition Height="Auto" />
  </Grid.RowDefinitions>
  <Button Content="Export" Command="{Binding PsCmd[ExportDrawingList]}" Width="150" HorizontalAlignment="Left" />
  <DataGrid Name="ListOfDrawings" Grid.Row="1">
    <DataGrid.Columns>
      <DataGridTextColumn Binding="{Binding Name}" Header="Number" Width="100" IsReadOnly="True"/>
      <DataGridTextColumn Binding="{Binding CreateUserName}" Header="User" Width="100" IsReadOnly="True"/>
      <DataGridTextColumn Binding="{Binding Rev}" Header="Rev" Width="100" IsReadOnly="True"/>
      <DataGridTextColumn Binding="{Binding Rev}" Header="Rev" Width="100" IsReadOnly="True"/>
    </DataGrid.Columns>
  </DataGrid>
</Grid>
```

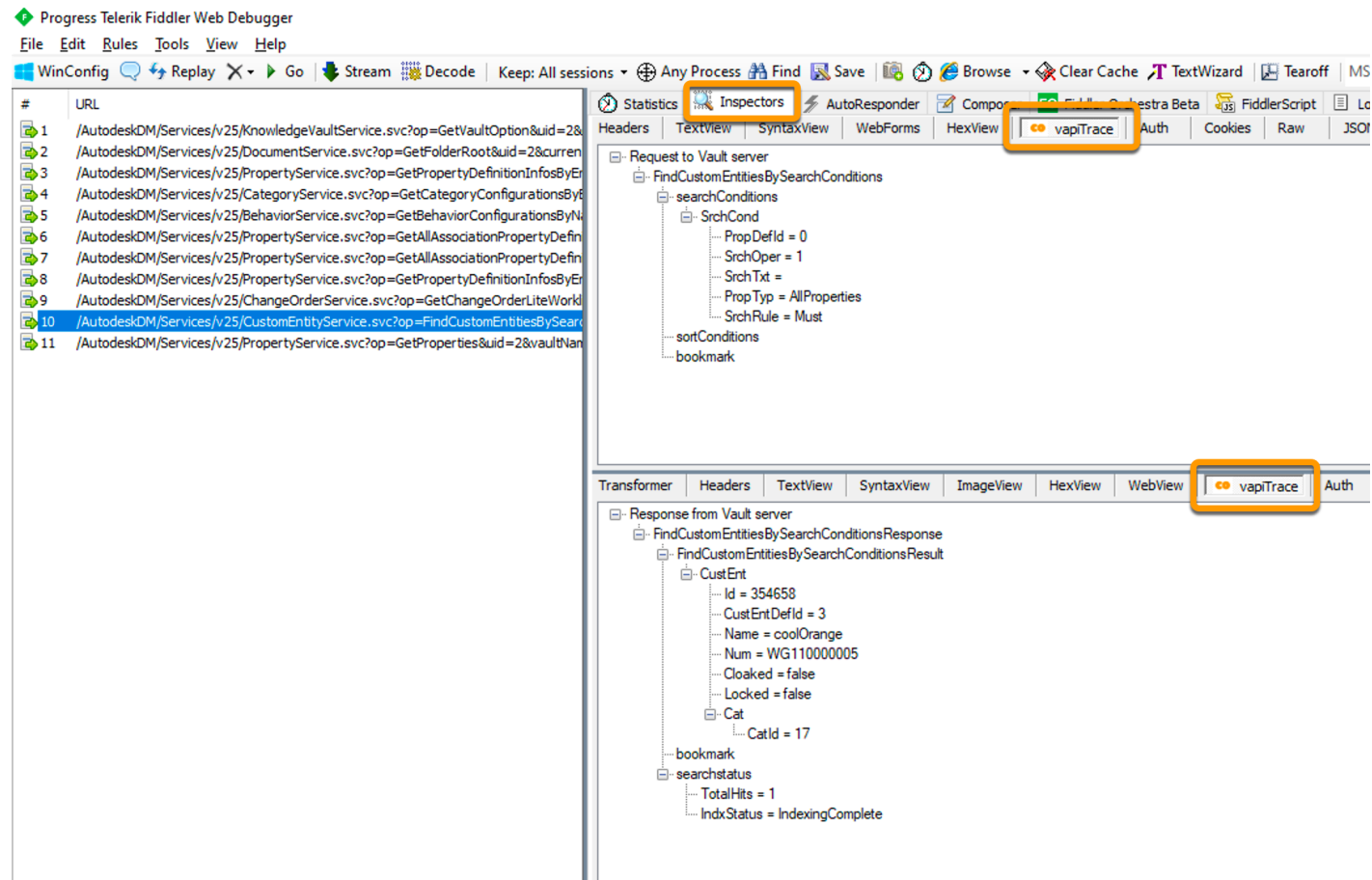
```
function ExportDrawingList
{
  #do something
}
```


Coding Techniques



How to find the right code

With vapiTrace you can monitor
the Vault client -> server communication
and identify the appropriate API calls
and parameters



Add_OnPropertyChanged

In order to react when the value of a property changes and execute some custom business logic

```
$Prop[ "PropertyName" ].add_PropertyChanged( {  
    Write your custom code here ...  
