
powerVault

coolOrange s.r.l

Mar 06, 2024

POWERSVULT

1	Installation	1
1.1	Requirements	1
1.2	Install Locations	1
1.3	Updates	2
1.4	Uninstall	2
2	Getting started	3
2.1	Start the PowerShell environment	3
2.2	Connect to Vault (optional)	3
2.3	Get a file from Vault	3
2.4	Examine the file	4
2.5	Update the file	4
2.6	Download the file	5
3	Multilingual Vault support	7
3.1	Property System Names	7
4	Code Reference	11
4.1	Cmdlets	11
4.2	Objects	43
5	Logging	73
5.1	When to change the logging behavior?	73
5.2	LogFile	73
5.3	PowerShell IDE	74
6	Change logs	75
6.1	powerVault v24	75
6.2	powerVault v23	76
6.3	powerVault v22	82
6.4	powerVault v21	84
6.5	powerVault v20	86
6.6	powerVault v19	87
6.7	powerVault v18	88
6.8	powerVault v17	90
6.9	powerVault v16	91

INSTALLATION

1.1 Requirements

As powerVault uses *Vault Explorer* components, the [Vault system requirements](#) defined by Autodesk leads.

Operating System: 64-bit only

- Microsoft Windows 10
- Microsoft Windows 11

Autodesk Vault Client: 2024 / 2023 / 2022 / 2021

- Vault Professional
- Vault Workgroup

Windows PowerShell: PowerShell 4.0 or [higher](#)

1.1.1 Setup

The powerVault setup is delivered as an executable and accepts the standard windows installer arguments documented [here](#).

To accept the products EULA when starting the setup in silent mode pass the **ACCEPT_EULA=1** argument.

1.2 Install Locations

powerVault is installed in the following locations on your system:

- All program libraries are placed in *C:\Program Files\coolOrange\Modules\powerVault*

Following shared libraries are installed in *GAC*:

- coolOrange.Logging.dll
- coolOrange.VaultServices_[Vault Version].dll

Following shortcuts are added to the start menu:

- **powerVault Console** - Opens the PowerShell Console and loads the powerVault module
- **powerVault Information** - Opens the About dialog with product related information
- **powerVault Logs** - Opens the log file location

1.3 Updates

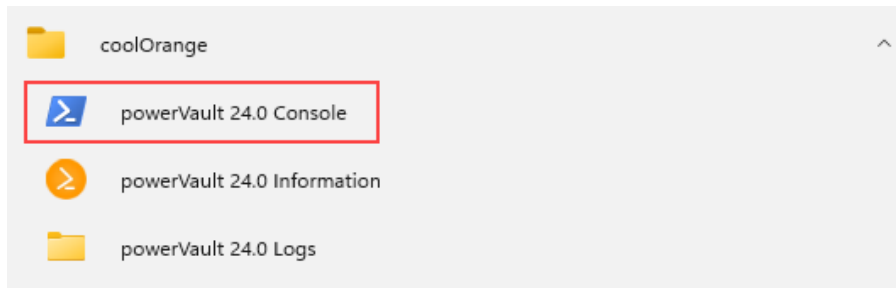
To install a newer version of powerVault just execute the setup file of the new version. This will automatically update the files in the existing installation.

1.4 Uninstall

In case you want to remove powerVault from your computer you can:

- Execute the setup file again. This will give you the option to repair or remove powerVault. Click on “Remove” to uninstall the program.
- Go to “Control Panel - Programs and Features”, find “coolOrange powerVault for Vault” and run “Uninstall”.

GETTING STARTED



2.1 Start the PowerShell environment

In order to get started either open any PowerShell IDE and load the powerVault Module by calling `Import-Module powerVault` or open the *powerVault Console* shortcut in the start menu, which already loads powerVault for you.

2.2 Connect to Vault (optional)

To use the powerVault Cmdlet's a Vault connections is required.

In Vault applications (inside *powerEvents* client customizations or Vault extensions such as *Data Standard*), the Cmdlet's automatically detect and **reuse** the application's latest Vault connection.

However, in Powershell IDE's where a Vault connection does not already exist, the *Open-VaultConnection* cmdlet can be used to create a new connection, passing the required credentials.

2.3 Get a file from Vault

With the Vault Connection available, you can start working with powerVault.

Lets search for a random file which was created by the connected user:

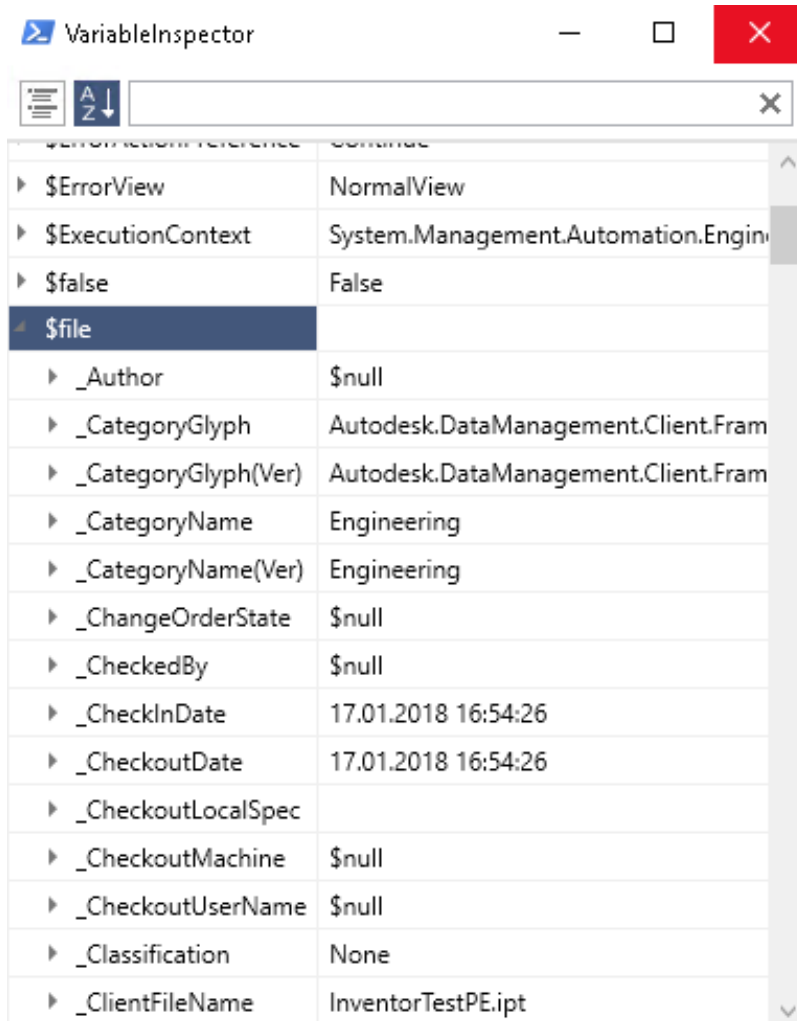
```
$user = $vaultConnection.UserName  
$file = Get-VaultFile -Properties @{"Created By"=$user}
```

2.4 Examine the file

To check which properties our file has, we have two options.

Either we just let the PowerShell Host print the `$file` object or we look at it in the Inspector window by calling the *Show-Inspector* Cmdlet.

```
Show-Inspector -Highlight 'file'
```



The screenshot shows a window titled 'VariableInspector' with a search bar and a list of variables. The '\$file' variable is selected and expanded, showing its properties and values.

Property Name	Value
\$ErrorView	NormalView
\$ExecutionContext	System.Management.Automation.Engine
\$false	False
\$file	
_Author	\$null
_CategoryGlyph	Autodesk.DataManagement.Client.Fram
_CategoryGlyph(Ver)	Autodesk.DataManagement.Client.Fram
_CategoryName	Engineering
_CategoryName(Ver)	Engineering
_ChangeOrderState	\$null
_CheckedBy	\$null
_CheckInDate	17.01.2018 16:54:26
_CheckoutDate	17.01.2018 16:54:26
_CheckoutLocalSpec	
_CheckoutMachine	\$null
_CheckoutUserName	\$null
_Classification	None
_ClientFileName	InventorTestPE.ipt

2.5 Update the file

In case you want to update the file you can achieve this with the *Update-VaultFile* Cmdlet.

In this example we add an attachment to the file:

```
$attachment = Get-VaultFile -Properties @{'File Extension'='dwf'} #Search for a .dwf file
$updatedFile = Update-VaultFile -File $file.'Full Path' -Attachments @($attachment.'Full_
Path') -Comment "Set attachment"
```


2.6 Download the file

You may want to download the file from Vault. For this operation you can use the *Save-VaultFile* Cmdlet. In this example we download the file with the recently added attachment:

```
$downloadedFiles = Save-VaultFile -File $updatedFile.'Full Path' -IncludeAttachments -  
➔DownloadDirectory "C:\Temp\Vault\Sample"  
  
$downloadedFiles | select {$_.LocalPath}  
#C:\Temp\Vault\Sample\Designs\InventorTestPE.ipt  
#C:\Temp\Vault\Sample\InventorTestPE.dwf
```


MULTILINGUAL VAULT SUPPORT

Since powerVault object like *File*, *Item*, *FileBomRow*, *ItemBomRow* etc. are structured in a way that all Vault properties are directly accessible via their display name, your jobs could look something like this:

```
if( $file.'File Name'.EndsWith('.ipt')) {  
    #...  
}
```

This job will only work on English Vault environments. When running against a German Vault Server the equivalent job would look like this:

```
if( $file.'Dateiname'.EndsWith('.ipt')) {  
    #...  
}
```

If you need to write language neutral scripts that are running independent of the Vault language, powerVault gives you the possibility to access all the default Vault Properties with their **System Name** too. Following script would work on both, German and English Vault environment:

```
if( $file._Name.EndsWith('.ipt')) {  
    #...  
}
```

3.1 Property System Names

PowerVault *Objects* have all the properties attached via their display names. Additionally, all the default properties are attached in the syntax “_SystemName”.

Note: Not all the properties are accessible via this approach on powerVault entities!

Custom User Defined Properties e.g. are not part of the default Vault Properties List since they have system names that are not human readable (they contain {GUID} values).

To get a list of all the properties that are available via their system names, just run following command on the entity that you want.

For *files* the property list can be retrieved like this:

```
$file | Format-List -Property _*
```

```

_Classification           : None
_VersionNumber            : 13
_Comment                  : Property Edit
_NumManualAttachments     : Autodesk.DataManagement.Client.Framework.Vault.Currency.
↳ Properties.ImageInfo
_DateVersionCreated       : 19.04.2018 17:35:26
_CreateUserName           : coolOrange
_CheckInDate              : 19.04.2018 17:35:26
_ClientFileName           : Pad Lock.iam
_ClientFileName(Ver)      : Pad Lock.iam
_ModDate                  : 19.04.2018 17:35:20
_FileSize                 : 1064448
_ItemLinked               : True
_CheckoutLocalSpec        :
_CheckoutMachine          :
_CheckoutDate             : 19.04.2018 17:35:24
_CheckoutUserName         :
_Hidden                   : False
_LatestVersion            : True
_ControlledByChangeOrder  : False
_ChangeOrderState         :
_VisualizationAttachment  : None
_Originator               : coolOrange
_OrigCreateDate           : 09.09.2015 16:00:20
_Thumbnail                : Autodesk.DataManagement.Client.Framework.Vault.Currency.
↳ Properties.ThumbnailInfo
_Provider                 : Inventor
_iLogicRuleStatus         :
_FolderPath               : $/Designs/Padlock/Assemblies
_Name                     : Pad Lock.iam
_Extension                : iam
_Compliance               : Compliant
_Compliance(Ver)          : Compliant
_LatestReleasedRevision   : False
_ReleasedRevision         : False
_InitReleaseDate          :
_InitApprover             :
_CategoryName             : Base
_CategoryName(Ver)        : Base
_CategoryGlyph            : Autodesk.DataManagement.Client.Framework.Vault.Currency.
↳ Properties.ImageInfo
_CategoryGlyph(Ver)       : Autodesk.DataManagement.Client.Framework.Vault.Currency.
↳ Properties.ImageInfo
_LifeCycleDefinition      :
_LifeCycleDefinition(Ver) :
_State                    :
_State(Ver)               :
_RevisionDefinition       :
_RevisionDefinition(Ver)  :
_Revision                 :
_FileReplicated           : True
_Author                   : B. ROEPKE
_Comments                 :

```

(continues on next page)

(continued from previous page)

```

_Keywords           : Vault, Tutorial, Padlock
_RevNumber          :
_Subject            : test
_Title              : Pad Lock
_Company            : Autodesk, Inc.
_DWGCreatorName     :
_DWGCreatorVersion  :
_CheckedBy          :
_Cost               : 17
_CostCenter         :
_CreationDate       :
_Description        : PAD LOCK ASSEMBLY
_Material           :
_PartNumber         : ERP-41881007
_StockNumber        :
_UserStatus         :
_GeoRss             :
_Designer           : B. ROEPKE
_Engineer           : B. ROEPKE
_EngrApprovedBy     : D. BRISSON
_Manager            : klaus
_MfgApprovedBy      :
_Project            : PADLOCK
_TypeTag            :
_ItemAssignable     : True
_Obsolete           : False
_HasDrawing         : True
_HasParentRelationship : True
_HasModelState     :
_IsTableDriven      :
_IsTrueModelState   :
_LatestReleaseDate  :
_LatestApprover     :
_EntityType         : File
_EntityTypeID        : FILE
_EntityIcon          : Autodesk.DataManagement.Client.Framework.Vault.Currency.
↪Properties.ImageInfo
_EntityPath          : $/Designs/Padlock/Assemblies
_FullPath           : $/Designs/Padlock/Assemblies/Pad Lock.iam
_HasAttachments     : False
_LinkTargetPath     :
_EntityDescription   : Autodesk Inventor Assembly
_VaultStatus        : Autodesk.DataManagement.Client.Framework.Vault.Currency.
↪Properties.EntityStatusImageInfo
_VaultStatusModifier : False

```


CODE REFERENCE

4.1 Cmdlets

4.1.1 Add-VaultChangeOrder

Creates a Change Order in Vault.

Syntax

```
Add-VaultChangeOrder -Number <String> [-Routing <String>] [<CommonParameters>]
```

Parameters

Type	Name	Description	Default value	Optional
String	Number	Number of the Change Order		no
String	Routing	The name of an active Routing that should be used for the Change Order.	The name of the default Routing Definition that is configured in Vault	yes

Return type

ChangeOrder ← on success
empty ← on failure

Remarks

An empty Change Order gets created that doesn't track any changes to design files yet and its *Due Date* will already be reached on the current day of creation.

Only routing entries from the list of active Routing Definitions can be passed to the **-Routing** parameter, and these are related to the default Workflow Definition that is configured in Vault and gets used for all the new Change Orders.

Examples

Create a new Change Order:

```
$changeOrder = Add-VaultChangeOrder -Number "ECO-0000022"
```

Display all the automatically assigned details of the new Change Order in Console:

```
$changeOrder = Add-VaultChangeOrder -Number "200-115"

#Print the Rounting, State, Due Date, Title, Detailed Description and the amount of
associated Item Records and File Attachments of the Change Order
$changeOrder | Format-Table Routing,_State,_ApproveDeadline,'_Title(Item,CO)','_
Description(Item,CO)',_NumItems,_NumFileAttachments

<#
Routing      _State _ApproveDeadline  _Title(Item,CO) _Description(Item,CO) _
NumItems _NumFileAttachments
-----
Production Change Create 11.05.2017 10:19:00
0 0
#>
```

Specifying a routing to be used for the created Change Order:

```
$changeOrder = Add-VaultChangeOrder -Number "ECO-000110" -Routing 'First Release'
```

4.1.2 Add-VaultFile

Adds a new file to Vault.

Syntax

```
Add-VaultFile -From <String> -To <String> [-Force <Bool>] [-Hidden <Bool>] [-Comment
<String>] [-FileClassification <String>] [<CommonParameters>]
```


Parameters

Type	Name	Description	Default value	Optional
String	From	Absolute Path to a local file		no
String	To	Absolute Vault path. It has to include the filename		no
Boolean	Force	If the target directory doesn't exist it will be created	True	yes
Boolean	Hidden	If true the file will not be displayed in the Vault Client	False	yes
String	Comment	A comment for the checkin	"Generated by coolOrange powerJobs"	yes
String	<i>FileClassification</i>	The classification of the file. See the table below	False	yes

File Classification

Name	Description
ConfigurationFactory	A template file (ex. iPart factories).
ConfigurationMember	A file created from a template (ex. iPart members).
DesignDocument	A design file containing data derived from a CAD file (ex. IDW or IPN)
DesignSubstitute	A design which is a substitute for a more complex design file.
DesignVisualization	A visualization file (ex. DWF). Although the Vault server will accept any file type as a visualization, many Vault clients will only work properly with DWF files.
Electrical-Project	A electrical CAD project file (ex. wdp).
None	The default file classification. This should be used for standard CAD files (ex. DWG, IPT, and IAM) and other types.

Return type

File ← on success
empty ← on failure

Remarks

The cmdlet uploads a local document to the Vault Server, and takes care about adding a new file or performing checkout- and checkin-operations.

There are three different situation on how the cmdlet behaves:

- When the target Vault-file **does not exist**, then it will be created with the local document
- When the target Vault-file **exist** and is **not checked out**, then the Vault-file will be checked-out and rechecked-in with the local document
- When the target Vault-file **exist** and is **checked out**, then the local document will be checked-in to Vault

When the target Vault-file already **exist** the cmdlet takes care about creating the new Vault-file version with the **exact same File associations** as the previous file version in Vault.

In other words, the File dependencies and attachments do not become manipulated at all!

