



Elysium
3DxSUITE™

Elysium 3DxSUITE
Data Package Studio Manual

Version EX9.1 2022

Elysium Co. Ltd.

Table of Contents

1. About This Documentation	1
2. What is New in Data Package Studio?	1
3. What is Data Package Studio?	7
4. Installation	8
4.1. Prerequisites	8
4.2. Install/Update 3DxSUITE	8
4.3. Install/Update Data Package Studio	8
4.4. Start Data Package Studio	9
5. Generate Your First TDP	9
6. How-to Guides	11
6.1. How-to Guides: Basic Functionality	11
6.1.1. How-to Add a Data Source	12
6.1.2. How-to Add and Customize a 3D View Widget	14
6.1.3. How-to Add and Customize a View Carousel Widget	15
6.1.4. How-to Add a Button Widget	16
6.1.5. How-to Add a Text Field Widget	17
6.1.6. How-to Add a PMI Table Widget	18
6.1.7. How-to Add a BoM Table Widget	19
6.1.8. How-to Add a Custom Table Widget	20
6.1.9. How-to Add a User Property Table Widget	21
6.1.10. How-to Add a Table Column Filter Widget	22
6.1.11. How-to Customize a Widget	23
6.1.12. How-to Add a Trigger/Action Connection	25
6.2. How-to Guides: Use Cases	25
6.2.1. How-to Create a TDP Template	25
6.2.2. How-to Customize a CAD Validator Report	28
6.2.3. How-to Include/Exclude Content in a Table or View Carousel	33
6.2.4. How-to Add a Bill of Characteristics Table	36
6.2.5. How-to Save Table Data using Elysium's TDP JavaScript API	41
6.2.6. How-to Create a Work Instructions TDP	44
6.2.7. How-to Enable Printing of Saved Views	52
6.2.8. How-to Add a 3D Toolbar using Elysium's TDP JavaScript API	53

6.3. How-to Guides: TDP Generation.....	59
6.3.1. How-to Generate a TDP with 3DxSUITE SmartLauncher	59
6.3.2. How-to Generate a TDP with 3DxSUITE SmartController.....	60
6.3.3. How-to Generate a TDP with 3DxSUITE TransServer	61
7. Reserved PDF Field Names	61
8. Frequently Asked Questions	62
8.1. Definitions	62
8.1.1. What is DPS?	62
8.1.2. What is a formula?.....	62
8.1.3. What is a data source?.....	62
8.1.4. What is a widget?	63
8.1.5. What is a trigger?	64
8.1.6. What is an action?	64
8.1.7. What triggers and actions are supported?	65
8.1.8. Why does Data Package Studio save three files?	68
8.1.9. What is a predefined property?	68
8.1.10. What is a user-defined property?.....	68
8.1.11. What is the difference between pre- and user-defined properties?	68
8.2. Extending Functionality.....	68
8.2.1. Can I extend the functionality of Data Package Studio?	68
8.2.2. Where do I find Data Package Studio API documentation?.....	68
8.3. Formatting and Customization.....	68
8.3.1. Can I add attachments to a TDP in batch?	68
8.3.2. Can I change the background color of the tables?	69
8.3.3. Can I change the formatting of elements/widgets in a CAD Validator report?	69
8.3.4. Can I change the size of table columns and rows in a TDP?	69
8.3.5. Can I remove columns of a table in the CAD Validator report?.....	69
8.3.6. Can I scroll in the tables generated by Elysium?	69
8.3.7. Where can I find the "fit all" and "clear selection" buttons that the CAD Validator report 3D view has?	69
8.3.8. Why is the View Carousel missing thumbnail images?.....	69
8.3.9. Will the file name of the attachment be imported automatically?.....	70
8.4. Installation and License	70