})
```

FindName

In order to modify a UI element from the code

```
$dsWindow.FindName( "NameOfTheUiElement" )
```

All the properties the given UI element has, are now accessible via PowerShell

i.e. `$dsWindow.FindName("NameOfTheUiElement").IsEnabled = $false`

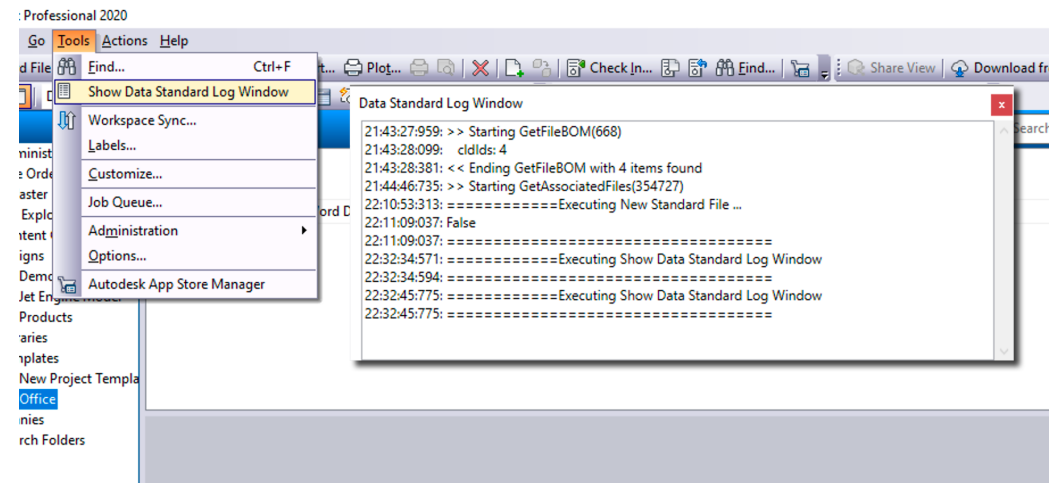
Debugging

```
$dsDiag.Clear()
```

```
$dsDiag.Trace("my message")
```

```
$dsDiag.ShowLog()
```

```
$dsDiag.Inspect()
```



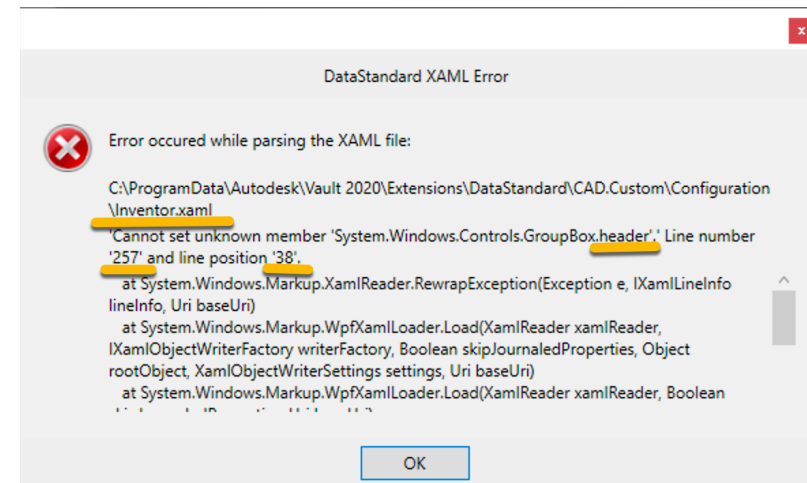
Errors in dialogs (XAML)

Upper/Lower cases

Wrong attributes

Missing quote elements

Missing brackets



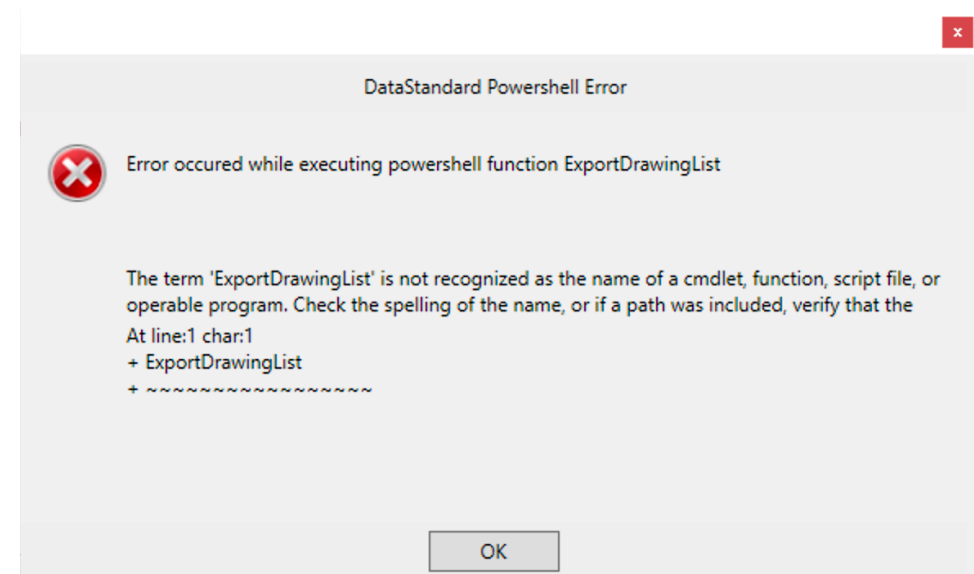
```
254         </DataGrid.Columns>
255     </DataGrid>
256 </Grid>
257 <GroupBox x:Name="GroupPath" header="{Binding UIString[LBL5], FallbackValue=
258     <TextBox Text="{Binding PathHandlerNameHandler.Path}" IsReadOnly="True"
259 </GroupBox>
```


Errors in PowerShell

Logic error

Syntax error

Missing brackets / quotes



```
<RowDefinition Height="Auto" />  
</Grid.RowDefinitions>  
<Button Content="Export" Command="{Binding PsCmd[ExportDrawingList]}" Width="150" Height="30" />  
<DataGrid Name="ListOfDrawings" Grid.Row="1" />  