Note: When calling Add-VaultFile with exact same parameters and the same file binary content as the latest file version, the Vault Server will not create a new version of the file, because it detects that nothing has changed!

However when changing only the -Comment, the server will also not create a new version.

Examples

Add File with comment

```
$file = Add-VaultFile -From "C:\Temp\pV_1.ipt" -To "$/PowerVaultTestFiles/pV_7.test" -  
↳Comment "Test"
```

Add hidden visualization file to Vault

```
$file = Add-VaultFile -From "C:\Temp\pV_1.dwf" -To "$/PowerVaultTestFiles/pV_7.test" -  
↳FileClassification "DesignVisualization" -Hidden $true
```

4.1.3 Add-VaultJob

Adds a Job to the Vault JobQueue.

Syntax

```
Add-VaultJob -Name <String> [-Parameters <Hashtable>] [-Description <String>] [-Priority
↪<int>] [<CommonParameters>]
```

Parameters

Type	Name	Description	Default value	Optional
String	Name	The name of job to add		no
Hashtable	Parameters	A hashtable of parameters for the job		yes
String	Description	A description of the job	“powerVault: added job”	yes
Integer	Priority	The priority of the job. Possible values are interger numbers or: <i>High, Medium</i> and <i>Low</i>	Medium	yes

Return type

Job ← on success

empty ← on failure

Remarks

The **Priority** argument can be specified by passing either a number or passing one of the following values: High, Medium, Low.

Passing a lower number means a higher priority. 1 is the lowest possible number.

High is equal to priority 1, *Medium* is equal to priority 10 and *Low* is equal to priority 20.

Examples

Adds a Job to the Vault JobQueue:

```
$job = Add-VaultJob -Name "Sample.CreatePDF" -Parameters @{"EntityId"]=88;"EntityClassId"=
↪"FILE"} -Description "Job for creating PDF"
```

Adds a Job with High Priority:

```
$job = Add-VaultJob -Name "TransferBOMToERP" -Parameters @{"EntityId"=$entity.Id;
↪"EntityClassId"=$entity._EntityTypeID} -Priority High
```

4.1.4 Get-VaultChangeOrder

Retrieves a specific Change Order from Vault.

Syntax

```
Get-VaultChangeOrder -Number <String> [<CommonParameters>]
```

```
Get-VaultChangeOrder -ChangeOrderId <Long> [<CommonParameters>]
```

Parameters

Type	Name	Description	Optional
String	Number	Number of the Change Order	no (optional when <i>ChangeOrderId</i> is used)
Long	ChangeOrderId	Id or MasterId of the Change Order	no (optional when <i>Number</i> is used)

Return type

ChangeOrder ← on success

empty ← on failure

Examples

Get Change Order via Number

```
$changeOrder = Get-VaultChangeOrder -Number 'ECO-00008'
```

Get Change Order via Id

```
$changeOrder = Get-VaultChangeOrder -ChangeOrderId 856
```

4.1.5 Get-VaultChangeOrderAssociations

Returns a collection of Change Order associations.

Syntax

```
Get-VaultChangeOrderAssociations -Number <String> -Type <ChangeOrderAssocType> [  
↪<CommonParameters>]
```

Parameters

Type	Name	Description	Optional
String	Number	Number of the Change Order	no
<i>ChangeOrderAssocType</i>	Type	Specifies which association type is retrieved	no

ChangeOrderAssocType

Name	Description
ItemRecords	Get the item record associations
FileRecords	Get the file record associations

Return type

Item[] / *File[]* ← on success
empty ← on failure

Remarks

The Cmdlet returns the **exact version** of the associations.

Examples

Get a Change Orders FileRecord associations

```
$files = Get-VaultChangeOrderAssociations -Number "ECO-100001" -Type FileRecords
```

Display all Item- and File-Record associations of a Change Order in console

```
$allAssocs = @( Get-VaultChangeOrderAssociations -Number "ECO-100001" -Type FileRecords;
Get-VaultChangeOrderAssociations -Number "ECO-100001" -Type ItemRecords)
$allAssocs | Format-Table Id, _EntityTypeID, Name, Revision, State

<#
Id  _EntityTypeID  Name           Revision  State
-----
15  FILE          Valve.ipt      2         Released
16  FILE          Pad_Lock.iam   2         Released
22  ITEM          1000002       1         Released
#>
```

4.1.6 Get-VaultFile

Retrieves a File from Vault.

It can also be used to download a file to a local directory.

Syntax

```
Get-VaultFile [-File <String>] [-FileId <Long>] [-Properties <Hashtable>] [-DownloadPath  
↪<String>] [<CommonParameters>]
```

Parameters

Type	Name	Description	Op- tional
String	File	Absolute Vault path to a file	yes
Long	FileId	Id or MasterId of the file	yes
Hashtable	Properties	Search for the file with matching properties	yes
String	Download- Path	This is the location where the file and all it's references will be down- loaded	yes

Return type

File ← on success

empty ← on failure

Remarks

If the search criteria's passed to the **-Properties** argument matches more than one file, then *only the first* file is returned. In case you want to get multiple files use the *Get-VaultFiles* Cmdlet.

In order to search a file by properties the **-Properties** argument allows to search the values of user defined properties and system properties which can be passed using their:

- *display names* (e.g. @{'Title' = ...} can be used with English Vault or @{'Titel' = ...} can be used with german Vault environments).

The **-DownloadPath** parameter allows downloading the file and all it's references into the specified directory.

One goal of this parameter is, that e.g. Inventor file can be opened without problems out from this directory.

The folder structure of the Vault files is maintained, and they will not be renamed.

Therefore the downloaded file is not always located directly within the specified folder and it could have been stored in a subfolder.

To retrieve the actual location, you can use the additional property `LocalPath`.

Note: For additional download options use the *Save-VaultFile* cmdlet.

Examples

Get File via absolute Vault path

```
$file = Get-VaultFile -File '$/PowerVaultTestFiles/pV_6.idw'
```

Get File that matches certain search criteria's

```
$file = Get-VaultFile -Properties @{"Description" = "TEMPLATE"; "Part Number" = "Dial"}
```

Downloading the file

```
$file = Get-VaultFile -File '$/Assemblies/Catch Assembly.iam' -DownloadPath "C:\Temp\
↳Test\Download"
$actualDownloadLocation = $file.LocalPath
```

If the directory “C:\Temp\Test\Download” does not exist, it will be created.

4.1.7 Get-VaultFileAssociations

Returns a collection of direct file relationships for the specified Vault file.

Syntax

```
Get-VaultFileAssociations -File <String> [-Attachments] [-Dependencies] [-Type_
↳FileAssociationType] [<CommonParameters>]
```

Parameters

Type	Name	Description	Default value	Optional
String	File	Full path to the Vault file		no
SwitchParameter	Attachments	Specifies that relationships of type <i>Attachment</i> will be returned		yes
SwitchParameter	Dependencies	Specifies that relationships of type <i>Dependency</i> will be returned		yes
<i>FileAssociationType</i>	Type	Specifies which kind and direction of associations should be retrieved	Children	yes

FileAssociationType

Name	Description
Children	Get all files used by the specified file
Drawings	Get only related <i>design documents</i> which use the specified file
Parents	Get all types of files which use the specified file

Return type

File[] ← on success

empty ← on failure

Remarks

The cmdlet returns all the files which can be linked in a parent/child or attachment relationship to the specified Vault file.

When no **-Dependencies** and **-Attachments** parameter is specified, then both types of file relationships are provided.

By default **Children** are returned, in fact in their *exact versions* which were used when the most recent version of the file was uploaded.

Alternatively all the **Parent** files, or only **Drawings**, can be retrieved which use the specified file.

Their *latest versions* are returned, regardless of whether they are using older versions of the specified file.

Examples

Display all the direct Uses of the Pad Lock.iam in Console

```

1 $file = Get-VaultFile -Properties @{Name = "Pad Lock.iam"}
2
3 $allDirectChildAssociations = Get-VaultFileAssociations -File $file._FullPath -Type_
  ↳ Children
4 $allDirectChildAssociations | Format-Table Name, Revision, 'State (Historical)'
5 <#
6 Name                      Revision State (Historical)
7 ----                      -
8 Retainer.ipt              B      Work in Progress
9 Combo Assembly.iam        A      Work in Progress
10 Catch Assembly.iam        A      Released
11 Case Back.ipt
12 Case Inner.ipt            A      Work in Progress
13 Case Outer.ipt            A      Work in Progress
14 Catch Post.ipt            1      Work in Progress
15 Catch.ipt                 1      Work in Progress
16 Combo Backing Plate.ipt   A      Work in Progress
17 Dial.ipt                  A      Work in Progress
18 Lock Shackle.ipt          A      Work in Progress
19 #>

```

Get the exact versions of used File Dependencies


```
$dependencies = Get-VaultFileAssociations -File '$/Libraries/Combo Plate iPart/Combo
↳Plate Lower.ipt' -Dependencies

$dependencies | Format-List _Name, _HasParentRelationship, _VersionNumber, _LatestVersion
<#
_Name                : Combo Plate iPart.ipt
_HasParentRelationship : True
_VersionNumber        : 1
_LatestVersion        : False
#>
```

Get only Design Visualization attachments of type PDF

```
$allAttachments = Get-VaultFileAssociations -File "$/Designs/Padlock/Assemblies/Combo
↳Assembly.idw" -Attachments

$pdfAttachments = $allAttachments | Where-Object { $_.Classification -eq "Design
↳Visualization" -and $_.Extension -eq "pdf"}
```

Display all the files where the Catch Assembly.iam is directly used

```
$allParentAssociations = Get-VaultFileAssociations -File '$/Designs/Padlock/Assemblies/
↳Catch Assembly.iam' -Type Parents

$allParentAssociations | Format-Table _FolderPath, _Name, _Classification, _
↳VersionNumber, _LatestVersion, IsCheckedOut
<#
_FolderPath          _Name                _Classification    _VersionNumber _
↳LatestVersion IsCheckedOut
-----
↳-----
$/Designs/Padlock/Assemblies Pad Lock.iam      None                16                ↳
↳ True            True
$/Designs/Padlock/Assemblies Catch Assembly.idw Design Document    5                ↳
↳ True            False
$/Designs/Padlock/Assemblies Catch Assembly.ipn Design Presentation 3                ↳
↳ True            False
#>
```

Get the associated Drawing which uses the specified model

```
$part = Get-VaultFile -Properties @{Name = "Case Back.ipt"}
if($part._HasDrawing) {
    $relatedDrawing = (Get-VaultFileAssociations -File $part._FullPath -Type Drawings)[0]
}
```

Get only Parent files which have a specific file attachment

```
$attachment = Get-VaultFile -Properties @{Name = "Spiral.xls"}

(Get-VaultFileAssociations -File $attachment._FullPath -Type Parents -Attachments) | ↳
↳Format-Table Name, _HasAttachments, _VisualizationAttachment
<#
```

(continues on next page)

(continued from previous page)

Name	_HasAttachments	_VisualizationAttachment
-----	-----	-----
<i>Fermat Spiral.ipt</i>	<i>True</i>	<i>None</i>
<i>Flexural Spiral.ipt</i>	<i>True</i>	<i>User</i>
<i>Hyperbolic Spiral.ipt</i>	<i>True</i>	<i>System</i>
#>		

4.1.8 Get-VaultFileBOM

Gets the Bill of Materials data from the CAD BOM of the specified file.

Syntax

```
Get-VaultFileBom -File <String> -GetChildrenBy <BomRowVersionType> [<CommonParameters>]
```

Parameters

Type	Name	Description	De- fault value	Op- tional
String	File	Full path to the file		no
<i>Bom- Model- State- Type</i>	Mod- el- State- Type	Indicates for which <i>model state</i> representation of the Inventor file the BOM should be retrieved. Possible options are <i>Master</i> , <i>All</i> or the name of the model state representation	Master	yes
<i>Bom- RowVer- sionType</i>	GetChil- drenBy	Which version of the children should be retrieved. Possible options are <i>ExactVersion</i> , <i>LatestVersion</i> , <i>LatestReleasedVersion</i> and <i>LatestReleasedVersionOfRevision</i>	ExactVer- sion	yes

Return type

FileBomRow[] ← on success

empty ← on failure

Remarks

The Cmdlet generates and provides the **Structured** file BOM and supports the following component types:

- **Normal:**
Components with type Normal have no special characteristics and are included in quantity calculations.
- **Phantom:**
Components with type Phantom are ignored in the BOM but their children are promoted and included in the quantity calculations.
Multiple same components can end up at the same level in the bill of materials when they get promoted.
They will end up as a single *FileBomRow* with summed quantities.

- **Reference:**

Components with type Reference and their children are ignored in the BOM and are treated as if they do not exist.

Read more about the different BOM structures [here](#).

For Inventor files with multiple [model states](#) the Bill of Materials of all the different design representations can be retrieved.

These can differ, for example, by suppressed components (which are not returned) or model state components with different part numbers.

Individual component instances which have different [Instance Properties](#) assigned are returned as single [FileBomRow](#) when the “Merge Instance Rows” option is enabled in Inventor 2023 or newer.

In situations where no CAD BOM information is available in Vault, the cmdlet behaves similar to the Vault “Assign Items” functionality.

For Inventor files, for example, the Structured BOM View should be activated, otherwise [empty position numbers and row orders](#) will be returned. Also, the cmdlet returns *empty* for *iAssembly* and *iPart factories*.

The different child component search strategies can be specified via **GetChildrenBy** option.

For strategies different than *ExactVersion*, it is possible that less [FileBomRows](#) are returned when they were removed in the detected Vault file version.

Like the Vault “Assign Items” functionality, also new components added in later Vault file versions are returned with empty position numbers and with row orders of 0.

The cmdlet handles files with *missing CAD BOMs*, BOMs that contain references to *unresolvable* Vault files or removed *model states* by returning all the intact and the corrupted [FileBomRows](#) with all the data it can provide. Accessing other properties can lead to exceptions.

BomModelStateType

Name	Description
Master	By default only the FileBomRows from the first model state entry, named <i>Master</i> , are returned.
All	All the available model states are returned, even if they have the same part number or delegated BOMs . Thus, such representations include also a <i>Master</i> model state component or components for automatically migrated Level Of Details or Substitute model states.
{ name of the model state }	Specify the name of a non-master model state (or Substitute model states) to retrieve its BOM representation.

Known issue with Model States and Vault Client 2022

When working with Inventor 2022 it is recommended to install the *Vault 2022.3 Update (Client)* or newer as several issues related to FileBOMs and Model States were addressed in the [Inventor Vault Add-in](#).

BomRowVersionType

Name	Description
ExactVersion	Get the exact version of the component.
LatestVersion	Get the latest version of the component.
LatestReleasedVersion	Get the latest released version of the component. If no released version is found, the exact version is used.
LatestReleasedVersionOfRevision	Get the latest released version from the same revision. If no released version is found in the Revision, the exact version is used.

Examples

Display the Structured file BOM for Pad Lock in Console

```
$file = Get-VaultFile -Properties @{Name = "Pad Lock.iam"}
$fileBom = Get-VaultFileBOM -File $file._fullPath

#Print BOM data as Table like in Inventor
$fileBom | sort-object {[int]$_._Bom_PositionNumber} | Format-Table Bom_PositionNumber,
    ↳ Bom_Number, Bom_Structure, Bom_Quantity, Bom_ItemQuantity, Bom_UnitQuantity, Bom_Unit

<#
Bom_PositionNumber Bom_Number Bom_Structure Bom_Quantity Bom_ItemQuantity Bom_
    ↳ UnitQuantity Bom_Unit
-----
↳ - -----
1                100002    Normal                1                1                ↳
↳ 1 Each
2                100004    Normal                1                1                ↳
↳ 1 Each
3                100005    Normal                1                1                ↳
↳ 1 Each
4                100006    Normal                1                1                ↳
↳ 1 Each
5                100008    Normal                1                1                ↳
↳ 1 Each
6                100011    Normal                1                1                ↳
↳ 1 Each
7                100010    Normal                1                1                ↳
↳ 1 Each
8                100003    Normal                1                1                ↳
↳ 1 Each
9                100012    Normal                1                1                ↳
↳ 1 Each
10               100016    Normal                1                1                ↳
↳ 1 Each
11               100009    Normal                1                1                ↳
↳ 1 Each
#>
```

Get the latest released file BOM data of assembly which includes new components in Phantom

```

$fileBom = Get-VaultFileBOM -File "$/Designs/AssemblyWithPhantom.iam" -GetChildrenBy_
↳LatestReleasedVersion

#Print BOM data of latest released component versions, including the promoted Phantom_
↳components
$fileBom | Format-Table Bom_RowOrder,Bom_PositionNumber,Bom_Number,Bom_Structure,Bom_
↳Quantity,Bom_ItemQuantity,Bom_UnitQuantity,Bom_Unit
<#
Bom_RowOrder Bom_PositionNumber Bom_Number Bom_Structure Bom_Quantity Bom_ItemQuantity_
↳Bom_UnitQuantity Bom_Unit
-----
↳ -----
0                               NewPart    Normal                1                1  ↳
↳
↳           1 Each
2           5                Part2      Normal                2                2  ↳
↳
↳           1 Each
#>

```

Retrieve the file BOM for all assembly model states

```

$file = Get-VaultFile -File '$/Designs/caster_w_MS.iam'
if($file._HasModelState -eq $null -or $file._HasModelState -ne $true) {
    Write-Host "File $($file._Name) is not an Inventor file or has only one Master_
↳model state!"
    return
}

$allModelStates = Get-VaultFileBOM -File '$/Designs/caster_w_MS.iam' -ModelStateType All

foreach($ModelState in $allModelStates) {
    Write-Host "$($ModelState._Name) ($($ModelState.Bom_ModelState)): $($ModelState.
↳'Bom_Part Number')" -NoNewline

    $fileBom = Get-VaultFileBOM -File '$/Designs/caster_w_MS.iam' -ModelStateType
↳$ModelState.Bom_ModelState
    $fileBom | Format-Table Bom_RowOrder,Bom_PositionNumber,Bom_Number,Bom_Structure,
↳Bom_ModelState,Bom_Quantity,Bom_ItemQuantity,Bom_UnitQuantity,Bom_Unit
    Write-Host ""
}

<#
caster_w_MS.iam (Master): FP-1000
Bom_RowOrder Bom_PositionNumber Bom_Number      Bom_Structure Bom_ModelState Bom_
↳Quantity Bom_ItemQuantity Bom_UnitQuantity Bom_Unit
-----
↳ -----
1           1                top plate    Normal        Master                ↳
↳1           1                1 Each
2           2                support arm   Normal        Simplified            ↳
↳2           2                1 Each
3           3                HEX FLANGE SREW Normal        Welded              ↳
↳4           4                1 Each
4           4                bushing     Normal        Master                ↳

```

(continues on next page)

(continued from previous page)

```
↪2          2          1 Each

...
caster_w_MS.iam (Stage2): FP-3000
Bom_RowOrder Bom_PositionNumber Bom_Number      Bom_Structure Bom_ModelState Bom_
↪Quantity Bom_ItemQuantity Bom_UnitQuantity Bom_Unit
-----
↪- -----
1          1          top plate      Normal      Master
↪1          1          1 Each
2          2          support arm    Normal      Simplified
↪2          2          1 Each
4          4          bushing        Normal      Face machining
↪1          1          1 Each
#>
```

Skip BomRows of unconditionally removed or erroneous file components

```
$fileBom = ( Get-VaultFileBOM -File "$/Designs/CorruptFileBom.iam" -GetChildrenBy_
↪LatestVersion ) | foreach {
    try {
        $successfullyRetrievedBomNumber = $_ | Select-Object -ExpandProperty Bom_
↪Number
        return $_
    } catch {
        # Skip unconditionally removed or erroneous file component, similar as_
↪Vault "Assign Items" functionality
    }
}
```

4.1.9 Get-VaultFiles

Returns an array of file objects that match your parameters.