8.4.1. Do I need a license to use Data Package Studio?	70
8.4.2. Where can I find the Adobe Acrobat plug_ins Folder?	70
8.5. PDF	70
8.5.1. How do I add an attachment to a PDF?	70
8.5.2. How do I add a date field to a PDF?	70
8.5.3. How do I add missing fonts to a PDF?	71
8.5.4. How do I add or organize pages in a PDF?	71
8.5.5. How do I add tabs/buttons to a PDF?	71
8.5.6. How do I add a watermark to a PDF?	72
8.5.7. How do I modify or create editable fields in PDF?	72
8.5.8. How do I create a template?	72
8.5.9. How do I define/set static text and images in PDF?	72
8.5.10. How do I digitally sign a PDF?	72
8.5.11. How do I enable 3D content in a PDF?	72
8.5.12. How do I protect a PDF with a password?	72
8.5.13. Where do I find attachments in a PDF, and how do I open them?	73
8.6. User Interface	73
8.6.1. Where do I find more information about the UI?	73
9. Example Templates and Data Sources	73
9.1. Example Files Setup	73
9.1.1. Copy Example Files	73
9.1.2. Copy Example Scenarios	74
9.1.3. Configure 3DxSUITE SmartLauncher to Run Example Scenarios	74
9.2. Example Files	75
9.2.1. Data Sources	75
9.2.2. Formulas	76
9.2.3. Output	77
9.2.4. Scenarios	77
9.2.5. Scripts	78
9.2.6. Templates	79
Appendix A: 3DxSUITE Parameters	80
A.1. Required Parameters for the 3D PDF Component	80
A.2. Optional Parameters for the 3D PDF Component	80
A.3. Required Parameters for CAD Validator Component	81

A.4. Optional Parameters for CAD Validator Component	82
A.5. Optional 3DxSUITE Parameters for CAD Translation.....	82
Appendix B: Predefined Model Properties	84
Appendix C: Predefined PMI Properties	90
C.1. Common Properties	90
C.2. Datum Properties.....	91
C.3. Datum Target Properties	91
C.4. Dimension Properties.....	92
C.5. Geometric Tolerance Properties	94
C.6. Line Weld Properties	95
C.7. Locator Properties.....	97
C.8. Roughness Properties.....	98
C.9. Spot Weld Properties	98
C.10. Bill of Characteristic Properties.....	99
Appendix D: Minimum Widget Sizes	101
Appendix E: Elysium TDP JavaScript API	106
E.1. Class: Ely3dpdf.....	106
E.1.1. Function: getApi.....	106
E.2. Class: ElyTdpApi	107
E.2.1. Function: getField	107
E.2.2. Function: getWidget	107
E.3. Class: ElyTdpWidget	108
E.3.1. Function: getType.....	108
E.4. Class: ElyTdp3dViewWidget	108
E.4.1. Function: reset	108
E.4.2. Function: fitAll	109
E.4.3. Function: fitSelection	109
E.4.4. Function: clearSelection	109
E.4.5. Function: getRenderModes	110
E.4.6. Function: setRenderMode	110
E.4.7. Function: resetRenderMode	111
E.4.8. Function: isValidRenderMode.....	111
E.4.9. Function: getViews.....	112
E.4.10. Function: setView	112

E.4.11. Function: resetView	113
E.4.12. Function: isValidView	113
E.4.13. Function: getShowPmi	113
E.4.14. Function: setShowPmi	114
E.5. Class: ElyTdpTableWidget	114
E.5.1. Function: clearFiltering	114
E.5.2. Function: clearSelection	115
E.5.3. Function: getData	115
E.5.4. Function: getHeader	116
E.5.5. Function: reset	117
E.5.6. Function: resetSorting	117
E.6. Class: ElyTdpTextWidget	117
E.6.1. Function: getReadOnly	117
E.6.2. Function: setReadOnly	118
E.6.3. Function: toggleReadOnly	118
Appendix F: Migrating to the New CAD Validator Report Template Format	120
F.1. CAD Validator Widget Types	120
F.1.1. 3D View (CAD Validator)	121
F.1.2. Assembly Tree (CAD Validator)	122
F.1.3. CAD Information (CAD Validator)	122
F.1.4. Color Bar (CAD Validator)	122
F.1.5. Component Summary (CAD Validator)	123
F.1.6. Detail List (CAD Validator)	123
F.1.7. Difference Summary (CAD Validator)	125
F.1.8. Execution Property (CAD Validator)	125
F.1.9. Property Table (CAD Validator)	126
F.1.10. Type Summary (CAD Validator)	126
F.1.11. Validation Setting (CAD Validator)	127
F.2. Merge the CAD Validator 3D View Widget Related Placeholders	127
F.3. Rename the CAD Validator Widget Placeholders	128
F.4. Create a Formula in Data Package Studio	129
Appendix G: Example TDP Pre-Process Script	130
Appendix H: Known Issues	132

1. About This Documentation

This documentation is for you if you are looking for ways to customize a Technical Data Package¹ (TDP), and/or:

- It is your first time using Data Package Studio (DPS).
- You want to know more about the many features of DPS.