<DataGrid.Columns>
```

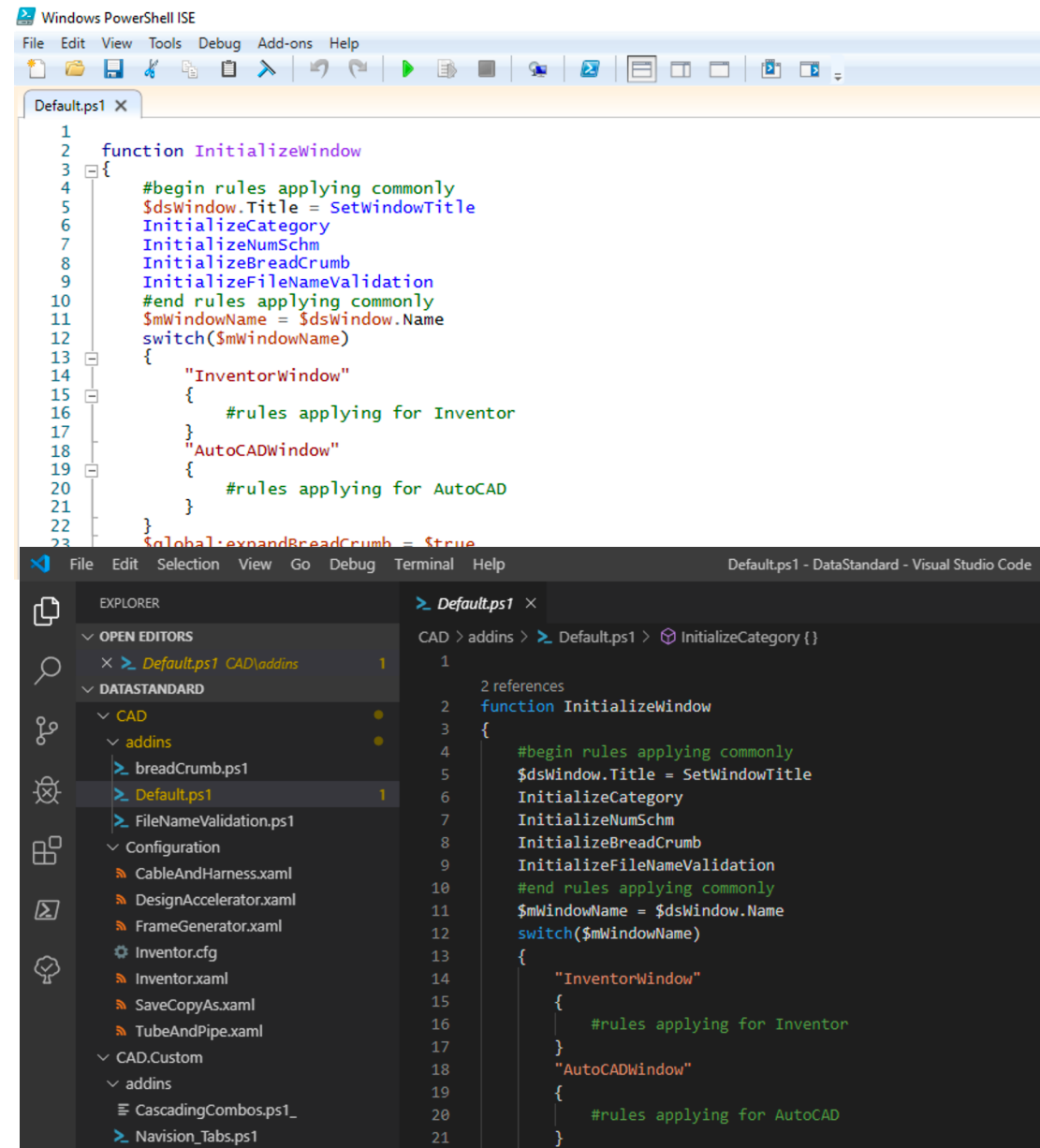
Editors



For PowerShell editing

PowerShell ISE

Visual Studio Code

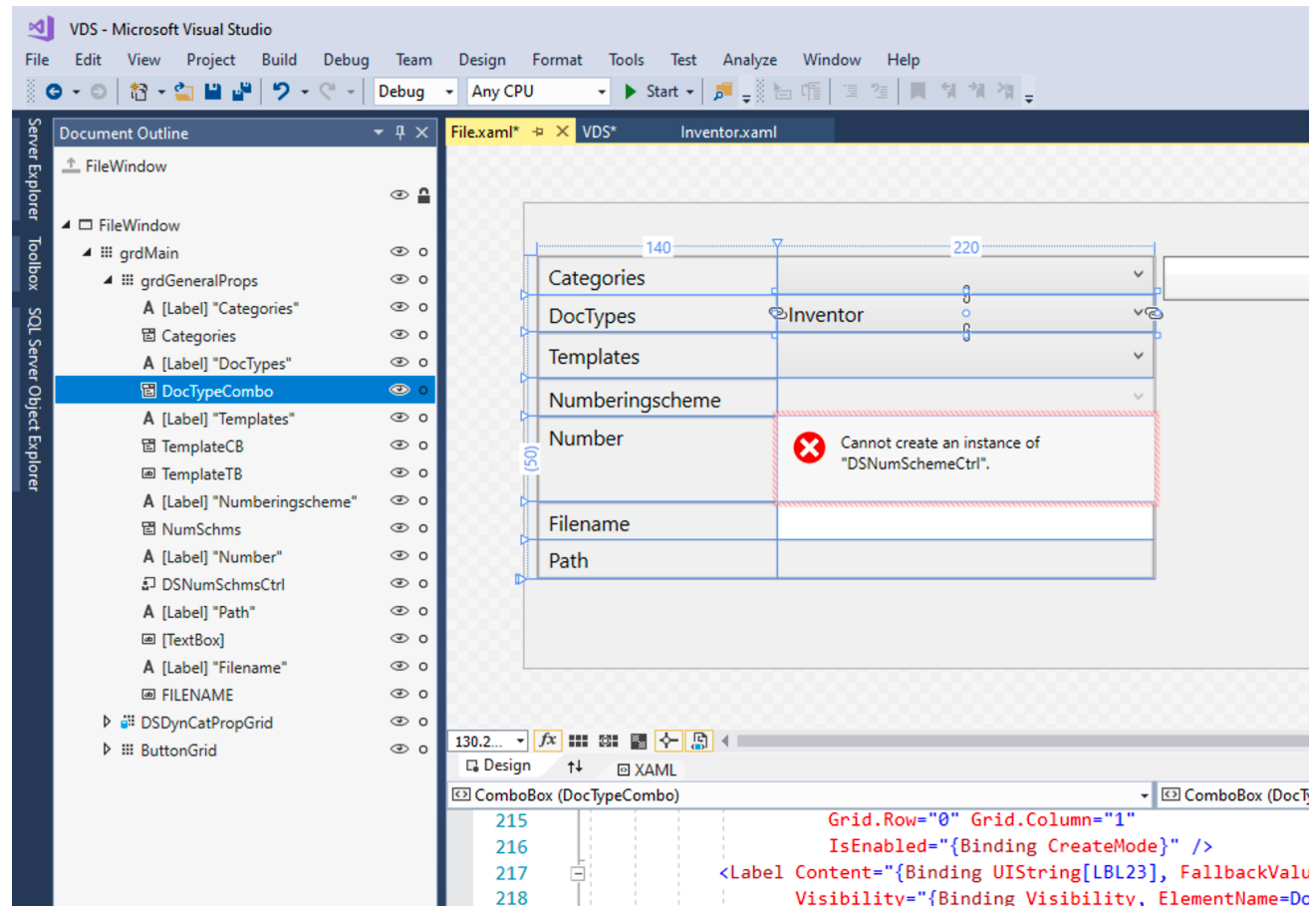


The image displays two side-by-side screenshots of PowerShell editing environments. The top screenshot shows the Windows PowerShell ISE (Integrated Scripting Environment) interface. It features a single-pane window titled 'Default.ps1' with a menu bar (File, Edit, View, Tools, Debug, Add-ons, Help) and a toolbar. The script content is displayed in a light blue theme with line numbers 1 through 23. The script defines a function 'InitializeWindow' that sets window title, initializes categories, and applies rules for 'InventorWindow' and 'AutoCADWindow'. The bottom screenshot shows the Visual Studio Code interface. It has a dark theme with a sidebar on the left containing 'EXPLORER' and 'OPEN EDITORS' views. The 'EXPLORER' view shows a file tree for 'DATASTANDARD' with subfolders 'CAD' and 'addins'. The 'OPEN EDITORS' view shows 'Default.ps1' as the active file. The main editor area displays the same script content as the PowerShell ISE, but with a dark background and a different syntax highlighting scheme. The status bar at the bottom indicates 'Default.ps1 - DataStandard - Visual Studio Code'.