Syntax

```
Get-VaultFiles [-FileName <String>] [-Folder <String>] [-Properties <Hashtable>] [
↪<CommonParameters>]
```

Parameters

Type	Name	Description	Optional
String	FileName	Name of a file in Vault	yes
String	Folder	Absolute path to a Vault folder	yes
Hashtable	Properties	Search for files with matching properties	yes

Return type

File[] ← on success

empty ← on failure

Remarks

If you want a single specific file use *Get-Vaultfile*.

In order to search files by properties the **-Properties** argument allows to search the values of user defined properties and system properties which can be passed using their:

- *display names* (e.g. @{'Title' = ...} can be used with English Vault or @{'Titel' = ...} can be used with german Vault environments).

Examples

Get all files from the specified Vault folder

```
$files = Get-VaultFiles -Folder '$/Designs/2014/03/0'

$files = Get-VaultFiles
foreach($file in $files){
    $file.'File Name'
}
```

4.1.10 Get-VaultItem

Retrieves an Item from Vault.

Syntax

```
Get-VaultItem [-Number <String>] [-ItemId <Long>] [-Properties <Hashtable>] [
    ↳<CommonParameters>]
```

Parameters

Type	Name	Description	Optional
String	Number	Number of the item	yes
Long	ItemId	Id or MasterId of the item	yes
Hashtable	Properties	Search for the item with matching properties	yes

Return type

Item ← on success
empty ← on failure

Remarks

If the search criteria's passed to the **-Properties** argument matches more than one item, then *only the first* item is returned.

In order to search an item by properties the **-Properties** argument allows to search the values of user defined properties and system properties which can be passed using their:

- *display names* (e.g. @{ 'Effectivity' = ... } can be used with English Vault or @{ 'Gültigkeit' = ... } can be used with german Vault environments).

Examples

Get Item via Number

```
$item = Get-VaultItem -Number '100001'
```

Get Item via Id

```
$item = Get-VaultItem -ItemId 142
```

Get Item that matches certain search criteria's

```
$item = Get-VaultItem -Properties @{"Description (Item,CO)" = "PAD LOCK ASSEMBLY";  
↪ "Lifecycle Definition" = "Item Release Process"}
```

4.1.11 Get-VaultItemAssociations

Syntax

Returns a collection of the item associations.

```
Get-VaultItemAssociations -Number <String> [-Attachments] [-Primary] [-Secondaries] [-  
↪ Tertiaries] [-StandardComponents] [<CommonParameters>]
```


Parameters

Type	Name	Description	Optional
String	Number	Number of the item	no
SwitchParameter	Attachments	If the SwitchParameter is set only attachments will be returned	yes
SwitchParameter	Primary	If the SwitchParameter is set only the primary will be returned	yes
SwitchParameter	Secondaries	If the SwitchParameter is set only secondaries will be returned	yes
SwitchParameter	Tertiaries	If the SwitchParameter is set only tertiaries will be returned	yes
SwitchParameter	StandardComponents	If the SwitchParameter is set only standard components will be returned	yes

Return type

File[] ← on success

empty ← on failure

Remarks

The cmdlet returns the **exact version** of the associated *files* of an *item*.

When no SwitchParameter is set, the cmdlet will return all types of associations for the item.

Examples

Get all associations

```
$allAssocs = Get-VaultItemAssociations -Number 100001
```

Get attachments only

```
$attachments = Get-VaultItemAssociations -Number 100001 -Attachments
```

4.1.12 Get-VaultItemBOM

Gets the Bill of Materials data for the item.

Syntax

```
Get-VaultItemBOM -Number <String> [-IncludeInactive] [-IncludeUnassigned] [-Recursive] [
↳<CommonParameters>]
```

Parameters

Type	Name	Description	Optional
String	Number	Number of the item	no
SwitchParameter	IncludeInactive	If the SwitchParameter is set also BOM rows which are set to inactive in vault are included	yes
SwitchParameter	IncludeUnassigned	If the SwitchParameter is set also BOM rows which dont have an associated item in vault are included	yes
SwitchParameter	Recursive	If the SwitchParameter is set all child BOM rows are included	yes

Return type

ItemBomRow[] ← on success

empty ← on failure

Examples

Get Item BOM and view data in Console

```
$itemBom = Get-VaultItemBOM -Number '100001'

#Print BOM data as Table like in Vault
$itemBom | sort-object {[int]$_.Bom_RowOrder} | Format-Table Bom_Number,Bom_RowOrder,Bom_
↳PositionNumber,Bom_Quantity,Bom_Unit,'_Title(Item,CO)',Bom_IsCad

<#
Bom_Number      Bom_RowOrder Bom_PositionNumber Bom_Quantity Bom_Unit  _Title(Item,CO)  ↳
↳      Bom_IsCad
-----
↳      -----
100002              1 1              1 Each      Combo Assembly.  ↳
↳iam              True
100004              2 2              1 Each      Catch Post.ipt   ↳
↳              True
100005              3 3              1 Each      Lock Shackle.ipt ↳
↳              True
100006              4 4              1 Each      Case Inner.ipt   ↳
↳              True
100008              5 5              1 Each      Case Outer.ipt   ↳
↳              True
100011              6 6              1 Each      Case Back.ipt    ↳
↳              True
```

(continues on next page)

(continued from previous page)

100010		7 7	1 Each	Dial.ipt	└
↪	True				
100003		8 8	1 Each	Catch Assembly.	
↪ iam	True				
100012		9 9	1 Each	Catch.ipt	└
↪	True				
100016		10 10	1 Each	Combo Backing	└
↪ Plate.ipt	True				
100009		11 11	1 Each	Retainer.ipt	

Gets complete Item BOM including inactive and unassigned rows

```
$itemBom = Get-VaultItemBOM -Number '100001' -IncludeInactive -IncludeUnassigned -
↪Recursive
```

4.1.13 Open-VaultConnection

Connects to the specified Vault Server.

Syntax

```
Open-VaultConnection [-Vault <String>] [-Server <String>] [-User <String>] [-Password
↪<String>] <CommonParameters>]
```

Parameters

Type	Name	Description	Default value	Optional
String	Vault	The name of the Vault for which the connection should be made	Vault	yes
String	Server	The Ip or hostname of the server on which the Vault server is installed	localhost	yes
String	User	A valid user, that is able to login to the target Vault	Administrator	yes
String	Password	The users password	""	yes

Return type

empty ← On failure the Exception/ErrorMessage can be accessed using `$Error`.

Remarks

After the Cmdlet successfully established a connection to Vault, the following PowerShell variables are created and let you communicate directly to Vault through it's API:

- *\$vault*
- *\$vaultconnection*
- *\$vaultExplorerUtil*

The cmdlet can be invoked **without arguments** in applications that are already connected to Vault (for example within the Vault Explorer), so that the mentioned PowerShell variables are exposed for the *existing Vault connection*.

4.1.14 Save-VaultFile

Downloads a file from Vault to a local directory.

Syntax

```
Save-VaultFile -File <String> [-DownloadDirectory <String>] [-Version  
↪<VersionGatheringOption>] [-ExcludeChildren] [-NoRecursiveChildren] [-NotOverride] [-  
↪IncludeAttachments] [-IncludeHiddenEntities] [-IncludeParents] [-RecursiveParents] [-  
↪IncludeRelatedDocumentation] [<CommonParameters>]
```

Parameters

Type	Name	Description	Default value	Optional
String	File	Path to the file, which should be downloaded		no
String	DownloadDirectory	Absolute Path to a local file. If NOT set, it will download the file to the vault workspace.If the target directory doesn't exist it will be created		yes
Enum	Version	Version of the related files to be downloaded.Possible values are: <i>Latest</i> , <i>Actual</i> and <i>Revision</i>	Latest	yes
Switch-Parameter	ExcludeChildren	Children of the source file will be excluded		yes
Switch-Parameter	NoRecursiveChildren	Only the first level of children will be included		yes
Switch-Parameter	ExcludeLibraries	Entities that are a part of a Vault Library will be excluded		yes
Switch-Parameter	NotOverride	Folder/Files will not be overwritten, if they already exists		yes
Switch-Parameter	IncludeAttachments	Attachments of the file will be included		yes
Switch-Parameter	IncludeHiddenEntities	Hidden entities will be included		yes
Switch-Parameter	IncludeParents	Parents of the source file will be included		yes
Switch-Parameter	RecursiveParents	Parents will be included recursively		yes
Switch-Parameter	IncludeRelatedDocumentation	The related documentation of the source file will be included.For example, a DWG might be considered related documentation for an assembly. The DWG is not considered an attachment or a dependent		yes

Return type

File[] ← on success

empty ← on failure. Exception/ErrorMessage can be accessed using *\$Error*.

Remarks

The cmdlet downloads a file from the Vault Server to a local directory.

The result of the cmdlet contains a list of all the downloaded files.

Each of those files have an additional property '*LocalPath*' containing the full path (including the file name) of the downloaded file.

Examples

Download a File

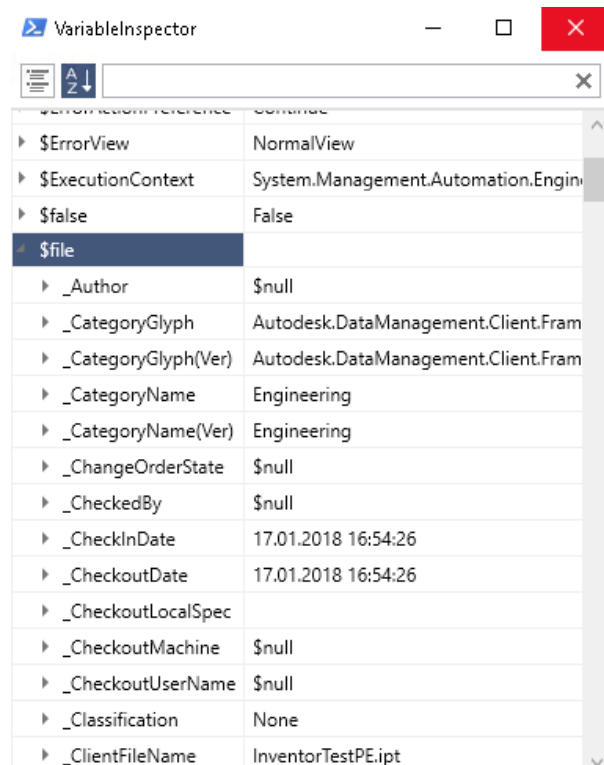
```
$result = Save-VaultFile -File "$/Designs/Sample/Not.iam" -DownloadDirectory "C:\temp\
Vault\Sample\"
$result[0].LocalPath #Returns C:\temp\Vault\Sample\Not.iam
```

Download a File to the vault workspace including related documentation files

```
$result = Save-VaultFile -File "$/Designs/Sample/Not.iam" -IncludeRelatedDocumentation
```

4.1.15 Show-Inspector

Opens the Inspector window, a debugging tool that displays all existing PowerShell variables and their values:



Syntax

```
Show-Inspector [-Highlight <String>] [<CommonParameters>]
```

Parameters

Type	Name	Description	Optional
String	Highlight	The inspector jumps to the passed variable (e.g. 'file')	yes

Return type

empty

Examples

Extend PowerShell script with Show-Inspector to analyse variables and find errors:

```
function Get-Path{
    param($switch)
    switch($switch){
        a{$path = "C:\Temp\a.txt";break}
        b{$path = "C:\Temp\b.txt";break}
        c{$path = "C:\Temp\.txt";break}
        d{$path = "C:\Temp\d.txt";break}
    }
    return $path
}

$path = Get-Path -switch 'c'

Show-Inspector 'path'

Get-Content $path
```

Admittedly the error in the previous script is obvious, but it's just intended to show a possible purpose of the Inspector window.

In more complex scripts you can easily check your PowerShell variables with this method without writing them out to a textfile.

4.1.16 Update-VaultChangeOrder

Edits a Change Order in Vault and manipulates its associations.

Syntax

```
Update-VaultChangeOrder -Number <String> [-DueDate <DateTime>] [-Title <String>] [-
↪Description <String>] [-Properties <Hashtable>] [-AddItemRecords <String[]>] [-
↪ItemRecords <String[]>] [-RemoveItemRecords <String[]>] [-AddAttachments <String[]>] [-
↪Attachments <String[]>] [-RemoveAttachments <String[]>] [<CommonParameters>]
```

Return type

ChangeOrder ← on success

empty ← on failure

Parameters

Type	Name	Description	Op-tional
String	Number	Number of the Change Order that should be edited	no
Date-Time	DueDate	New date and time by which the Change Order must be approved.	yes
String	Title	The new title of the Change Order	yes
String	Description	The new description of the Change Order	yes
Hashtab	Properties	The user-defined properties and their values which should be updated on the Change Order	yes
String[]	AddItem-Records	Vault items - identified by their Number - that should be associated as new Item Records of the Change Order	yes
String[]	ItemRecords	Vault items - identified by their Number - that will replace the existing Item Records of the Change Order	yes
String[]	RemoveItem-Records	Vault items - identified by their Number - which should be removed from the Item Records of the Change Order	yes
String[]	AddAttach-ments	Full paths to the Vault files that should be attached to the Change Order	yes
String[]	Attachments	Full paths to the Vault files that will replace the existing attachments of the Change Order	yes
String[]	RemoveAt-tachments	Full paths to the Vault files which should be detached from the Change Order	yes

Remarks

In order to update properties of a Change Order the **-Properties** argument allows editing the values of used-defined properties which can be passed using their *display names*.

The **-AddItemRecords** parameter can be used to **add Vault items** to a Change Order by passing an array of Vault item numbers.

All the existing Item Records that are part of the change management process can even be replaced using the **-ItemRecords** parameter and individual Vault item associations can be removed again using the **-RemoveItemRecords** parameter.

Vault files can be **attached** to a Change Order using the **-AddAttachments** parameter, by passing an array of Vault file paths.

In the same way, all the existing attachment associations can be replaced using the **-Attachments** parameter or individual Vault files can be detached using **-RemoveAttachments**.

When some of the attachments are checked-out, then powerVault will attach the *previous file version* of them.

Examples

Edit the details of a previously created Change Order:

```
$changeOrder = Add-VaultChangeOrder -Number "ECO-000110"

$changeOrderWithUpdatedDetails = Update-VaultChangeOrder -Number $changeOrder._Number `
    -DueDate (Get-Date).AddDays(10) `
    -Title "High Welding Frame adaption" `
    -Description "Please copy this design to create a new High Welding Frame.`r`nH = 2100mm`r`nW = 600mm`r`nD = 600mm" `
    -Properties @{'Test' = 6.66}
```

Assign Item Records to a Change Order:

```
$item = Get-VaultItem -Number '100001'
$allRelatedItemRecords = @($item) + (Get-VaultItemBOM -Number $item.Number)

$changeOrder = Update-VaultChangeOrder -Number 'ECO-0000022' -ItemRecords (
    ↳ $allRelatedItemRecords | Select-Object -ExpandProperty Number)
$changeOrder.'Number of Items' #Returns the amount of assigned Item Records
```

Attach a Markup file to a Change Order

```
$changeOrder = Update-VaultChangeOrder -Number 'ECO-000138' -AddAttachments @('$/ECO_
    ↳ Markup DWF/ECO-000138 Markup coolOrange.dwf')

$changeOrder._HasAttachments #Returns True
$changeOrder.'Number of File Attachments' #Returns the amount of attached files
```

4.1.17 Update-VaultFile

Updates a File in Vault and manipulates its associations.