If you are an administrator looking for information about how to set up 3DxSUITE, please refer to the 3DxSUITE documentation. You should find the documentation at "<INSTALL_FOLDER>\doc" - e.g., "C:\Program Files\Elysium\3DxSUITE\doc".

1: A Technical Data Package contains product information that is adequate and relevant, e.g., for acquisition, production, engineering, and logistics.



Access our online documentation (<https://doc.elysium-global.com/>) for the latest updates (ask your administrator for the login credentials).

2. What is New in Data Package Studio?

New in Data Package Studio EX9.1

EX9.1 is a major release that provides the following enhancements:

- Adobe Acrobat 64bit
 - Enabled Data Package Studio to run in Adobe Acrobat 64bit - use the 64bit installer.
 - Enabled generated 3D PDF files to open and behave as expected in Adobe Acrobat 64bit.



Highlighting PMI related geometry and element color in views may not work as expected in Adobe Acrobat 64bit - see [Known Issues](#).

- CAD Validator
 - Enabled option to show/hide PMI in 3D View at startup - to eliminate the PMI "furball".
 - Enabled opening a URL from the Property Table (when provided by a diff property).
 - Note that this requires customization of CAD Validator comparison.
 - Useful if you, e.g., want to provide documentation related to an element type to simplify report reviewing.
 - Enabled alignment with view direction when selecting a group that includes one view.
 - Enabled sorting of groups.
 - Removed fitting and highlighting of differences when selecting the Ungrouped group.
- Work Instructions (enabled by many of the Table enhancements below)
 - Enabled displaying Work Instruction steps in Custom Table.

- Enabled displaying Work Instruction steps related information in:
 - A Text Field, e.g., step description.
 - A Custom Table, e.g., tools, parts, and safety instructions associated with a step.
- Enabled displaying 3D views associated with a Work Instruction step.

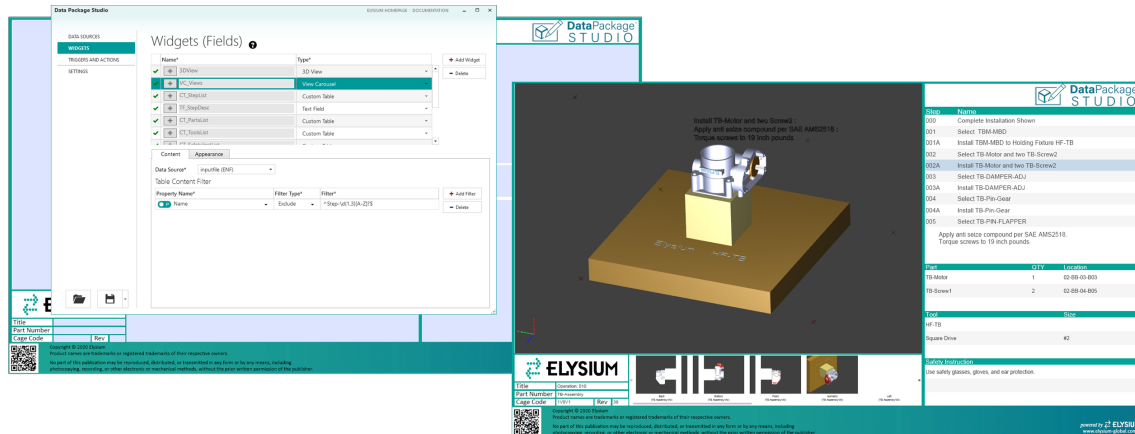


Figure 1. Example of a generated Work Instruction TDP (right) and how to filter views related to a step (left) - all steps matching "Step-" + 1 to 3 digits + (optionally) an uppercase character.

• Table

- Enhanced filtering.
 - Enabled Regular Expression for all table filtering.
 - Enabled content filtering - control the content of a table by specifying exclude/include filters.
 - Enabled "drill down" column filtering.
 - Applies a filter on a table column - click the header and select the filter option.
 - Filters on multiple columns result in a "drill down" where only the content matching all filters are displayed.
 - Enabled Custom Table filtering based on the item selected in another Custom Table.
 - Applies the filter specified in a property of the second (triggering) table.
 - Useful in combination with the "empty Custom Table" enhancement below to display information in a 1:n relationship.
 - Enabled Table Column Filter on multiple columns - items that match the filter for one or more of the specified columns are displayed.
- Enhanced sorting - click the header and select the sort order A to Z, or Z to A.
- Enabled selecting a 3D view based on the item selected in a Custom Table.
 - Applies the view name specified in a property of the custom table to a 3D view.
 - Useful when you want to associate a specific view with an item in a Custom Table.
- Enabled displaying the property of a selected Custom Table item in a Text Field.