```
1 function InitializeWindow
2 {
3     #begin rules applying commonly
4     $dsWindow.Title = SetWindowTitle
5     InitializeCategory
6     InitializeNumSchm
7     InitializeBreadCrumb
8     InitializeFileNameValidation
9     #end rules applying commonly
10    $mWindowName = $dsWindow.Name
11    switch($mWindowName)
12    {
13        "InventorWindow"
14        {
15            #rules applying for Inventor
16        }
17        "AutoCADWindow"
18        {
19            #rules applying for AutoCAD
20        }
21    }
22    $global:expandBreadCrumb = $true
23 }
```

For UI editing

Visual Studio



<https://blog.coolorange.com/2015/07/24/editing-data-standard-2016-files-with-visual-studio/>

Where to find help

Autodesk knowledge base for official documentation

<http://help.autodesk.com/view/VAULT/2020/ENU/?guid=GUID-5030830E-69F8-4A57-ABE4-D1D430C58F95>

Autodesk Vault forum for help

<https://forums.autodesk.com/t5/vault-customization/bd-p/301>

Your reseller

coolOrange blog

<http://blog.coolOrange.com>

Download this material

<https://github.com/coolOrangeLabs/DataStandardTrainingMaterial>



15 minutes for a chance at some Autodesk Bling and a \$200 Amazon gift card?



Take the Data Management/Collaboration Survey

Just follow this link!

<https://www.surveymonkey.com/r/AU2019CollabSurvey>

Don't forget to include your name and contact information for a chance to win**

Drawing will be held at 2:00 PT on Thursday, November 21st

**Must pick up prize in person (more info in the survey)

***Autodesk employees, Autodesk Partners (Resellers, SI's, ADN, etc.) are not eligible





Autodesk and the Autodesk logo are registered trademarks or trademarks of Autodesk, Inc., and/or its subsidiaries and/or affiliates in the USA and/or other countries. All other brand names, product names, or trademarks belong to their respective holders. Autodesk reserves the right to alter product and services offerings, and specifications and pricing at any time without notice, and is not responsible for typographical or graphical errors that may appear in this document.

© 2019 Autodesk. All rights reserved.