Syntax

```
Update-VaultFile -File <String> [-Comment <String>] [-Category <String>] [-Status
    ↳ <String>] [-LifecycleDefinition <String>] [-Revision <String>] [-RevisionDefinition
    ↳ <String>] [-Properties <Hashtable>] [-AddChilds <String[]>] [-Childs <String[]>] [-
    ↳ RemoveChilds <String[]>] [-AddAttachments <String[]>] [-Attachments <String[]>] [-
    ↳ RemoveAttachments <String[]>] [<CommonParameters>]
```

Parameters

Type	Name	Description	Optional
String	File	Full path to the Vault file, that should be updated	no
String	Comment	The comment associated with the new Vault file version	yes
String	Category	The name of the new categories that should be assign to the Vault file	yes
String	Status	The name of the new lifecycle state for the Vault file	yes
String	LifecycleDefinition	The name of the new lifecycle definition that should be assigned to the Vault file	yes
String	Revision	The new revision number of the Vault file	yes
String	RevisionDefinition	The name of the revision definition that should be assigned to the Vault file	yes
Hashtable	Properties	The user-defined properties and their values which should be updated on the Vault file	yes
String	AddChilds	Full paths to the Vault files that should be associated as new children of the Vault file	yes
String	Childs	Full paths to the Vault files that will replace the existing children of the Vault file	yes
String	RemoveChilds	Full paths to the Vault files which should be removed from the children of the Vault file	yes
String[]	AddAttachments	Full paths to the Vault files that should be attached to the Vault file	yes
String[]	Attachments	Full paths to the Vault files that will replace the existing attachments of the Vault file	yes
String[]	RemoveAttachments	Full paths to the Vault files which should be detached from the Vault file	yes

Return type

File ← on success
empty ← on failure

Remarks

Unlike in the Vault VDF there are no Ids needed for the Update-VaultFile parameters. The Ids are handled by powerVault.

- To refer to a file you need to pass an absolute Vault path (e.g. "\$/Design/powerJobsTest/pJ_5.ipt").
- For categories, states, etc. you need to pass their names (e.g. "Engineering", "Work in Progress")

If you want to update the **RevisionDefinition** you should care about the current Revision on this file. So, if you change it to a RevisionDefinition it almost always requires a value depending on this revision.

Almost the same behaviour applies to the **LifeCycleDefinition**: when assigning a new LifeCycleDefinition and the current state of the file does not exist in the new LifeCycleDefinition, than the **Status** must be updated as well.

In order to update properties of a Vault file the **-Properties** argument allows updating the values of used-defined properties which can be passed using their:

- *system names* are prefided with '_' and can be used with all Vault language environments (e.g. @{'_Title' = ...})

- *display names* (e.g. @{'Title' = ...} can be used with English Vault or @{'Titel' = ...} can be used with german Vault environments).

Vault files can be associated as children to a Vault file using the **-AddChilds** parameter, by passing an array with of Vault file paths.

In the same way even Vault files can be attached using the **-AddAttachments** parameter.

All the existing child associations and attachments can be replaced using the **-Child** and **-Attachments** parameters.

Individual Vault files can be detached using the **-RemoveAttachments** parameter or existing child associations can be removed using **-RemoveChilds**.

When some existing references are checked-out, then powerVault will check-in the *previous file version* of them.

Warning: You can only make changes to the files you could do with the Vault Client.
For example it is not possible to change the state of a file to something that isn't defined in the lifecycle definition.

Examples

Update the lifecycle state and definition of a file in Vault:

```
$file = Get-VaultFile -File "$/Designs/Pad Lock.iam"

$updated = Update-VaultFile -File $file.'Full Path' -LifecycleDefinition "Flexible_
↳Release Process" -Status "Work in Progress"
```

Update the revision number and definition of a Vault file:

```
$file = Get-VaultFile -File "$/Designs/Pad Lock.iam"

$updated = Update-VaultFile -File $file.'Full Path' -Revision "B" -RevisionDefinition
↳"Standard Alphabetic Format" -Comment "Revision updated"
```

Assign attachments of a Vault file:

```
$parent = Get-VaultFile -File "$/Designs/pV_4.iam" $child = Add-VaultFile -From "C:\Temp\
↳pV_1.ipt" -To "$/Designs/pV_4_3.ipt"
$parentUpdated = Update-VaultFile -File

$parent.'Full Path' -Attachments @($child.'Full Path') -Comment "Set one attachment only"
```

Remove child dependencies from a Vault file:

```
$parent = Get-VaultFile -File "$/Designs/pV_2.iam"
$child1 = Get-VaultFile -File "$/Designs/pV_2_1.ipt"
$child2 = Get-VaultFile -File "$/Designs/pV_2_2.ipt"
$child3 = Get-VaultFile -File "$/Designs/pV_2_3.ipt"

$parentUpdated = Update-VaultFile -File $parent.'Full Path' -RemoveChilds @($child3.
↳'Full Path') -Comment "Child 3 removed"
```

4.1.18 Update-VaultItem

Updates an Item in Vault and manipulates its associations.

Syntax

```
Update-VaultItem -Number <String> [-NewNumber <String>] [-Title <String>] [-Description
↪<String>] [-Properties <Hashtable>] [-AddAttachments <String[]>] [-Attachments
↪<String[]>] [-RemoveAttachments <String[]>] [<CommonParameters>]
```

Parameters

Type	Name	Description	Optional
String	Number	Number of the Item, that should be updated	no
String	NewNumber	New Number that renames the Item by using the ‘ <i>Mapped</i> ’ Numbering Scheme. This scheme allows entering free item names.	yes
String	Title	The new title of the Item	yes
String	Description	The new description of the Item	yes
Hashtable	Properties	The user-defined properties and their values which should be updated on the Item	yes
String[]	AddAttachments	Full paths to the Vault files that should be attached to the Item	yes
String[]	Attachments	Full paths to the Vault files that will replace the existing attachments of the Item	yes
String[]	RemoveAttachments	Full paths to the Vault files which should be detached from the Item	yes

Return type

Item ← on success

empty ← on failure

Remarks

In order to update properties of an Item the **-Properties** argument allows updating the values of user-defined properties which can be passed using their:

- *system names* are prefixed with ‘_’ and can be used with all Vault language environments (e.g. @{'_ItemEffectivity' = ...})
- *display names* (e.g. @{'Effectivity' = ...} can be used with English Vault or @{'Gültigkeit' = ...} can be used with German Vault environments).

Vault files can be attached to an Item using the **-AddAttachments** parameter, by passing an array of Vault file paths. In the same way, all the existing attachment associations can be replaced using the **-Attachments** parameter and individual Vault files can be detached using **-RemoveAttachments**.

When some existing attachments are checked-out, then powerVault will check-in the *previous file version* of them.

Warning: You can only make changes to the items you could do with the Vault Client.

Examples

Rename an Item in Vault:

```
$item = Update-VaultItem -Number '100017' -NewNumber '111111'
```

Update the details of a Vault Item:

```
$item = Update-VaultItem -Number '100017' `
    -Title "FINAL PART" `
    -Description "COMBO STAND OFF" `
    -Properties @{'Test'='Updated'}
```

Assign attachments to a Vault Item:

```
$parent = Get-VaultItem -Number 100001 $attachment = Get-VaultFile -File "$/Designs/ABC.
↪ipt"

$parentUpdated = Update-VaultItem -Number $parent.Number -Attachments @($attachment.
↪'Full Path')
```

Detach files from a Vault Item:

```
$item = Get-VaultItem -Number 100001 $attachment = Get-VaultFile -File "$/Designs/ABC.ipt
↪"

$itemUpdated = Update-VaultItem -Number $item.Number -RemoveAttachments @($attachment.
↪'Full Path')
```

Attach a new file to a Vault Item:

```
$item= Get-VaultItem -Number 100001 $attachment= Add-VaultFile -From "C:\Temp\ABC.ipt" -
↪To "$/Designs/ABC.ipt"

$itemUpdated = Update-VaultItem -Number $item.Number -AddAttachments @($attachment.'Full
↪Path')
```

4.1.19 Connection

Cmdlets related to the Vault connection.

Name	Description
<i>Open-VaultConnection</i>	Connects to a Vault server.

4.1.20 Debug

Cmdlets to help identifying issues in scripts.

Name	Description
<i>Show-Inspector</i>	Opens a dialog in which all available PowerShell variables and their values are displayed.

Note: The following cmdlets require an existing Vault connection in the application being used to work with them.

4.1.21 Files

Cmdlets for adding, updating and retrieving Vault File(s) and their associations.

Name	Description
<i>Get-VaultFile</i>	Retrieves a File from Vault.
<i>Get-VaultFiles</i>	Retrieves multiples File from Vault.
<i>Get-VaultFileAssociations</i>	Retrieves the associations of a File in Vault.
<i>Get-VaultFileBom</i>	Retrieves the Bill of Material data of a of a File in Vault.
<i>Add-VaultFile</i>	Adds a new File to the Vault.
<i>Update-VaultFile</i>	Updates an existing File in Vault.
<i>Save-VaultFile</i>	Download a File from Vault to a local directory.

4.1.22 Items

Cmdlets for updating and retrieving Vault Item(s) and their associations.

Name	Description
<i>Get-VaultItem</i>	Retrieves an Item from Vault.
<i>Get-VaultItemAssociations</i>	Retrieves the associations of an Item in Vault.
<i>Get-VaultItemBom</i>	Retrieves the Bill of Material data of a of an Item in Vault.
<i>Update-VaultItem</i>	Updates an existing Item in Vault.

4.1.23 Change Orders

Cmdlets for adding, updating and retrieving Change Order(s) and their associations.

Name	Description
<i>Get-VaultChangeOrder</i>	Retrieves a Change Order from Vault.
<i>Get-VaultChangeOrderAssociations</i>	Retrieves the associations of a Change Order from Vault.
<i>Add-VaultChangeOrder</i>	Creates a Change Order in Vault.
<i>Update-VaultChangeOrder</i>	Updates an existing Change Order in Vault.

4.1.24 Jobs

Cmdlets related to Jobs and the Vault JobQueue.

Name	Description
<i>Add-VaultJob</i>	Adds a new Job to the Vault JobQueue.

4.2 Objects

4.2.1 Change Order

The ChangeOrder object is of type PsObject and represents a Change Order item in Vault.

The \$changeOrder object is dynamically generated based on the defined ChangeOrder Properties in Vault.

The properties are named the same as in Vault, including whitespaces. If you want to access such a property you have to enclose it in single quotes.

Syntax

```
$changeOrder._Number
```

The following properties are always added :

Type	Name	Description
long	Id	The unique identifier for the object.
string	Routing	The unique name of the Routing that the Change Order was created with.
Uri	ThinClientHyperLink	The unique Hyperlink to the Change Order, that redirects to the Autodesk Thin-Client
Uri	ThickClientHyper-Link	The unique Hyperlink to the Change Order, that redirects to Autodesk Vault Client

Localization

\$changeOrder is supporting this *feature*:

- all active properties can be accessed via the displayName
- all active properties can be accessed via _SystemName

except: the ones with guids (=user defined)

Examples

Accessing 'Number of Items' property for different Vault languages

```

1 $changeOrder.'Number of Items' #can be used with English Vault
2 $changeOrder.'Anzahl Artikel' #can be used with German Vault
3 $changeOrder._NumItems #can be used with all Vault language environments

```

ChangeOrder-object on an english environment:

```

1 Created By : coolOrange
2 _CreateUserName : coolOrange
3 Date Modified : 30.11.2017 13:03:58
4 _ModDate : 30.11.2017 13:03:58
5 Name : ECO-0000022
6 _Name : ECO-0000022
7 State : Review
8 _State : Review
9 Last Updated By : coolOrange
10 _LastModifiedUserName : coolOrange
11 Number : ECO-0000022
12 _Number : ECO-0000022
13 Title (Item,CO) : New Variation - High Welding Frame
14 _Title(Item,CO) : New Variation - High Welding Frame
15 Description (Item,CO) : Please copy this design to create a new High Welding Frame.
16 H = 2100mm
17 W = 600mm
18 D = 600mm
19 A = 60 deg
20 _Description(Item,CO) : Please copy this design to create a new High Welding Frame.
21 H = 2100mm
22 W = 600mm
23 D = 600mm
24 A = 60 deg
25 Due Date : 11.05.2018 10:19:00
26 _ApproveDeadline : 11.05.2018 10:19:00
27 Submitted By : coolOrange
28 _SubmitUser : coolOrange
29 Date Submitted : 30.11.2017 13:03:07
30 _SubmitDate : 30.11.2017 13:03:07
31 Date Created : 08.05.2017 10:27:39
32 _CreationDate(CO) : 08.05.2017 10:27:39
33 Number of File Attachments : 0
34 _NumFileAttachments : 0
35 Number of Items : 1
36 _NumItems : 1
37 Entity Type : Change Order
38 _EntityType : Change Order
39 Entity Type ID : CO
40 _EntityTypeID : CO
41 Entity Icon : Autodesk.DataManagement.Client.Framework.Vault.Currency.
42 ↪Properties.ImageInfo
43 _EntityIcon : Autodesk.DataManagement.Client.Framework.Vault.Currency.
44 ↪Properties.ImageInfo

```

(continues on next page)

(continued from previous page)

```

43 Path :
44 _EntityPath :
45 Full Path :
46 _FullPath :
47 Attachments : False
48 _HasAttachments : False
49 Link Target Path :
50 _LinkTargetPath :
51 Date Modified (Date Only) : 30.11.2017 00:00:00
52 Date Modified (Time Only) : 01.01.0001 13:03:00
53 Due Date (Date Only) : 11.05.2017 00:00:00
54 Due Date (Time Only) : 01.01.0001 10:19:00
55 Date Submitted (Date Only) : 30.11.2017 00:00:00
56 Date Submitted (Time Only) : 01.01.0001 13:03:00
57 Date Created (Date Only) : 08.05.2017 00:00:00
58 Date Created (Time Only) : 01.01.0001 10:27:00
59 Type Description : Change Order
60 _EntityDescription : Change Order
61 Vault Status : Autodesk.DataManagement.Client.Framework.Vault.Currency.
   ↳ Properties.EntityStatusImageInfo
62 _VaultStatus : Autodesk.DataManagement.Client.Framework.Vault.Currency.
   ↳ Properties.EntityStatusImageInfo
63 Id : 101368
64 MasterId : 101368
65 PersistentId :
66 PersistentMasterId :
67 Routing : Production Change
68 ThinClientHyperLink : http://localhost/AutodeskTC/Vault/changeorders/changeorder/
   ↳ 101368
69 ThickClientHyperLink : http://localhost/AutodeskDM/Services/
   ↳ EntityDataCommandRequest.aspx?Vault=Vault&ObjectId=ECO-0000022&ObjectType=ECO&
   ↳ Command=Select

```

4.2.2 Comment

A Comment object is of type *PsObject* and represents a comment used in Vault *Change Orders*.

The \$comment object is dynamically generated based on the properties which are fixed by the Vault client. Therefore all properties are directly available on this object.

The properties are named the same as in Vault, including whitespaces. If you want to access such a property you have to enclose it in single quotes.

Syntax

```
1 $comment.Message
```

The following properties are always added :

Type	Name	Description
string[]	Attachments	Array of the files (the FullPath) attached to the comment.

Localization

\$powerVaultComment is supporting this *feature*:

- all active properties can be accessed via the displayName
- all active properties can be accessed via _SystemName

except: the ones with guids (=user defined)

Examples

Comment-object on an english environment:

```

1 Created By           : Administrator
2 _CreateUserName      : Administrator
3 State               : Create
4 _State              : Create
5 Subject (CO)        : We require a fire heater
6 _ForumMessageSubject : We require a fire heater
7 Message             : The client requires a fire heater as well now!
8 _ForumMessageText    : The client requires a fire heater as well now!
9 Created             : 5/30/2017 10:16:53 AM
10 _ForumMessageCreateDate : 5/30/2017 10:16:53 AM
11 Created (Date Only)  : 5/30/2017 12:00:00 AM
12 Created (Time Only)  : 1/1/0001 10:16:00 AM
13 Id                  : 15272
14 MasterId            : 0
15 PersistentId         :
16 PersistentMasterId   :
17 Attachments          : {$/Designs/autodesk_inventor_samples/Models/AEC Exchange/Water_
↵ Heater/Water Heater.ipt, $/Designs/autodesk_inventor_samples/Models/AEC Exchange/Water_
↵ Heater/Water Heater.idw}
```

4.2.3 Custom Object

A CustomObject object is of type *PsObject* and represents a Vault Custom Entity.

The \$customObject object is dynamically generated based on the defined Custom Entity *Properties* in Vault. Therefore all properties are directly available on this object.

The properties are named the same as in Vault, including whitespaces. If you want to access such a property you have to enclose it in single quotes.