- Enabled displaying a Custom Table as empty and only display items that matches an applied filter.
 - When specified (opt-in), the table is empty at startup and when unfiltered.
 - Useful in combination with the "Custom Table filtering" enhancement above.
- Enabled physical properties (e.g., mass and volume) to be displayed in a BoM Table.
- Extended support for character encoding of Text data sources (same support as for a CSV data source).
- JavaScript API
 - Enabled lock/unlock of editable text fields.

You will also see the following changes in the documentation.

Table 1. New in Data Package Studio EX9.1 documentation.

New/Added Section	Note
How-to Include/Exclude Content in a Table or View Carousel	Added to describe how to use filters to include or exclude content in Tables and View Carousel widgets.
How-to Create a Work Instructions TDP	Added to explain how to create a Work Instructions TDP.
Minimum Widget Sizes	Added to clarify the minimum width and height for widgets to be created.

Table 2. Updated in Data Package Studio EX9.1 documentation.

Updated Section	Note
How-to Customize a CAD Validator Report	Updated to include the new Show PMI (at startup) option, and added tip for how to specify TDP generation parameters in the CAD Validator component.
Predefined Model Properties	Added description of how 3DEXPERIENCE / V6 properties are mapped to DPS properties.
Class: ElyTdpTextWidget	Added description of the Text Field JavaScript API to unlock and lock editing.

Updated Section	Note
Installation	Updated to reflect the latest changes to Data Package Studio, e.g., version references, new table content filter, and changes to example files like the use of scenario parameter sets.
How-to Add and Customize a View Carousel Widget	
How-to Add a PMI Table Widget	
How-to Add a BoM Table Widget	
How-to Add a Custom Table Widget	
How-to Add a User Property Table Widget	
How-to Add a Table Column Filter Widget	
How-to Add a Bill of Characteristics Table	
How-to Save Table Data using Elysium's TDP JavaScript API	
How-to Enable Printing of Saved Views	
How-to Add a 3D Toolbar using Elysium's TDP JavaScript API	
How-to Generate a TDP with 3DxSUITE SmartLauncher	
How-to Generate a TDP with 3DxSUITE SmartController	
Example Templates and Data Sources	
3DxSUITE Parameters	
Known Issues	

New in Data Package Studio EX9.0.5



The EX9.0.5 release is a backend (3DxSUITE 3D PDF adapter) update, which means it does not come with a Data Package Studio installer.



See [Installation](#) section for details on how to install/update 3DxSUITE and how to update to Data Package Studio EX9.0 from an older version.

EX9.0.5 provides the following enhancements:

- Simplified the workflow of using CAD Validator model input files as data sources for non-CAD Validator widgets in Data Package Studio.
- Enhanced handling of assembly/part instances (e.g., for more accurate visibility in views).

We have also fixed the following issues:

- Unexpected clipping in views that could cause elements to partly or completely disappear.
- Unexpected visibility of PMI items in exploded views.

You will also see the following changes in the documentation.

Table 3. New in Data Package Studio EX9.0.5 documentation.

New/Added Section	Note
How-to Name an ENF Data Source	Added to clarify naming convention of ENF data sources generally and how to access CAD Validator input files specifically.

Table 4. Updated in Data Package Studio EX9.0.5 documentation.

Updated Section	Note
How-to Customize a CAD Validator Report	Added a tip about accessing CAD Validator input files as data sources in Data Package Studio.
Installation	Updated to reflect the latest changes to Data Package Studio.
Known Issues	

New in Data Package Studio EX9.0

Data Package Studio EX9.0 provides the following enhancements:

- CAD Validator widgets customization (show/hide) - see example in figure below and [How-to Customize a CAD Validator Report](#).
- 3D View widget:

- Support for "exploded" views.
- Option to specify the default view.
- Option to specify visibility (show/hide) of cross sections.
- Parameter (SectionCapping) to control the capping of cross sections - see [3DxSUITE Parameters](#).
- Parameter (XConvertFlatToScreenPMI) to control the conversion of flat-to-screen PMIs - see [3DxSUITE Parameters](#).
- Text Field widget:
 - Option to enable editing of text fields.
- Fixed an issue where the model default color was used instead of the volume color for faces without a specified color.
- Parameter "Script=BasicTDP" has been deprecated.