Syntax

```
1 $customObject.Name
```

Localization

\$customObject is supporting this *feature*:

- all active properties can be accessed via the displayName
- all active properties can be accessed via _SystemName

except: the ones with guids (=user defined)

Example:

\$customObject.'Category Name' can be used when working against english vault \$customObject.Kategorienname can be used when working against german vault \$customObject._CategoryName can be used on all vault language environments

Examples

CustomObject-object on an english environment:

```
1 Created By           : Administrator
2 _CreateUserName      : Administrator
3 Name                 : Apple
4 _Name                : Apple
5 Property Compliance  : Compliant
6 _Compliance          : Compliant
7 Category Name        : Base
8 _CategoryName        : Base
9 Category Glyph       : Autodesk.DataManagement.Client.Framework.Vault.Currency.
10 ↪Properties.ImageInfo
11 _CategoryGlyph       : Autodesk.DataManagement.Client.Framework.Vault.Currency.
12 ↪Properties.ImageInfo
13 Lifecycle Definition : Simple Release Process
14 _LifecycleDefinition : Simple Release Process
15 State                : Work in Progress
16 _State               : Work in Progress
17 State Glyph          : Autodesk.DataManagement.Client.Framework.Vault.Currency.
18 ↪Properties.ImageInfo
```

(continues on next page)

(continued from previous page)

```

16 _StateGlyph           : Autodesk.DataManagement.Client.Framework.Vault.Currency.
   ↳ Properties.ImageInfo
17 Custom Object Name    : Fruit
18 _CustomEntityName     : Fruit
19 Custom Object System Name : e50f0cd1-dfdf-46d0-a5ea-fedb0378d459
20 _CustomEntitySystemName : e50f0cd1-dfdf-46d0-a5ea-fedb0378d459
21 Create Date           : 27.07.2015 15:57:07
22 _CreateDate            : 27.07.2015 15:57:07
23 Obsolete               : False
24 _Obsolete              : False
25 Create Date (Date Only) : 27.07.2015 00:00:00
26 Create Date (Time Only) : 01.01.0001 15:57:00
27 Entity Type            : Fruit
28 _EntityType            : Fruit
29 Entity Type ID          : e50f0cd1-dfdf-46d0-a5ea-fedb0378d459
30 _EntityTypeID           : e50f0cd1-dfdf-46d0-a5ea-fedb0378d459
31 Entity Icon            : Autodesk.DataManagement.Client.Framework.Vault.Currency.
   ↳ Properties.ImageInfo
32 _EntityIcon            : Autodesk.DataManagement.Client.Framework.Vault.Currency.
   ↳ Properties.ImageInfo
33 Path                   :
34 _EntityPath            :
35 Full Path              :
36 _FullPath              :
37 Link Target Path       :
38 _LinkTargetPath        :
39 Type Description        : Fruit
40 _EntityDescription      : Fruit
41 Id                     : 4986
42 MasterId               : 4986
43 PersistentId           :
44 PersistentMasterId      :

```

4.2.4 Email

An Email object is of type *PsObject* and represents the Vault *Change Order* Email Notification.

Syntax

```
1 $email.Addresses
```

The following properties are available :

Type	Name	Description
string	Body	The body of the email.
string	Subject	The subject of the email.
string[]	Addresses	A list of people to send the email to.

Example

```

1 Addresses : {info@colorange.com}
2 Subject   : Autodesk Data Management - ECO-0000022 - Create - New Comment
3 Body      : The following Autodesk Data Management events have occurred:
4             New Comment: ECO-0000022 - Create
5             http://localhost/AutodeskDM/Services/EntityDataCommandRequest.aspx?Vault=Vault&
6             ↪ObjectID=ECO-0000022&ObjectType=ECO&Command=Select
7
8             Vault: Vault
9             User: Administrator
             Time: 4/20/2021 9:03:00 AM

```

4.2.5 File

A File object is of type *PsObject* and represents a file in Vault.

The \$file object is dynamically generated based on the defined File *Properties* in Vault. Therefore all file properties are directly available on this object.

The properties are named the same as in Vault, including whitespaces. If you want to access such a property you have to enclose it in single quotes.

Syntax

```

1 $file.'Full Path'

```

Remarks

Depending on the configured Property types in Vault the properties are converted to a corresponding PowerShell data type.

Vault	PowerShell
Number	Long
Text	String
Boolean	Boolean
Thumbnail	ThumbnailType

Localization

\$file is supporting this *feature*:

- all active properties can be accessed via the displayName
- all active properties can be accessed via _SystemName

except: the ones with guids (=user defined)

except: Internal ones ending with !dateonly or !timeonly (Date Version Created (Time only) and Date Only -> can be accessed via DateVersionCreated.Date or DateVersionCreated.Time)

internal properties starting with Entity!... or File!... can be accessed directly without Entity! (e.g. File!VaultStatus, Entity!HasAttachments)

Examples

Accessing Title for different Vault languages

```

1 $file.Title #can be used with English Vault
2 $file.Titel #can be used with German Vault
3 $file._Title #can be used with all Vault language environments

```

File-object on an english environment:

```

1 Classification : None
2 _Classification : None
3 Version : 13
4 _VersionNumber : 13
5 Comment : Property Edit
6 _Comment : Property Edit
7 Number of Attachments : Autodesk.DataManagement.Client.Framework.Vault.
8   ↳ Currency.Properties.ImageInfo
9   ↳ NumManualAttachments : Autodesk.DataManagement.Client.Framework.Vault.
10   ↳ Currency.Properties.ImageInfo
11 Date Version Created : 19.04.2018 17:35:26
12 _DateVersionCreated : 19.04.2018 17:35:26
13 Created By : coolOrange
14 _CreateUserName : coolOrange
15 Checked In : 19.04.2018 17:35:26
16 _CheckInDate : 19.04.2018 17:35:26
17 File Name : Pad Lock.iam
18 _ClientFileName : Pad Lock.iam
19 File Name (Historical) : Pad Lock.iam
20 _ClientFileName(Ver) : Pad Lock.iam
21 Date Modified : 19.04.2018 17:35:20
22 _ModDate : 19.04.2018 17:35:20
23 File Size : 1064448
24 _FileSize : 1064448
25 Linked to Item : True
26 _ItemLinked : True
27 Checked Out Local Spec :
28 _CheckoutLocalSpec :
29 Checked Out Machine :
30 _CheckoutMachine :
31 Checked Out : 19.04.2018 17:35:24
32 _CheckoutDate : 19.04.2018 17:35:24
33 Checked Out By :
34 _CheckoutUserName :
35 Hidden : False
36 _Hidden : False
37 Latest Version : True
38 _LatestVersion : True
39 Controlled By Change Order : False
40 _ControlledByChangeOrder : False
41 Change Order State :
42 _ChangeOrderState :
43 Visualization Attachment : None
44 _VisualizationAttachment : None

```

(continues on next page)

(continued from previous page)

```

43 Originator : coolOrange
44 _Originator : coolOrange
45 Original Create Date : 09.09.2015 16:00:20
46 _OrigCreateDate : 09.09.2015 16:00:20
47 Thumbnail : Autodesk.DataManagement.Client.Framework.Vault.
   ↳ Currency.Properties.ThumbnailInfo
48 _Thumbnail : Autodesk.DataManagement.Client.Framework.Vault.
   ↳ Currency.Properties.ThumbnailInfo
49 Provider : Inventor
50 _Provider : Inventor
51 iLogicRuleStatus :
52 _iLogicRuleStatus :
53 Folder Path : $/Designs/Padlock/Assemblies
54 _FolderPath : $/Designs/Padlock/Assemblies
55 Name : Pad Lock.iam
56 _Name : Pad Lock.iam
57 File Extension : iam
58 _Extension : iam
59 Property Compliance : Compliant
60 _Compliance : Compliant
61 Property Compliance (Historical) : Noncompliant equivalence
62 _Compliance(Ver) : Noncompliant equivalence
63 Latest Released Revision : False
64 _LatestReleasedRevision : False
65 Released Revision : False
66 _ReleasedRevision : False
67 Initial Release Date :
68 _InitReleaseDate :
69 Initial Approver :
70 _InitApprover :
71 Category Name : Base
72 _CategoryName : Base
73 Category Name (Historical) : Base
74 _CategoryName(Ver) : Base
75 Category Glyph : Autodesk.DataManagement.Client.Framework.Vault.
   ↳ Currency.Properties.ImageInfo
76 _CategoryGlyph : Autodesk.DataManagement.Client.Framework.Vault.
   ↳ Currency.Properties.ImageInfo
77 Category Glyph (Historical) : Autodesk.DataManagement.Client.Framework.Vault.
   ↳ Currency.Properties.ImageInfo
78 _CategoryGlyph(Ver) : Autodesk.DataManagement.Client.Framework.Vault.
   ↳ Currency.Properties.ImageInfo
79 Lifecycle Definition : Flexible Release Process
80 _LifecycleDefinition : Flexible Release Process
81 Lifecycle Definition (Historical) : Flexible Release Process
82 _LifecycleDefinition(Ver) : Flexible Release Process
83 State : Work in Progress
84 _State : Work in Progress
85 State (Historical) : Work in Progress
86 _State(Ver) : Work in Progress
87 State Glyph : Autodesk.DataManagement.Client.Framework.Vault.
   ↳ Currency.Properties.ImageInfo

```

(continues on next page)

(continued from previous page)

```

88  _StateGlyph                : Autodesk.DataManagement.Client.Framework.Vault.
    ↳ Currency.Properties.ImageInfo
89  State Glyph (Historical)   : Autodesk.DataManagement.Client.Framework.Vault.
    ↳ Currency.Properties.ImageInfo
90  _StateGlyph(Ver)          : Autodesk.DataManagement.Client.Framework.Vault.
    ↳ Currency.Properties.ImageInfo
91  Has Markup                 : False
92  _HasMarkup                : False
93  Has Markup (Historical)   : False
94  _HasMarkup(Ver)           : False
95  Revision Scheme           : Standard Alphabetic Format
96  _RevisionDefinition       : Standard Alphabetic Format
97  Revision Scheme (Historical) : Standard Alphabetic Format
98  _RevisionDefinition(Ver)  : Standard Alphabetic Format
99  Revision                  : A
100 _Revision                 : A
101 File Replicated           : True
102 _FileReplicated           : True
103 Author                    : B. ROEPKE
104 _Author                   : B. ROEPKE
105 Comments                  :
106 _Comments                 :
107 Keywords                  : Vault, Tutorial, Padlock
108 _Keywords                 : Vault, Tutorial, Padlock
109 Rev Number                :
110 _RevNumber                :
111 Subject                   : test
112 _Subject                  : test
113 Title                     : Pad Lock
114 _Title                    : Pad Lock
115 Company                   : Autodesk, Inc.
116 _Company                  : Autodesk, Inc.
117 DWG Creator Name          :
118 _DWGCreatorName           :
119 DWG Creator Version        :
120 _DWGCreatorVersion        :
121 Checked By                : B. ROEPKE
122 _CheckedBy                : B. ROEPKE
123 Cost                      : 17
124 _Cost                     : 17
125 Cost Center               :
126 _CostCenter               :
127 Date File Created         :
128 _CreationDate             :
129 Description                : PAD LOCK ASSEMBLY
130 _Description              : PAD LOCK ASSEMBLY
131 Material                  :
132 _Material                 :
133 Part Number               : ERP-41881007
134 _PartNumber               : ERP-41881007
135 Stock Number              :
136 _StockNumber              :

```

(continues on next page)

(continued from previous page)

```

137 User Status :
138 _UserStatus :
139 GeoRss :
140 _GeoRss :
141 Designer : B. ROEPKE
142 _Designer : B. ROEPKE
143 Engineer : B. ROEPKE
144 _Engineer : B. ROEPKE
145 Engr Approved By : D. BRISSON
146 _EngrApprovedBy : D. BRISSON
147 Manager : klaus
148 _Manager : klaus
149 Mfg Approved By :
150 _MfgApprovedBy :
151 Project : PADLOCK
152 _Project : PADLOCK
153 Type Tag :
154 _TypeTag :
155 Item Assignable : True
156 _ItemAssignable : True
157 Obsolete : False
158 _Obsolete : False
159 Has Drawing : True
160 _HasDrawing : True
161 Has Parent Relationship : True
162 _HasParentRelationship : True
163 Has Model State :
164 _HasModelState :
165 Is Table Driven :
166 _IsTableDriven :
167 Is True Model State :
168 _IsTrueModelState :
169 Latest Released Date :
170 _LatestReleaseDate :
171 Latest Approver :
172 _LatestApprover :
173 Entity Type : File
174 _EntityType : File
175 Entity Type ID : FILE
176 _EntityTypeID : FILE
177 Entity Icon : Autodesk.DataManagement.Client.Framework.Vault.
    ↳ Currency.Properties.ImageInfo
178 _EntityIcon : Autodesk.DataManagement.Client.Framework.Vault.
    ↳ Currency.Properties.ImageInfo
179 Path : $/Designs/Padlock/Assemblies
180 _EntityPath : $/Designs/Padlock/Assemblies
181 Full Path : $/Designs/Padlock/Assemblies/Pad Lock.iam
182 _FullPath : $/Designs/Padlock/Assemblies/Pad Lock.iam
183 Attachments : False
184 _HasAttachments : False
185 Link Target Path :
186 _LinkTargetPath :

```

(continues on next page)

(continued from previous page)

```

187 Date Version Created (Date Only) : 19.04.2018 00:00:00
188 Date Version Created (Time Only) : 01.01.0001 17:35:00
189 Checked In (Date Only) : 19.04.2018 00:00:00
190 Checked In (Time Only) : 01.01.0001 17:35:00
191 Date Modified (Date Only) : 19.04.2018 00:00:00
192 Date Modified (Time Only) : 01.01.0001 17:35:00
193 Checked Out (Date Only) : 19.04.2018 00:00:00
194 Checked Out (Time Only) : 01.01.0001 17:35:00
195 Original Create Date (Date Only) : 09.09.2015 00:00:00
196 Original Create Date (Time Only) : 01.01.0001 16:00:00
197 Initial Release Date (Date Only) :
198 Initial Release Date (Time Only) :
199 Latest Released Date (Date Only) :
200 Latest Released Date (Time Only) :
201 Type Description : Autodesk Inventor Assembly
202 _EntityDescription : Autodesk Inventor Assembly
203 Vault Status : Autodesk.DataManagement.Client.Framework.Vault.
    ↳ Currency.Properties.EntityStatusImageInfo
204 _VaultStatus : Autodesk.DataManagement.Client.Framework.Vault.
    ↳ Currency.Properties.EntityStatusImageInfo
205 Vault Status Modifier : False
206 _VaultStatusModifier : False
207 Id : 138381
208 MasterId : 28663
209 PersistentId :
210 PersistentMasterId :
211 IsCheckedOut : False
212 ThinClientHyperLink : http://localhost/AutodeskTC/Vault/explore/file/28663
213 ThickClientHyperLink : http://localhost/AutodeskDM/Services/
    ↳ EntityDataCommandRequest.aspx?Va
214 ult=Vault&ObjectId=%24%2FDesigns%2FPadlock%2FAssemblies
    ↳ %2FPad+Lock.iam
215 &ObjectType=File&Command=Select
216 LocalPath :
```

4.2.6 FileBomRow

A file BomRow is of type *PsObject* and represents a single row entry in the **CAD BOM** from a file in Vault.

The \$fileBomRow object is dynamically generated based on the *Properties* coming from the CAD BOM and provides additionally all the members from its corresponding *File* (except for *Virtual Components*).

The *BOM specific properties* are named the same as in the CAD BOM, including whitespaces and starting with the 'Bom_' prefix. If you want to access such a property you have to enclose it in single quotes.

Syntax

```
$bomRow.'Bom_Part Number'
```

The following properties are always added in addition to the *File* members:

Type	Name	Description
string	Bom_Num	The Part Number of the component. When the component has no Part Number, it is the component name.
int	Bom_Row	The Order of the component in the BOM. It doesn't really affect the BOM structure, but it helps with sorting rows. 0 for components for which no information is available in the Structured BOM View (e.g. Structured View disabled)
string	Bom_Positi	The Position Number of the component in the BOM. Empty for components for which no information is available in the Structured BOM View (e.g. Structured View disabled)
string	Bom_Unit	The Unit of Measure of the component (e.g. Each, inch, liter, kg,...).
string	Bom_XRe	The XRefTyp which specifies whether the component is internal or external in relation to the design file.
string	Bom_Struc	The BomStructure of the component can be Normal, Phantom, Purchased, Reference, DynamicPhantom or Inseperable.
double	Bom_Quan	The Quantity (QTY) is the total quantity of the occurrence calculated based on the components Item Quantity, its Unit Quantity and overridden component Quantities.
int	Bom_Item	The Item Quantity (Item QTY) is the number of instances of a component in the BOM.
double	Bom_Unit	The Unit Quantity (Unit QTY) is the amount which each discrete instance of a component adds to the total quantity.
string	Bom_Mod	The model state name of the used component in the BOM. "Master" specifies the primary model state and is also returned for files without associated model states.