Scenarios using the "Script=BasicTDP" parameter value will not break for now. They will run using the "Script=Simple" parameter value and generate a warning that BasicTDP is no longer supported.



Update your scenarios to use the Simple script or your own template/formula instead of the BasicTDP script.

You will also see the following changes in the documentation.

Table 5. New in Data Package Studio EX9.0 documentation.

New/Added Section	Note
Migrating to the New CAD Validator Report Template Format	Added to describe how to migrate a CAD Validator report template to the new format (only required if you want to customize widgets).

Table 6. Updated in Data Package Studio EX9.0 documentation.

Updated Section	Note
How-to Customize a CAD Validator Report	Updated to align with Data Package Studio's ability to customize CAD Validator widgets.
How-to Add and Customize a 3D View Widget	Updated to cover how to specify a default view and show/hide cross sections.
How-to Add a Text Field Widget	Updated with description of how to enable editing of a Text Field widget.

Updated Section	Note
3DxSUITE Parameters	Updated to reflect the latest changes to Data Package Studio.
Installation	
Known Issues	

3. What is Data Package Studio?

Data Package Studio (DPS) is a tool that helps you customize how 3DxSUITE generates a TDP based on a template and selected data sources. It is used in one of the three steps required to produce a TDP - see figure below.

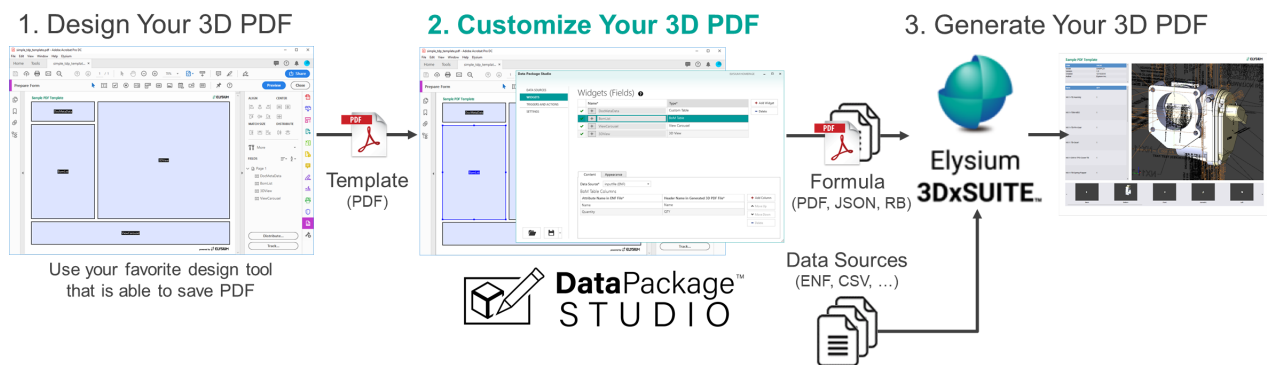


Figure 2. Data Package Studio customizes how 3DxSUITE generates a TDP.

In DPS, you can customize the formula to generate a TDP by specifying:

1. What data sources to bind to which data placeholders (text fields) in your template.
2. How data placeholders should present their data (e.g., BoM list, 3D view).
3. How data placeholders should behave when a user interacts with your TDP, e.g., highlighting of a part in a 3D view when it is selected in a BoM list.

The "magic" happens when you input your formula in 3DxSUITE together with the files you want to use as your data sources. The result is a shiny new TDP.

4. Installation

4.1. Prerequisites

Table 7. Prerequisites to install Data Package Studio successfully.

OS	Version: <ul style="list-style-type: none"> • Windows 10 (32bit/64bit) • Windows 11 (32bit/64bit)
Adobe Acrobat¹	<ul style="list-style-type: none"> • Adobe Acrobat Pro DC Continuous Release • Adobe Acrobat Pro 2020 Release (Classic) • Adobe Acrobat Pro 2017 Release (Classic)

1: Adobe Acrobat is a registered trademark of Adobe Inc.

4.2. Install/Update 3DxSUITE

To enable the latest feature(s) of Data Package Studio when generating your TDP you need 3DxSUITE EX9.1.



You need *administrator privileges* to install/update 3DxSUITE.

To update/install you:

1. Launch "**Elysium_3DxSUITE_EX*.*)_Components.exe**"¹ from your 3DxSUITE install folder.
2. Follow the installer instructions until finished.

1. The double asterisks (*.*) refer to the 3DxSUITE version, e.g., 9.1.

4.3. Install/Update Data Package Studio

To install/update Data Package Studio you:

1. Launch (from the "**Utility_Tool**" folder):
 - For Acrobat Pro 32bit:
"**Elysium_3DxSUITE_EX*.*)_Data_Package_Studio_for_Acrobat_32bit.exe**"¹
 - For Acrobat Pro 64bit:
"**Elysium_3DxSUITE_EX*.*)_Data_Package_Studio_for_Acrobat_64bit.exe**"¹
2. Follow the installer instructions until finished.

1. The double asterisks (*.*) refer to the 3DxSUITE version, e.g., 9.1.



1. You need *administrator privileges* to install Data Package Studio.
2. Install Data Package Studio in the [Adobe Acrobat's "plug_ins" folder](#).

4.4. Start Data Package Studio

You start Data Package Studio in two simple steps:

1. Start Adobe Acrobat.
2. Select **Elysium** › **Start Data Package Studio** from the top menu bar.

You are now ready to [Generate Your First TDP](#).

5. Generate Your First TDP

This section will walk you through the steps to generate your first TDP within 10 minutes.

Prerequisites:

- Data Package Studio (DPS) is set up.
- 3DxSUITE SmartLauncher (SLS) is set up.

Example Files:

You will use the files listed in the table below. You can find the files in the "**examples**" folder next to this document.

File	Description
" templates\basic_how_to_guide_template.pdf "	Template file.
" data_sources\example_throttle_body_r01.enf "	Model input.

The video below will guide you through the steps required to customize and generate a TDP.

Steps:

Customize your TDP template with Data Package Studio:

1. Open [basic_how_to_guide_template.pdf](#) (your template) in Adobe Acrobat.
2. Start DPS (click **Elysium** › **Start Data Package Studio**).
3. In the [**Data Sources**] tab/page, add a ENF type of data source and name it **inputfile**.
4. In the [**Widgets**] tab/page:
 - a. Add a **BoM Table** widget (it should have **inputfile (ENF)** as its data source and columns for **Part Quantity**, **Part Number**, and **Part Material**).

- b. Add a **3D View** widget (it should have **inputfile (ENF)** as its data source).
- c. Add a **View Carousel** widget (it should have **inputfile (ENF)** as its data source).

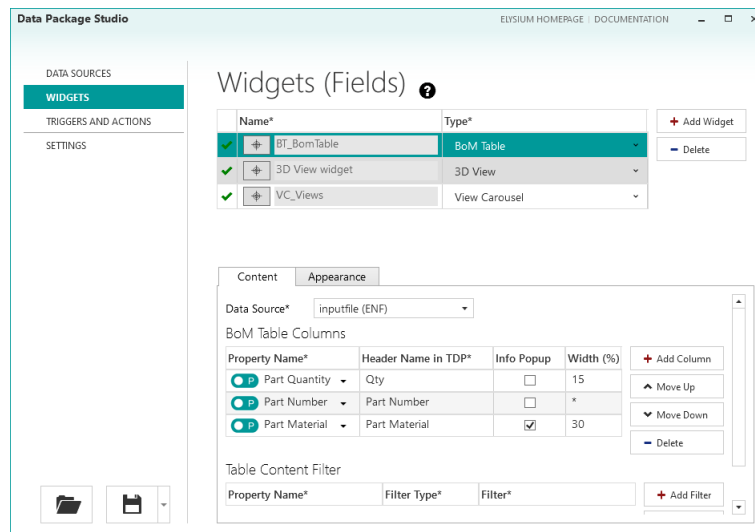


Figure 3. Example of the widgets you have just added.

5. In the **[Triggers and Actions]** tab/page, add a trigger/action that connects the BoM Table widget (trigger) and the 3D View (target) - this enables clicking an item in a BoM table to zoom into it in a connected 3D view.

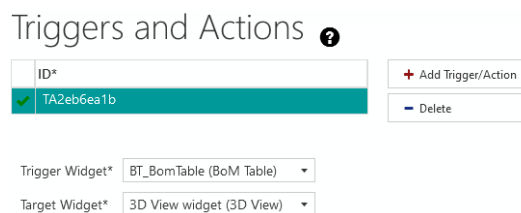


Figure 4. Example of the trigger/action you have just added.