Localization

\$fileBomRow is supporting this *feature*

- all active properties can be accessed via the displayName
- all active properties can be accessed via _SystemName

except: the ones with guids (=user defined)

Examples

Example of a bomRow (on an english environment):

```
Bom_RowOrder           : 3
Bom_PositionNumber     : 7
Bom_Number             : ERP-41880947
Bom_Unit               : Each
Bom_ItemQuantity       : 2
Bom_UnitQuantity       : 1.2
Bom_Quantity           : 2.4
Bom_XRefTyp            : External
Bom_Structure          : Normal
Bom_ModelState         : Master
Bom_Author             : B. ROEPKE
```

(continues on next page)

(continued from previous page)

```

Bom_KeyWords          : Vault, Tutorial, Padlock
Bom_Revision          : A
Bom_Category          : Vault Sample Models
Bom_Company           : Autodesk, Inc.
Bom_Cost Center       : PRODUCT DESIGN
Bom_Creation Time     : 11.04.1976 22:30:33
Bom_Description       : 164987124
Bom_Designer          : B. ROEPKE
Bom_Engineer          : B. ROEPKE
Bom_Cost              : 17
Bom_Part Number       : ERP-41880947
Bom_Project           : PADLOCK
Bom_Vendor            : Autodesk, Inc.
Bom_Checked By        : B. ROEPKE
Bom_Date Checked      : 19.12.2002 08:09:56
Bom_Design Status     : 3
Bom_Engr Approved By : D. BRISSON
Bom_Engr Date Approved : 24.12.2002 08:09:56
Bom_Mfg Date Approved : 01.01.1601 00:00:00
Bom_User Status       : RELEASED
Bom_Catalog Web Link  : http://www.autodesk.com/inventor/
Bom_Document SubType : {E60F81E1-49B3-11D0-93C3-7E0706000000}
Bom_Document SubType Name : Assembly
Bom_EquivalenceValue  : 100002
Bom_Subject           : Bach
Bom_Manager           : Georg
Bom_Title             : 2077753685
Classification        : None
_Classification       : None
Version              : 3
_VersionNumber        : 3
Comment              : Property Edit
_Comment             : Property Edit
Number of Attachments : Autodesk.DataManagement.Client.Framework.Vault.
↪Currency.Properties.ImageInfo
_NumManualAttachments : Autodesk.DataManagement.Client.Framework.Vault.
↪Currency.Properties.ImageInfo
Date Version Created  : 19.04.2018 17:35:27
_DateVersionCreated   : 19.04.2018 17:35:27
Created By           : coolOrange
_CreateUserName       : coolOrange
Checked In           : 19.04.2018 17:35:27
_CheckInDate         : 19.04.2018 17:35:27
File Name            : Combo Assembly.iam
_ClientFileName       : Combo Assembly.iam
IsCheckedOut         : False
...
Id                   : 138383
MasterId             : 28630
PersistentId         :
PersistentMasterId   :
ThinClientHyperLink  : http://localhost/AutodeskTC/Vault/explore/file/28630

```

(continues on next page)

(continued from previous page)

```
ThickClientHyperLink      : http://localhost/AutodeskDM/Services/
↳ EntityDataCommandRequest.aspx?Vault=Vault&ObjectI
                           d=%24%2FDesigns%2FCombo+Assembly.iam&ObjectType=File&
↳ Command=Select
```

Example of a bomRow for a Virtual component (provides only CAD BOM properties):

```
Bom_RowOrder      : 8
Bom_PositionNumber : 8
Bom_Number        : SomeVirtualComponent
Bom_Unit          : Each
Bom_ItemQuantity  : 1
Bom_UnitQuantity  : 1
Bom_Quantity      : 100
Bom_XRefTyp       : Internal
Bom_Structure     : Purchased
Bom_ModelState    :
Bom_Creation Time : 12/31/1600 23:00:00
Bom_Description   : This is a virtual component
Bom_Cost          : 0
Bom_Part Number   : SomeVirtualComponent
Bom_Date Checked  : 12/31/1600 23:00:00
Bom_Design Status : 0
Bom_Engr Date Approved : 12/31/1600 23:00:00
Bom_Mfg Date Approved : 12/31/1600 23:00:00
Bom_Material      : Generic
Bom_Stock Number  : SOME_VIRTUAL
Bom_EquivalenceValue : SomeVirtualComponent
Bom_Content Center File : False
```

Example of corrupt BOM where file of according row is not available any more in Vault (got removed or purged):

FileBomRows that are not resolvable to any existing Vault file can only provide part of the Bill of Materials data, such as the *Bom_RowOrder* and the *Bom_PositionNumber*.

Accessing other properties on those rows can throw a *MissingCadBomException* or a *CorruptCadBomException*, even if those are suppressed from PowerShell by default.

```
# foreach($prop in $bomRow.psobject.properties) {
#   try {
#     # this forces the exception to be thrown
#     $value = $prop.get_Value()
#   } catch [Exception] {
#     $bomRow | Add-Member -MemberType NoteProperty -Name $prop.Name -Value $prop.Value-
↳ Force
#   }
# }
Bom_RowOrder      : 1
Bom_PositionNumber : 1
Bom_ItemQuantity  : 2
Bom_XRefTyp       : External
Bom_Number        : # throws: coolOrange.VaultServices.Vault.FileBom.
↳ CorruptCadBomException: Please checkout and re-checkin the file 'CorruptFileBom.iam'
↳ (Id: 138296) ...
```

(continues on next page)

(continued from previous page)

```

Bom_Unit          : # throws: coolOrange.VaultServices.Vault.FileBom.
↳ CorruptCadBomException: Please checkout and re-checkin the file 'CorruptFileBom.iam'
↳ (Id: 138296) ...
Bom_UnitQuantity  : # throws: coolOrange.VaultServices.Vault.FileBom.
↳ CorruptCadBomException: Please checkout and re-checkin the file 'CorruptFileBom.iam'
↳ (Id: 138296) ...
Bom_Quantity      : # throws: coolOrange.VaultServices.Vault.FileBom.
↳ CorruptCadBomException: Please checkout and re-checkin the file 'CorruptFileBom.iam'
↳ (Id: 138296) ...
Bom_Structure     : # throws: coolOrange.VaultServices.Vault.FileBom.
↳ CorruptCadBomException: Please checkout and re-checkin the file 'CorruptFileBom.iam'
↳ (Id: 138296) ...
Bom_ModelState    : # throws: coolOrange.VaultServices.Vault.FileBom.
↳ CorruptCadBomException: Please checkout and re-checkin the file 'CorruptFileBom.iam'
↳ (Id: 138296) ...

```

4.2.7 Folder

A Folder object is of type *PsObject* and represents a folder in Vault.

The \$folder object is dynamically generated based on the defined Folder *Properties* in Vault.

The properties are named the same as in Vault, including whitespaces. If you want to access such a property you have to enclose it in single quotes.

Syntax

```
$folder.'Folder Path'
```

Localization

\$folder is supporting this *feature*:

- all active properties can be accessed via the displayName
- all active properties can be accessed via _SystemName

except: the ones with guids (=user defined)

Example:

\$folder.'Folder Path' can be used when working against english vault \$folder.Ordnerpfad can be used when working against german vault \$folder._FolderPath can be used on all vault language environments

Examples

Folder-object on an english environment:

```

1 Created By           : Administrator
2 _CreateUserName      : Administrator
3 Folder Path          : $/Designs/Vault
4 _FolderPath          : $/Designs/Vault
5 Name                 : Vault
6 _Name                : Vault
7 Property Compliance  : Compliant
8 _Compliance          : Compliant
9 Category Name        : Folder
10 _CategoryName        : Folder
11 Category Glyph       : Autodesk.DataManagement.Client.Framework.Vault.Currency.
12 ↪Properties.ImageInfo
13 _CategoryGlyph       : Autodesk.DataManagement.Client.Framework.Vault.Currency.
14 ↪Properties.ImageInfo
15 Lifecycle Definition : Basic Release Process
16 _LifeCycleDefinition : Basic Release Process
17 State                : Work in Progress
18 _State               : Work in Progress
19 State Glyph          : Autodesk.DataManagement.Client.Framework.Vault.Currency.
20 ↪Properties.ImageInfo
21 _StateGlyph          : Autodesk.DataManagement.Client.Framework.Vault.Currency.
22 ↪Properties.ImageInfo
23 Create Date          : 11.04.2016 15:53:47
24 _CreateDate          : 11.04.2016 15:53:47
25 Type Tag            :
26 _TypeTag             :
27 Share Path           :
28 _SharePath           :
29 Obsolete             : False
30 _Obsolete            : False
31 Entity Type         : Folder
32 _EntityType          : Folder
33 Entity Type ID      : FLDR
34 _EntityTypeID        : FLDR
35 Entity Icon          : Autodesk.DataManagement.Client.Framework.Vault.Currency.
36 ↪Properties.ImageInfo
37 _EntityIcon          : Autodesk.DataManagement.Client.Framework.Vault.Currency.
38 ↪Properties.ImageInfo
39 Path                 : $/Designs
40 _EntityPath          : $/Designs
41 Full Path            : $/Designs/Vault
42 _FullPath            : $/Designs/Vault
43 Link Target Path     :
44 _LinkTargetPath      :
45 Create Date (Date Only) : 11.04.2016 00:00:00
46 Create Date (Time Only) : 01.01.0001 15:53:00
47 Type Description    : Folder
48 _EntityDescription    : Folder
49 Folder Create Date    : 11.04.2016 15:53:47

```

(continues on next page)

(continued from previous page)

```

44 _Folder!CreateDate      : 11.04.2016 15:53:47
45 Id                     : 26766
46 MasterId               : 26766
47 PersistentId           :
48 PersistentMasterId     :
49 IsLibrary               : False
50 ThinClientHyperLink    : http://localhost/AutodeskTC/Vault/explore/folder/26766
51 ThickClientHyperLink   : http://localhost/AutodeskDM/Services/EntityDataCommandRequest.
    ↪ aspx?Vault=Vault&ObjectId=%24%2FDesigns%2FVault&ObjectType=Folder&
52    ↪ Command=Select

```

4.2.8 Item

An Item object is of type *PsObject* and represents an Item located in Vault.

The \$item object is dynamically generated based on the defined Item *Properties* in Vault. Therefore all item properties are directly available on this object.

The properties are named the same as in Vault, including whitespaces. If you want to access such a property you have to enclose it in single quotes.

Syntax

```
1 $item.Number
```

Remarks

Item objects returned by the *Get-VaultChangeOrderAssociations* cmdlet also include *User Defined Link properties* if they are available in Vault.

These Linked properties names are prefixed with 'Record_' on the Item object.

Localization

\$item is supporting this *feature*:

- all active properties can be accessed via the displayName
- all active properties can be accessed via _SystemName

except: the ones with guids (=user defined)

Examples

Accessing Number for different Vault languages:

```
1 $item.Number #can be used with English Vault
2 $item.Nummer #can be used with German Vault
3 $item._Number #can be used with all Vault language environments
```

Accessing a Link property named VersionToChange returned by *Get-VaultChangeOrderAssociations*:

```
1 $item.Record_VersionToChange
```

Item-object on an english environment:

```
1 Version : 5509
2 _VersionNumber : 5509
3 Comment :
4 _Comment :
5 Number of Attachments : Autodesk.DataManagement.Client.Framework.Vault.
6 ↳Currency.Properties.ImageInfo
7 _NumManualAttachments : Autodesk.DataManagement.Client.Framework.Vault.
8 ↳Currency.Properties.ImageInfo
9 Date Modified : 13.12.2017 19:19:56
10 _ModDate : 13.12.2017 19:19:56
11 Latest Version : True
12 _LatestVersion : True
13 Controlled By Change Order : False
14 _ControlledByChangeOrder : False
15 Change Order State :
16 _ChangeOrderState :
17 Originator : coolOrange
18 _Originator : coolOrange
19 Original Create Date : 09.07.2015 10:18:00
20 _OrigCreateDate : 09.07.2015 10:18:00
21 Thumbnail : Autodesk.DataManagement.Client.Framework.Vault.
22 ↳Currency.Properties.ThumbnailInfo
23 _Thumbnail : Autodesk.DataManagement.Client.Framework.Vault.
24 ↳Currency.Properties.ThumbnailInfo
25 Provider : Inventor
26 _Provider : Inventor
27 Name : 100001
28 _Name : 100001
29 Property Compliance : Compliant
30 _Compliance : Compliant
31 Property Compliance (Historical) : Compliant
32 _Compliance(Ver) : Compliant
33 Latest Released Revision : False
34 _LatestReleasedRevision : False
35 Released Revision : False
36 _ReleasedRevision : False
37 Initial Release Date :
38 _InitReleaseDate :
39 Initial Approver :
40 _InitApprover :
```

(continues on next page)

(continued from previous page)

```

37 Category Name                : Document
38 _CategoryName                : Document
39 Category Name (Historical)    : Document
40 _CategoryName(Ver)            : Document
41 Category Glyph                : Autodesk.DataManagement.Client.Framework.Vault.
    ↳ Currency.Properties.ImageInfo
42 _CategoryGlyph                : Autodesk.DataManagement.Client.Framework.Vault.
    ↳ Currency.Properties.ImageInfo
43 Category Glyph (Historical)    : Autodesk.DataManagement.Client.Framework.Vault.
    ↳ Currency.Properties.ImageInfo
44 _CategoryGlyph(Ver)           : Autodesk.DataManagement.Client.Framework.Vault.
    ↳ Currency.Properties.ImageInfo
45 Lifecycle Definition          : Item Release Process
46 _LifecycleDefinition          : Item Release Process
47 Lifecycle Definition (Historical) : Item Release Process
48 _LifecycleDefinition(Ver)      : Item Release Process
49 State                        : Work in Progress
50 _State                       : Work in Progress
51 State (Historical)           : Work in Progress
52 _State(Ver)                  : Work in Progress
53 State Glyph                  : Autodesk.DataManagement.Client.Framework.Vault.
    ↳ Currency.Properties.ImageInfo
54 _StateGlyph                  : Autodesk.DataManagement.Client.Framework.Vault.
    ↳ Currency.Properties.ImageInfo
55 State Glyph (Historical)      : Autodesk.DataManagement.Client.Framework.Vault.
    ↳ Currency.Properties.ImageInfo
56 _StateGlyph(Ver)             : Autodesk.DataManagement.Client.Framework.Vault.
    ↳ Currency.Properties.ImageInfo
57 Revision Scheme              : Standard Alphabetic Format
58 _RevisionDefinition          : Standard Alphabetic Format
59 Revision Scheme (Historical)  : Standard Alphabetic Format
60 _RevisionDefinition(Ver)      : Standard Alphabetic Format
61 Revision                     : A
62 _Revision                    : A
63 Last Updated By              : coolOrange
64 _LastModifiedUserName         : coolOrange
65 Number                       : 100001
66 _Number                      : 100001
67 Title (Item,CO)              : Front Hub Carrier R.ipt
68 _Title(Item,CO)              : Front Hub Carrier R.ipt
69 Units                        : Each
70 _Units                       : Each
71 Description (Item,CO)         : PAD LOCK ASSEMBLY
72 _Description(Item,CO)         : PAD LOCK ASSEMBLY
73 Type                        :
74 _ItemClass                   :
75 Equivalence Value            : 100001
76 _EquivalenceValue            : 100001
77 Effectivity                  :
78 _ItemEffectivity             :
79 Eff. Start                   :
80 _ItemEffectivityStart        :

```

(continues on next page)

(continued from previous page)

```

81 Eff. End :
82 _ItemEffectivityEnd :
83 Obsolete : False
84 _Obsolete : False
85 File Link State : Current
86 _FileLinkState : Current
87 Latest Released Date :
88 _LatestReleaseDate :
89 Latest Approver :
90 _LatestApprover :
91 Entity Type : Item
92 _EntityType : Item
93 Entity Type ID : ITEM
94 _EntityTypeID : ITEM
95 Entity Icon : Autodesk.DataManagement.Client.Framework.Vault.
    ↳ Currency.Properties.ImageInfo
96 _EntityIcon : Autodesk.DataManagement.Client.Framework.Vault.
    ↳ Currency.Properties.ImageInfo
97 Path :
98 _EntityPath :
99 Full Path :
100 _FullPath :
101 Attachments : True
102 _HasAttachments : True
103 Link Target Path :
104 _LinkTargetPath :
105 Date Modified (Date Only) : 13.12.2017 00:00:00
106 Date Modified (Time Only) : 01.01.0001 19:19:00
107 Original Create Date (Date Only) : 09.07.2015 00:00:00
108 Original Create Date (Time Only) : 01.01.0001 10:18:00
109 Initial Release Date (Date Only) :
110 Initial Release Date (Time Only) :
111 Latest Released Date (Date Only) :
112 Latest Released Date (Time Only) :
113 Type Description : Item
114 _EntityDescription : Item
115 Vault Status :
116 _VaultStatus :
117 Id : 138286
118 MasterId : 16437
119 PersistentId :
120 PersistentMasterId :
121 ThinClientHyperLink : http://localhost/AutodeskTC/Vault/items/item/16437
122 ThickClientHyperLink : http://localhost/AutodeskDM/Services/
    ↳ EntityDataCommandRequest.
123 aspx?Vault=Vault&ObjectId=100001&
    ↳ ObjectType=ItemRevision&Comma
124 nd=Select

```

4.2.9 ItemBomRow

An ItemBomRow is of type *Item* and represents a single row entry in the BOM of an Item in Vault.

The \$ItemBomRow object is dynamically generated with additional *BOM specific properties*.

Syntax

```
$ItemBomRow.Bom_Number
```

Following properties are always available in addition to the *Item* members:

Type	Name	Description
string	Bom_Number	The Number of the item row. For rows assigned to an item this is the item Number.
int	Bom_RowOrder	The Order of the item in the BOM.
string	Bom_PositionNumber	The Position Number of the item row.
double	Bom_Quantity	The Quantity of the item row.
string	Bom_Unit	The Unit of Measure of the item row (e.g. Each, inch, liter, kg,...).
bool	Bom_IsCad	Indicates if the item row is created by CAD or manually.

Remarks

ItemBomRows that are not linked to any existing Vault item only provide the BOM specific properties. *No Item properties* are available for unassigned rows.

Localization

\$ItemBomRow is supporting this *feature*:

- all active properties can be accessed via the displayName
- all active properties can be accessed via _SystemName

except: the ones with guids (=user defined)

Examples

ItemBomRow-object on an english environment:

```

1 Bom_Number           : NU-41880947
2 Bom_Unit             : Each
3 Bom_Quantity         : 1
4 Bom_RowOrder         : 1
5 Bom_PositionNumber   : 1
6 Bom_IsCad            : True
7 Version              : 29
8 _VersionNumber       : 29
9 Comment              :
10 _Comment            :
11 Number of Attachments : Autodesk.DataManagement.Client.Framework.Vault.
```

(continues on next page)

(continued from previous page)

```

12  ↪Currency.Properties.ImageInfo
    _NumManualAttachments      : Autodesk.DataManagement.Client.Framework.Vault.
13  ↪Currency.Properties.ImageInfo
14  Date Modified              : 18.06.2019 15:08:46
15  _ModDate                   : 18.06.2019 15:08:46
16  Equivalence Value          : ERP-16209830
17  _EquivalenceValue          : ERP-16209830
18  ...
19  Id                          : 138730
20  MasterId                   : 569
21  PersistentId                :
22  PersistentMasterId          :
23  ThinClientHyperLink         : http://localhost/AutodeskTC/Vault/items/item/569
24  ThickClientHyperLink        : http://localhost/AutodeskDM/Services/
    ↪EntityDataCommandRequest.
    aspx?Vault=Vault&ObjectId=NU-41880947&
    ↪ObjectType=ItemRevision&Comma
    nd=Select
25

```

4.2.10 Job

A Job object is of type *PsObject* and represents a queued VaultJob.