6. Save your TDP customization formula to, e.g., **"C:\dps\my_formula.rb"**.

Generate your TDP with SLS:

1. Open **example_throttle_body_r01.enf** in SLS (right click the file and select **Elysium > Convert/Change File Type...**).
2. Set the 'Save as type' to **3D PDF**.
3. Configure the 3D PDF options (click **[Option...]**):
 - a. Set the 'Script File for 3D PDF Export (Script)' to **Custom**.
 - b. Set the 'Customized Script File | Absolute Path (CustomizedScript)' to the location of your formula, e.g., **"C:\dps\my_formula.rb"**.
 - c. Set any additional parameters to your liking (see **3DxSUITE Parameters** for details).
 - d. Confirm and save the options.
4. Save the 3D PDF to a location of your liking, e.g., **"C:\dps\my_tdp.pdf"**

► <https://vimeo.com/341267962/be3f3fb882> (Vimeo video)

6. How-to Guides

In this section you will find how-to guides that will help you do more advanced formula creation and TDP generation. Most of the guides will take less than 5 minutes to complete.



You can find all template and example files used in the guides in the "examples" folder next to this document. See [Example Files Setup](#) for how to use them.

You will need Data Package Studio and 3DxSUITE installed (and properly set up) to complete most of the guides. Other prerequisites will be listed in the respective guide.

We recommend that you complete the [Generate Your First TDP](#) guide before starting here. This will help you understand the general workflow of DPS and how to generate a TDP to view the result.

The how-to guides have three parts:

- [How-to Guides: Basic Functionality](#) - covers the basics, e.g., adding a [BoM Table widget](#).
- [How-to Guides: Use Cases](#) - covers common uses cases, e.g., [How-to Customize a CAD Validator Report](#).
- [How-to Guides: TDP Generation](#) - covers generation, e.g., with [3DxSUITE SmartController](#).

6.1. How-to Guides: Basic Functionality

The how-to guides in this section will walk you through the basic functionality of Data Package Studio (DPS). Use the example files below when you follow the guides and as a starting point to explore all the customization options in DPS.



Modify the "basic_how_to_guide_formula.rb" to try different customization options.

Example Files:

File	Description
"templates\basic_how_to_guide_template.pdf"	Template file.
"formulas\basic_how_to_guide_formula.rb"	Formula file.
"data_sources\example_throttle_body_r01.enf"	Model input.
"data_sources\example_revision_history.csv"	Revision history input.

File	Description
"scenarios\dps_scenario.esa"	Scenario file with the How-To Guide Formula parameter set.

6.1.1. How-to Add a Data Source


In this how-to guide you will learn how to add a data source to your customization formula using Data Package Studio (DPS). A data source provides the content for your TDP. It is used as an input for **widgets** and/or as an attachment.

We are going to walk you through the steps to add two **ENF** files and a **CSV** file. We will also add the CSV file as an attachment to the TDP.

An ENF file contains part and 3D data that you may want to display, e.g., in a **BoM Table** widget, a **3D View** widget, or **PMI Table**. It also contains predefined properties (e.g., Part Number) and user-defined properties that you may want to present in separate **Text Field** widgets. The CSV file may contain information that you would like to add to a **Custom Table** widget.

Steps:

1. Open your TDP template in Adobe Acrobat and start DPS.
2. Click the **[Data Sources]** tab/page.
3. Add your first ENF data source and set its ID to **inputfile**.
 - a. Click the **[+ Add Data Source]** button.
 - b. Confirm that the Type is **ENF** and that the ID is **inputfile**.
4. Add your second ENF data source, which you can name as you like (e.g., **enf2**).
 - a. Click the **[+ Add Data Source]** button.
 - b. Confirm that the Type is **ENF**.
 - c. Change the ID to **enf2**.
5. Add your CSV data source, which you can also name as you like (e.g., **revision_history**).
 - a. Click the **[+ Add Data Source]** button.
 - b. Change the Type to **CSV**.
 - c. Change the ID to **revision_history**.
 - d. Change the Encoding to, e.g., **UTF-8** (depending on your CSV).

Data Sources 

ID*	Type*	Encoding*	Attachment	Description
✓ inputfile	ENF ▾	-	<input type="checkbox"/>	
✓ enf2	ENF ▾	-	<input type="checkbox"/>	
✓ revision_history	CSV ▾	UTF-8 ▾	<input checked="" type="checkbox"/>	

Figure 5. Three added data sources.

6. Add the **revision_history** data source as an attachment.
 - a. Check the "Attachment" checkbox to add the data source as an attachment.

Data Sources ?

ID*	Type*	Encoding*	Attachment	Description
✓ inputfile	ENF	-	<input type="checkbox"/>	
✓ enf2	ENF	-	<input type="checkbox"/>	
✓ revision_history	CSV	UTF-8	<input checked="" type="checkbox"/>	

Figure 6. The custom_data data source will be added as an attachment.