The \$job object is dynamically generated based on the job *Parameters*. Therefore all job parameters are directly available on this object.

When accessing parameters containing whitespaces, you have to enclose such a property in single quotes.

Syntax

```
1 $job.Name
```

The following properties are always added :

Type	Name	Description
long	Id	A unique identifier for the job.
string	Name	The name of the job.
string	Description	A description of the job.
int	Priority	The priority of the job.
string	CreateUserName	The name of the user who created the job.
DateTime	CreationDate	The date the job was created.
bool	Reserved	Indicates whether the job was reserved or not.
string	ReserveDate	The date that the job was reserved.
DateTime	ReservedMachine	The computer name of the client that the job is reserved to.
string	Status	The status of the job. Possible values: Pending,Running,Success,Failed
string	ErrorMessage	The error message for failed jobs.

Remarks

The data types of the *parameters* on the object are all of type *string*.
When parameters with one of the following property names are set, they do not get overwritten.

Example

```
1 Id      : 261026
2 Name    : Sample.CreatePDF
3 Description : Job for creating PDF
4 Priority : 10
5 CreateUserName : Administrator
6 CreationDate : 29.05.2019 13:08:29
7 Reserved  : False
8 ReserveDate : 01.01.0001 00:00:00
9 ReservedMachine :
10 Status    : Pending
11 ErrorMessage :
12 EntityId   : 88
13 EntityClassId : FILE
```

4.2.11 User

The User object is of type *PsObject* and represents a Vault Security User or *Change Order* Routing Members.

Syntax

```
1 $user.Name
```

The following properties are always added :

Type	Name	Description
long	Id	The Id of the user.
string	Name	The Name of the User.
string[]	Roles	The associated Vault roles for the user.

Example

```
1 Id      : 1
2 Name    : Administrator
3 Roles   : {Change Requestor, Change Administrator, Reviewer, Approver...}
```

4.2.12 Vault

The Vault object is of type *Autodesk.Connectivity.WebServicesTools.WebServiceManager* and grants direct access to the Vault APIs.

Syntax

```
$vault.DocumentService
```

Following properties are available:

Type	Name	Description	Access type
	AdminService	Contains methods for manipulating users and groups.	read-only
	AnalyticsService	Contains methods for Analytics within a Vault.	read-only
	AuthService	Contains methods for authenticating to the Vault server.	read-only
	AutodeskAuthService	Contains methods for authenticating to Vault using Autodesk ID credentials.	read-only
	BehaviorService	Contains methods for manipulating behaviors.	read-only
	CategoryService	Contains methods for manipulating categories.	read-only
	ChangeOrderService	Contains methods for creating and manipulating change orders.	read-only
	CustomEntityService	A collection of methods related to the Custom Entity entity type.	read-only
	DocumentService	Contains methods for manipulating files and folders within a vault.	read-only
	DocumentService-Extensions	Contains more methods for manipulating files and folders within a vault.	read-only
	FilestoreService	Contains methods for uploading and downloading binary file data.	read-only
	FilestoreVaultService	Contains methods to determine information about the Knowledge Vaults.	read-only
	ForumService	Contains methods for posting messages.	read-only
	InformationService	Contains methods to determine information about the server, such as the version and product level.	read-only
	ItemService	Contains methods for manipulating items.	read-only
	JobService	Contains methods for manipulating the job queue.	read-only
	KnowledgeVaultService	Contains methods for getting information about the vaults on the server.	read-only
	LifeCycleService	Contains methods related to the lifecycle behavior.	read-only
	NumberingService	Contains methods for managing numbering schemes for Files, Items, Change Orders and Items.	read-only
	PackageService	Contains methods for importing and exporting item data.	read-only
	PropertyService	Contains methods for manipulating properties on Entities.	read-only
	ReplicationService	Contains methods for transferring ownership between workgroups.	read-only
bool	ReSignIn	Gets or sets the re-sign in behavior on all the services in the WebServiceManager.	write-only
	RevisionService	Contains methods for manipulating revision values and schemes for Entities.	read-only
	SecurityService	Contains methods for logging into and out of vaults and setting security on specific Entities.	read-only
	WebServiceCredentials		read-write
68	WinAuthService	Contains methods for logging into and out of vaults using Windows credentials.	read-only

Remarks

All the properties of the Vault object are part of the Vault API.
You can look them up in the VaultSDK documentation under *WebServiceManager Class Members*.

4.2.13 VaultConnection

The VaultConnection object is of type *Autodesk.DataManagement.Client.Framework.Vault.Currency.Connections.Connection* and grants direct access to the Vault Development Framework.

Syntax

```
$VaultConnection.FileManager
```

Following properties are available:

Type	Name	Description	Access type
	Category-Manager	Gets an object which encapsulates all access to Vault Categories	read-only
	ChangeOrderManager	Gets an object which encapsulates all access to Vault Change Orders	read-only
	ConfigurationManager	Gets an object which manages configuration data on the vault server.	read-only
	CustomObjectManager	Gets an object which encapsulates all access to Vault Custom Objects	read-only
	EntityOperations	Gets an object which provides a set of workflows that can be applied to any entity object.	read-only
	FileManager	Gets an object which encapsulates all access to Vault Files	read-only
	FolderManager	Gets an object which encapsulates all access to Vault Folders	read-only
	ItemManager	Gets an object which encapsulates all access to Vault Items	read-only
	LinkManager	Gets an object which encapsulates all access to Links	read-only
	PersistableIdManager	Gets an object which encapsulates all interaction with persistable ids	read-only
	PropertyManager	Gets an object which encapsulates all access to vault property definitions and property values.	read-only
	WebServiceManager	Gets the low level Web Service Manager which stores the underlying physical connection to the server. This can be used to make direct calls to the web service layer for any functionality that is not provided by the Framework.	read-only
	WorkingFoldersManager	Gets an object which encapsulates all access to a vaults working folders	read-only
string	IdentityKey	Gets a string which uniquely identifies this connection.	read-only
bool	IsAnonymousConnection	Gets a value which identifies whether or not we have an anonymous connection to the server.	read-only
bool	IsAutodeskAuthenticatedConnection	Gets a value which identifies whether or not the connection to the server used Autodesk ID authentication.	read-only
bool	IsConnected	Gets a values which tells if this connection object has an active connection. If a logout occurs, then the connection will essentially be invalid.	read-only
bool	IsReadOnly	Gets a value which identifies whether or not this is a read only connection to the server	read-only
bool	IsServerOnlyConnection	Gets a value which identifies whether or not we have a connection to a server but not to a specific vault.	read-only
bool	IsWindowsAuthenticatedConnection	Gets a value which identifies whether or not the connection to the server used windows authentication.	read-only
string	Server	Gets the name of the server that we are connected to.	read-only
string	Ticket	Gets an encrypted ticket that represents the unique connection to the server.	read-only
long	UserID	Gets the unique identity of the authenticated user on the vault server.	read-only
70 string	UserName	Gets the name of the authenticated user.	read-only
string	Vault	Gets the name of the vault that we are connected to.	read-only

Remarks

All the members of the VaultConnection object are part of the Vault VDF API.
You can look them up in the VaultSDK documentation under *Connection Class Members*.

4.2.14 VaultExplorerUtil

The VaultExplorerUtil object is of type *Autodesk.Connectivity.Explorer.ExtensibilityTools.IExplorerUtil* and allows working directly with the Vault Explorer.

Syntax

```
$vaultExplorerUtil
```

Following methods are available:

Icons	Type	Name	Description
	void	UpdateFileProperties(File file, System.Collections.Generic.Dictionary<F props)	Updates a set of properties for a file.

Remarks

All the methods of the VaultExplorerUtil object are part of the Vault API.
You can look them up in the VaultSDK documentation under *IExplorerUtil Interface Members*.

LOGGING

powerVault uses [Apache log4net](#) as core logging library, and additionally [PostSharp Diagnostics](#) for extended Debug logging.

By default, all the logs are stored in a logfile located in 'C:\Users\{USER}\AppData\Local\coolOrange\powerVault\Logs\powerVault.log' and it contains only Infos, Warnings and Errors.

The log4net settings file is located in C:\Program Files\coolOrange\Modules\powerVault\powerVault.log4net. Further information about log4Net Configurations can be found [here](#).

5.1 When to change the logging behavior?

When you have issues or when you want to get a more detailed knowledge about what went wrong, you can increase the [loglevel](#).

Note: When changing the loglevel to *DEBUG* [PostSharp Diagnostics](#) will be enabled and will log all the function calls into the log files.

This could cause performance issues

5.2 LogFile

You can see, that there are multiple logging-Appenders used.

If you want to change the [Logging Levels](#) for the logfile, please visit following appender:

```
1 <appender name="FileAppender" type="log4net.Appender.RollingFileAppender">
```

In the lines

```
1 <root>
2   <level value="INFO" />
3   <appender-ref ref="FileAppender"/>
4   <appender-ref ref="ColoredConsoleAppender" />
5 </root>
```

you can configure the logginglevel for all appenders.

You could set the level to "DEBUG", than all the appenders will log in debug.

In the line

```
1 <param name="File" value="${LOCALAPPDATA}\\coolOrange\\powerVault\\Logs\\powerVault.log" />
```

you can configure the outputpath and name of the logfile.

5.3 PowerShell IDE

PowerShell IDEs like PowerShell console (and PowerShell ISE) are configured to show the logging levels in a different color.

```
1 <mapping>
2     <level value="DEBUG" />
3     <foreColor value="Black" />
4     <backColor value="White" />
5 </mapping>
6 <mapping>
7     <level value="INFO" />
8     <backColor value="DarkGreen" />
9 </mapping>
10 <mapping>
11     <level value="WARN" />
12     <backColor value="DarkYellow" />
13 </mapping>
14 <mapping>
15     <level value="ERROR" />
16     <backColor value="Red" />
17 </mapping>
18 <mapping>
19     <level value="FATAL" />
20     <backColor value="DarkRed" />
21 </mapping>
```

These and many other options can be configured in the appender named **ColoredConsoleAppender**.

CHANGE LOGS

6.1 powerVault v24

6.1.1 v24.0.9

05-03-2024

General

- The type names (`TypeNameOfValue`) for properties of the following objects are now returned correctly:
File, Item, Folder, Change Order, FileBomRow, ItemBomRow and *Custom Object*.

6.1.2 v24.0.8

23-02-2024

General

- Updated internally used coolOrange.VaultServices library to: 24.0.2

6.1.3 v24.0.7

13-10-2023

Fixed

- Deadlock issue in *Show-Inspector* where no Inspector window opened within required BOM Window functions (using powerGate v24.0.5 and later)

6.1.4 v24.0.5

25-08-2023

Fixed

- Issue with converting *Thin/ThickClientHyperLink* to string which sometimes returned a URL decoded URI instead of an encoded one

6.1.5 v24.0.4

23-08-2023

Features

- New properties for *File*, *Item*, *Folder* and *ChangeOrder* objects:
 - *ThinClientHyperLink* provides a link to open the entity in the [Vault ThinClient](#)
 - *ThickClientHyperLink* provides a link to open the entity with the Vault Client

6.1.6 v24.0.2

30-06-2023

General

- Updated Licensing to version: [18.3.1](#)

6.1.7 v24.0.1

18-04-2023

General

- Added support for Vault 2024
- Updated Licensing to version: [18.2.29](#)
- End User License Agreement (EULA) has changed

6.2 powerVault v23

6.2.1 v23.0.15

14-04-2023

General

- Updated internally used PowerShell library to: [1.0.32](#)

6.2.2 v23.0.14

22-02-2023

Features

- *Show-Inspector* now allows passing the variable name case-insensitive to the *-Highlight* argument

Fixed

- *Show-Inspector*:
 - Issues when displaying WPF list control variables that caused either the host application to crash or the Inspector window to suddenly close when scrolling to such variables

- Deadlock issue where the Inspector window freezes when the cmdlet is executed directly on the UI thread (e.g. inside [powerEvents](#) actions or [Vault DataStandard](#) functions).
- Issue with incorrectly displayed *\$null* values for properties of PowerShell variables that cannot be retrieved within 3 seconds (e.g. for all WPF dependency properties)

6.2.3 v23.0.11

09-02-2023

General

- Internal API changes to search a folder by its Id required for the powerEvents client customization [SubmitJobsOnVaultMenuItemClick](#)

6.2.4 v23.0.10

23-01-2023

Fixed

- *Get-VaultFileBOM*:
 - Issue with *-GetChildrenBy LatestVersion* and *LatestReleasedVersion* where accessing file properties for *FileBomRows* of promoted *Virtual Components* that originated from updated *Phantom* assemblies logged misleading warning messages
 - Issue with *-GetChildrenBy ExactVersion* where the *FileBomRow* returned file properties of the phantom assembly of promoted *Virtual Components* instead of no file properties

6.2.5 v23.0.9

16-01-2023

Fixed

- *Get-VaultFileBOM* with *-GetChildrenBy LatestVersion*: Issue where promoted parts that originated from nested updated *Phantom* assemblies returned **Bom_PositionNumber** 0 and empty **Bom_RowOrder**

6.2.6 v23.0.8

18-10-2022

General

- The setup has been adjusted so that the message for the required but not installed Vault Client version is displayed immediately when starting the setup

6.2.7 v23.0.7

27-09-2022

General

- The setup has been extended to provide a dependency key that is required for other product setups that depend on powerVault

6.2.8 v23.0.6

12-07-2022

Fixed

- *Get-VaultFileBOM*:
 - Instability with still incorrect `Bom_Quantity` and `Bom_ItemQuantity` results after assemblies containing custom Levels of Details are *converted to Model States* in Inventor 2022 or newer.
This fix covers occasional situations (Inventor Vault Add-in duplicates *BomInst* occurrences in BOM blob) which are not covered by *v23.0.2*.
 - Issue with incorrectly calculated `Bom_Quantity` for promoted components in Phantom assemblies that have overridden quantities

6.2.9 v23.0.5

04-07-2022

Fixed

- Compatibility-Issue with Import-Module `powerVault` when called in DataStandard for Inventor customizations, preventing the user from logging in to Vault. Error dialogs with the title “Autodesk Addin” were displayed in Inventor 2022 and later.

Breaking Changes

powerVault Objects: Several properties are now available in correct display language on non-English environments

Multilingual Vault support:

File, Item, Folder, Custom Object and *Change Order* objects provided some Vault System Property names in English instead of the Vault’s language when used in *powerVault Console* or PowerShell IDEs.

The properties are now available via the respective display names and must therefore be adapted on non-English environments.

For example, the following properties have changed on **german** Vault environments:

File object

Previous property name:	Property name to be used:
Änderungsdatum (Date Only)	Änderungsdatum (Nur Datum)
Änderungsdatum (Time Only)	Änderungsdatum (Nur Uhrzeit)
Attachments	Anhänge
Ausgecheckt (Date Only)	Ausgecheckt (Nur Datum)
Ausgecheckt (Time Only)	Ausgecheckt (Nur Uhrzeit)
Eingecheckt (Date Only)	Eingecheckt (Nur Datum)
Eingecheckt (Time Only)	Eingecheckt (Nur Uhrzeit)
Entity Icon	Objektsymbol
Entity Type	Objekttyp
Entity Type ID	Objekttyp-ID
Ersterstellung (Date Only)	Ersterstellung (Nur Datum)
Ersterstellung (Time Only)	Ersterstellung (Nur Uhrzeit)
Erstveröffentlicht (Date Only)	Erstveröffentlicht (Nur Datum)
Erstveröffentlicht (Time Only)	Erstveröffentlicht (Nur Uhrzeit)
Full Path	Vollständiger Pfad
Letztes Freigabedatum (Date Only)	Letztes Freigabedatum (Nur Datum)
Letztes Freigabedatum (Time Only)	Letztes Freigabedatum (Nur Uhrzeit)
Link Target Path	Verknüpfungszielpfad
Path	Pfad
Type Description	Beschreibung des Typs
Vault Status	Tresorstatus
Vault Status Modifier	Vault-Statusmodifikator
Versionserstellungsdatum (Date Only)	Versionserstellungsdatum (Nur Datum)
Versionserstellungsdatum (Time Only)	Versionserstellungsdatum (Nur Uhrzeit)

Item object

Previous property name:	Property name to be used:
Änderungsdatum (Date Only)	Änderungsdatum (Nur Datum)
Änderungsdatum (Time Only)	Änderungsdatum (Nur Uhrzeit)
Attachments	Anhänge
Entity Icon	Objektsymbol
Entity Type	Objekttyp
Entity Type ID	Objekttyp-ID
Ersterstellung (Date Only)	Ersterstellung (Nur Datum)
Ersterstellung (Time Only)	Ersterstellung (Nur Uhrzeit)
Erstveröffentlicht (Date Only)	Erstveröffentlicht (Nur Datum)
Erstveröffentlicht (Time Only)	Erstveröffentlicht (Nur Uhrzeit)
Full Path	Vollständiger Pfad
Letztes Freigabedatum (Date Only)	Letztes Freigabedatum (Nur Datum)
Letztes Freigabedatum (Time Only)	Letztes Freigabedatum (Nur Uhrzeit)
Link Target Path	Verknüpfungszielpfad
Path	Pfad
Type Description	Beschreibung des Typs
Vault Status	Tresorstatus

Folder object

Previous property name:	Property name to be used:
Entity Icon	Objektsymbol
Entity Type	Objekttyp
Entity Type ID	Objekttyp-ID
Erstellt am (Date Only)	Erstellt am (Nur Datum)
Erstellt am (Time Only)	Erstellt am (Nur Uhrzeit)
Folder Create Date	Ordnererstellungsdatum
Full Path	Vollständiger Pfad
Link Target Path	Verknüpfungszielpfad
Path	Pfad
Type Description	Beschreibung des Typs

CustomObject object

Previous property name:	Property name to be used:
Entity Icon	Objektsymbol
Entity Type	Objekttyp
Entity Type ID	Objekttyp-ID
Erstellt am (Date Only)	Erstellt am (Nur Datum)
Erstellt am (Time Only)	Erstellt am (Nur Uhrzeit)
Full Path	Vollständiger Pfad
Link Target Path	Verknüpfungszielpfad
Path	Pfad
Type Description	Beschreibung des Typs

ChangeOrder object

Previous property name:	Property name to be used:
Änderungsdatum (Date Only)	Änderungsdatum (Nur Datum)
Änderungsdatum (Time Only)	Änderungsdatum (Nur Uhrzeit)
Attachments	Anhänge
Entity Icon	Objektsymbol
Entity Type	Objekttyp
Entity Type ID	Objekttyp-ID
Erstellt am (Date Only)	Erstellt am (Nur Datum)
Erstellt am (Time Only)	Erstellt am (Nur Uhrzeit)
Fälligkeitsdatum (Date Only)	Fälligkeitsdatum (Nur Datum)
Fälligkeitsdatum (Time Only)	Fälligkeitsdatum (Nur Uhrzeit)
Full Path	Vollständiger Pfad
Link Target Path	Verknüpfungszielpfad
Path	Pfad
Type Description	Beschreibung des Typs
Übermittlungsdatum (Date Only)	Übermittlungsdatum (Nur Datum)
Übermittlungsdatum (Time Only)	Übermittlungsdatum (Nur Uhrzeit)
Vault Status	Tresorstatus

6.2.10 v23.0.4

22-06-2022

General

- Changed the initialization logic and type of the *\$vaultExplorerUtil* variable (now *ExplorerUtilProxy*) which further reduces the memory usage of the *Open-VaultConnection* cmdlet
- All *Vault Cmdlets* automatically detect and reuse the application's latest Vault connection (and not only when *Connectivity.Application.VaultBase.ConnectionManager.Instance.Connection* is available). This means that *Open-VaultConnection* calls are no longer mandatory in all types of Vault applications.

Fixed

- Compatibility-Issue of all *Vault Cmdlets* with Vault DataStandard customizations, which were not extended to invoke the *Open-VaultConnection* cmdlet without parameters (required since 22.0.7)

6.2.11 v23.0.3

01-06-2022

Features

- *Get-VaultFileBOM* provides Bill of Materials for Inventor files also if their Structured View is disabled.
Note: FileBomRows have empty position numbers and row orders in such situations. In case of ERP integrations, it is recommended to check whether they should be managed specifically (see [powerGate example: Return all BOM rows of a Vault file and warn about disabled Structured View](#)).

General

- Improved and reduced *Get-VaultFileBOM* log messages when Structured BOM Views are disabled in Inventor

Fixed

- *Get-VaultFileBOM* returned rows with incorrect *Bom_ItemQuantity* when assigning Instance Properties to individual components in Inventor 2023 assemblies
- Issue with *Add-VaultFile* logging misleading error messages when uploading new files into Vault

6.2.12 v23.0.2

12-05-2022

Features

- *Get-VaultFileBOM* allows retrieving Bill of Materials for each Model State of Inventor files using the new - *ModelStateType* parameter.

Known issue with Model States and Vault Client 2022

When working with Inventor 2022 it is recommended to install the *Vault 2022.3 Update (Client)* or newer as several issues related to FileBOMs and Model States were addressed in the [Inventor Vault Add-in](#).

General

- *FileBomRow* results return the used model state for all components with a new *Bom_ModelState* property

Breaking Changes

Corrected the type named ‘ReadBomRowVersionType’ to documented type ‘BomRowVersionType’:

The enumeration type of the *Get-VaultFileBom -GetChildrenBy* argument got renamed from *ReadBomRowVersionType* to *BomRowVersionType*.

Fixed

- Instability with *Get-VaultFileBOM* and incorrectly returned *Bom_Quantity* (much too large) and *Bom_ItemQuantity* after assemblies containing custom Levels of Details are converted to Model States in Inventor 2022 or newer.
The issue mainly appeared when the following conditions were met:
 - File contains a Phantom assembly
 - File was checked in with Vault Client 2022.2.2 or Vault Client 2023 (and newer) installed
- “Page not found” error page opens after clicking Help button in Control Panel → Add or Remove Programs

6.2.13 v23.0.1

13-04-2022

General

- Added support for Vault 2023

6.3 powerVault v22

6.3.1 v22.0.9

11-04-2022

General

- *Vault cmdlets* write error logs when no Vault connection is available and the *Open-VaultConnection* cmdlet must be executed

Fixed

- Compatibility-Issue with the *Open-VaultConnection* cmdlet of *powerJobs Processor 22.0.27*

6.3.2 v22.0.8

04-04-2022

Fixed

- Issue with *Get-VaultFileBom* and arguments *-GetChildrenBy LatestVersion*, *LatestReleasedVersion* and *LatestReleasedVersionOfRevision* with erroneously returned Master model state information for non-Master model state components that were removed in referenced Inventor files

6.3.3 v22.0.7

29-03-2022

General

- *Open-VaultConnection* reuses the application's existing Vault connection only when being invoked without parameters.
When new credentials are passed, a new Vault connection will be established.

Fixed

- Increasing memory usage caused by *\$vaultExplorerUtil* with frequent *Update-VaultFile -Properties* or *Open-VaultConnection* calls in long-running processes

Breaking Changes

Compatibility-Issue with powerJobs Processor

This version is not compatible with *powerJobs Processor 22.0.22* or earlier. Therefore powerJobs Processor should be upgraded to a **newer version**.

Alternatively if upgrading powerJobs Processor is not possible, the issue can be workarounded by running *powerVault\Open-VaultConnection* without parameters after the extended *Open-VaultConnection* of powerJobs Processor.

6.3.4 v22.0.6

15-03-2022

Fixed

- *Get-VaultFileBOM* returned wrong *FileBomRows* when non-Master model state components are used in Structured Inventor BOM.
The issue affected all properties with a *Bom_* prefix except of *Bom_RowOrder* and *Bom_PositionNumber*, as the same component's Master model state information was returned for all occurrences.

6.3.5 v22.0.5

07-03-2022

Features

- Added support for *console logs* in **PowerShell ISE**

General

- Updated Licensing to version: *18.2.27*

Fixed

- Vulnerability in *Logging* configuration files by updating *log4net* to v2.0.14 (CVE-2018-1285)
- Issue with *ColoredConsoleAppender* that caused *powershell remote hosts* to crash when appender was logging to console

6.3.6 v22.0.4

29-09-2021

General

- Updated Licensing to version: 18.2.26

6.3.7 v22.0.3

18-08-2021

Features

- Extended cmdlet *Get-VaultFileAssociations* with parameter *-Type* for retrieving directly associated drawing files and other parent file associations

6.3.8 v22.0.2

26-04-2021

Features

- Added support for Vault 2022

General

- Updated Licensing to version: 18.1.24
- End User License Agreement (EULA) has changed

6.4 powerVault v21

6.4.1 v21.0.10

07-04-2021

Features

- New cmdlet: *Get-VaultChangeOrder* to retrieve a Change Order from Vault
- New cmdlet: *Get-VaultChangeOrderAssociations* to retrieve the associations of a Change Order from Vault
- New cmdlet: *Add-VaultChangeOrder* to create a new Change Order in Vault
- New cmdlet: *Update-VaultChangeOrder* for updating a Vault Change Order and manipulating its associations

Fixed

- *Update-VaultFile* and *Update-VaultItem*:
 - The *-Properties* parameters fail to update user-defined properties when values are passed as *PSCustomObject*. This could happen for example, when using brackets in inline declarations of *Hashtables*
 - Updating user-defined properties fails when also system properties are wrongly passed to the *-Properties* parameter
 - *File* and *Item* results are returned even when the update operation fails

- Issue with invalid log message when the passed *Update-VaultItem -Properties* are already up-to-date and therefore do not have to be modified

6.4.2 v21.0.7

20-01-2021

General

- Updated Licensing to version: 18.1.22

6.4.3 v21.0.6

18-12-2020

General

- Updated Licensing to version: 18.1.21
- Copyright notices have changed

Fixed

- Issue that led to an unusable machine and failing Jobs after running JobProcessor for a long time

6.4.4 v21.0.3

20-05-2020

Fixed

- Compatibility-Issue with other coolOrange products using an older *Logging* version

6.4.5 v21.0.2

11-05-2020

Fixed

- Compatibility-Issue where powerJobs Modules could not be imported

6.4.6 v21.0.1

23-04-2020

Features

- Added support for Vault 2021

General

- End User License Agreement (EULA) has changed
- Added Licensing version: 18.1.17
- Added *powerVault Information* shortcut to startmenu
- Removed *powerVault Help* shortcut from startmenu as it can be accessed via powerVault Information shortcut

- Removed Splashscreen

Fixed

- Issue that changing *\$ErrorActionPreference* or passing *-ErrorAction* to the *Open-VaultConnection* cmdlet had no effect

6.5 powerVault v20

6.5.1 v20.0.7

03-07-2019

General

- Improved *Get-VaultFileBom* performance by not including *Parts Only*, *Structured-Multilevel* components and old versions in any Vault API calls
- Improved *Get-VaultItemAssociations* performance by retrieving all files in a single API call

Fixed

- *Get-VaultFileBom* with arguments *-GetChildrenBy LatestVersion*, *LatestReleasedVersion* and *LatestReleased-VersionOfRevision* results in following issues:
 - *VaultServiceErrorException*: “1013” for certain BOMs with *Parts Only*- or *Structured-Multilevel-Views* enabled and *purged* or *deleted* components
 - *InvalidOperationException*: “Sequence contains no elements” for assemblies with Phantoms that got updated with new components that are promoted and merged on higher level
 - *Bom_Quantity* of component on higher level is returned when Quantity is overridden, instead returning sum of quantities with all new components that are promoted from a updated Phantom assembly
 - *ArgumentOutOfRangeException*: “Index was out of range” for assemblies with Phantoms that got updated and have the same Part one time used with type *Normal* and one time used with type *Reference*

6.5.2 v20.0.2

24-05-2019

- Official Release

General

- End User License Agreement (EULA) has changed

6.5.3 v20.0.1beta

11-01-2019

Features

- Added support for Vault 2020 (BETA)

6.6 powerVault v19

6.6.1 v19.0.9

24-04-2018

- Official Release

Features

- powerVault *cmdlets* can be used in every IDE

General

- Updated to PowerShell 4.0
- Removed *powerVault ISE* shortcut from startmenu
- Assembly *coolOrange.VaultServices_[Vault Version]* gets installed into the GAC

6.6.2 v19.0.5beta

12-03-2018

Fixed

- *Get-VaultFileBOM* with arguments *-GetChildrenBy LatestVersion, LatestReleasedVersion* and *LatestReleased-VersionOfRevision* results in following issues:
 - *InvalidOperationException*: “*Sequence contains more than one matching element*” for assemblies with enabled *multilevel* Structured-BOM containing updated sub-assembly with *copied components*
 - *Bom_ItemQuantity* shows *1* for the components of a referenced *Phantom* assembly that got updated, and no new version of the main assembly is checked-in

6.6.3 v19.0.3beta

12-02-2018

General

- Replaced Log4PostSharp with *PostSharp Diagnostics* for extended Debug logging
- Re-enabled extended Debug *Logging* for Vault 2019

Fixed

- Issue where logging did not work when powerVault was used in a 32-Bit process

6.6.4 v19.0.2beta

06-02-2018

Fixed

- *Get-VaultFileBOM* with arguments *-GetChildrenBy LatestVersion, LatestReleasedVersion* and *LatestReleased-VersionOfRevision* results in one of following issues:
 - *NullReferenceException*: “Object reference not set to an instance of an object” occurs when a referenced *Phantom* assembly and it’s *merged* child component are updated, without updating the main assembly
 - *Bom_PositionNumber* is *empty* for the component of a referenced *Phantom* assembly that got updated, and no new version of the main assembly is checked-in

6.6.5 v19.0.1beta

15-01-2018

Features

- Added support for Vault 2019 (BETA)

Warning: Extended Debug <i>Logging</i> is disabled for Vault 2019
--

6.7 powerVault v18

6.7.1 v18.0.19

14-12-2017

General

- Changed *logging* to have a single centralized configuration file and *shared logging assembly* in GAC

Fixed

- Highly increasing memory usage when calling *Cmdlets* very often (*>100 times*)
- *Update-VaultFile* / *Update-VaultItem*: When updating references (*dependencies or attachments*) and some references were already checked-out, then powerVault didn’t change any references and kept the file checked-out.

6.7.2 v18.0.17

24-08-2017

Fixed

- *Get-VaultItem -Properties* didn’t work for all item properties, for instance ‘Description (Item,CO)’, ‘Title (Item,CO)’, ‘Units’, ‘Number’...

6.7.3 v18.0.15

18-08-2017

Fixed

- Fixed typing mistake of special property *Attachments* on PsObject *Comment*
- Extended *PsComment* with support for comments that do not exist in vault

6.7.4 v18.0.12

17-07-2017

General

- Extended with following PsObjects: *Folder*, *Custom Object*, *Change Order*, *Comment*, *Email*

6.7.5 v18.0.10

24-05-2017

Features

- New cmdlet: *Add-VaultJob*
- *Open VaultConnection* now reuses existing connection when application is already connected to Vault

6.7.6 v18.0.8

21-04-2017

- Official Release

General

- Added support for Vault 2018
- Removed support for Vault 2014 & 2015R2
- Introduced extended [/logging`.
- Installed “powerVault 18.0 Logs” shortcut in start-menu section of powerVault
- Changed registry keys to “HKLM\Software\coolOrange s.r.l.\powerVault”: Location and Version

Fixed

- *Add-VaultFile*: issue that local file becomes locked when adding as new file to Vault.
Therefore removing locale file was not always possible immediately after this cmdlet (only if Garbage Collection released file handle).
- *Update-VaultFile*: invalid log error-message “Failed to checkin the file” when *-Comment* parameter is used for released vault files.
The file does not become checked-out and checked-in with the new comment any more, when file is locked.

6.8 powerVault v17

6.8.1 v17.0.24

20-02-2017

Features

- In PowerShell you are now able to change properties of *File*, *FileBomRow*, *Item* and *ItemBomRow*. We have made them settable.

Fixed

- *Get-VaultFileBOM*: Issue when using argument *-GetChildrenBy LatestVersion*. In some specific cases the Cmdlet was not behaving correctly.

6.8.2 v17.0.22

17-02-2017

Features

- *Get-VaultFileBOM*: Now supports ItemQuantity, UnitQuantity as well as TotalQuantity.

Fixed

- Fixed Issue where a RegistryKey was not pointing to the proper installation Directory

6.8.3 v17.0.20

07-02-2017

Fixed

- *Update-VaultFile* deletes CAD-related information on existing fileAssociations when using any “Childs” or “Attachments” parameter

6.8.4 v17.0.18

28-11-2016

- Improved performance for cmdlets returning multiple entities, by fetching all entity properties in a single call
- Properties are retrieved on demand

6.8.5 v17.0.15

30-06-2016

Fixed

- *Get-VaultFileBOM*: Issues with phantom components
- *Get-VaultFileBOM*: Issue with quantity calculation
- *Get-VaultFileBOM*: Issue with merged components

6.8.6 v17.0.12

07-06-2016

- Official Release
- *Get-VaultFileBOM*: supports reference components
- *Get-VaultFileBOM*: supports phantom components

6.8.7 v17.0.6beta

22-03-2016

Fixed

- Issue in obfuscation

6.8.8 v17.0.4beta

07-03-2016

Features

- Added support for Vault 2017 (BETA)
- Added registry keys under “HKCU\Software\coolOrange s.r.l.\powerVault”: Location and Version
- Added Windows 10 support
- Applied new coolOrange standards

Fixed

- Install / Upgrade with same version

6.9 powerVault v16

6.9.1 v16.1.21

23-10-2015

Features

- New cmdlet: *Get-VaultItemAssociations*
- New cmdlet: *Save-VaultFile*
- *Update-VaultItem* with new parameter: `-NewNumber`, `-Attachments`, `-AddAttachments` and `-RemoveAttachments`

6.9.2 v16.0.42

23-07-2015

Features

- Vault Entities have additional system properties with a prefix “_”

Fixed

- *Update-VaultFile/Update-VaultItem -Properties* should accept systemName properties
- *Get-VaultFileBom* should handle virtual components and purchased components from inventor BOM

6.9.3 v16.0.38

01-07-2015

Features

Show-Inspector:

- Add a Search toolbar to be able to search for properties / values
- renamed parameter *-ElementToHighlight* to *-Highlight*
- *-Highlight* parameter is now a positional parameter

Fixed

- *profile.ps1* is now handled correctly on uninstall / upgrade
- *Update-VaultFile -Status* and *-Category* sets first state and than category
- *Get-VaultFileBom* failes when in inventor file iProperties with same name as system properties are added



powerVault is a native PowerShell extension that allows you to talk with Vault via pure PowerShell [cmdlets](#).

With powerVault, you get command-lets for dealing with files and items

For instance *Open-VaultConnection* for establishing a connection to Vault, or *Get-VaultFile* for getting/downloading a file, or *Get-VaultItem* for getting an item and many more.

Such command-lets make it very easy to either get information from Vault or creating and updating information in Vault.

You don't have to deal with the complexity of the Vault API, instead you can just execute the command you like and let the command-let do the rest.