That is it! You now have three different data sources that you can use to customize your TDP.

How-to Name an ENF Data Source

DPS uses a simple and important naming convention for ENF data sources.

If you are customizing a CAD Validator (CV) report and want to use a CV input file as a data source in DPS, then you need to name it **inputfile1** (CV source) or **inputfile2** (CV target) - see the figure below.

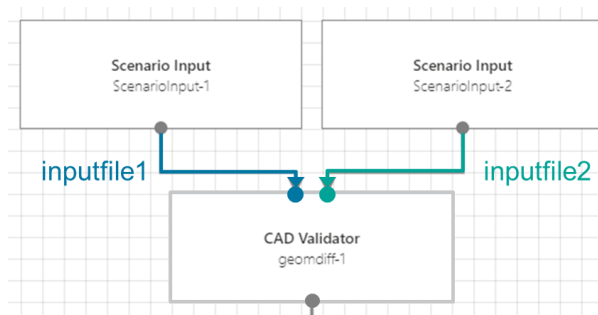


Figure 7. CAD Validator's source and target files are mapped to **inputfile1** and **inputfile2** respectively.

You only need to explicitly add a CV input file as a data source if you want to use the ENF for non-CV widgets. All the CV widgets will use the **CAD Validator Result** data source type as input. So, if you are customizing a CV report and want to use both CV input files as data sources in DPS, then you should have a set of data sources similar to the figure below.

Data Sources ?

ID*	Type*	Encoding*	Attachment	Description
✓ ely_cv_result_source	CAD Validator Result	-	<input checked="" type="checkbox"/>	
✓ inputfile1	ENF	-	<input type="checkbox"/>	
✓ inputfile2	ENF	-	<input type="checkbox"/>	

Figure 8. Example of using both the CV input files as ENF data sources in DPS.



See also [How-to Customize a CAD Validator Report](#).

For all other cases, one of your ENF data sources needs to be named **inputfile**. Additional ENF data sources can be named anything as long as the name is unique. So, if you have two ENF files as input you should have something that looks like the figure below.

Data Sources ?

	ID*	Type*	Encoding*	Attachment	Description
✓	inputfile	ENF ▾	-	<input type="checkbox"/>	
✓	enf2	ENF ▾	-	<input type="checkbox"/>	

Figure 9. Example of using two ENF data sources in DPS.

6.1.2. How-to Add and Customize a 3D View Widget

In this how-to guide you will go through the steps of adding and customizing a 3D View widget. A 3D View widget takes an [ENF](#) input and displays its content as 3D geometry, annotations, etc. It enables users of your TDP to interact with the 3D content, such as by rotating and zooming.

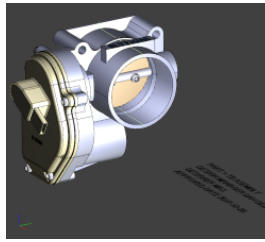


Figure 10. Example of a 3D view.

When adding a 3D View widget, a trigger/action pair between itself (action) and a [View Carousel](#) widget (trigger) connected to the same data source is also added. The trigger/action pair enables users of a TDP to select which saved view to display in the connected 3D view.



Go through [How-to Add a Data Source](#) before you start this how-to guide.

Steps:

1. Open your TDP template in Adobe Acrobat and start DPS.
2. Add a ENF data source and name it **inputfile**.
3. Add a 3D View widget.
 - a. Click the [+ Add Widget] button.
 - b. Select **3D View** as the type of the new widget.
 - c. Connect the new 3D View widget with its placeholder in the TDP template.

Widgets (Fields) ?

Name*	Type*	
✓ [+] 3D View widget	3D View ▾	<input type="button" value="+ Add Widget"/> <input type="button" value="- Delete"/>
✓ [+] VC_VIEWS	View Carousel ▾	
✓ [+] PT_PmiTable	PMI Table ▾	

Figure 11. Adding a 3D View widget.

4. Specify the 3D View widget's content.
 - a. Select **inputfile** (ENF) as the data source.
 - b. Specify the default view (i.e., the view that will be displayed when the TDP is opened).¹
 - i. **First View (ordered by name)** is the first view found when sorting views by name in

alpha-numerical order.

- ii. **CAD Default View** is the view specified as the default in the original CAD system.
- iii. **Named View** is the first view found that matches the specified name.

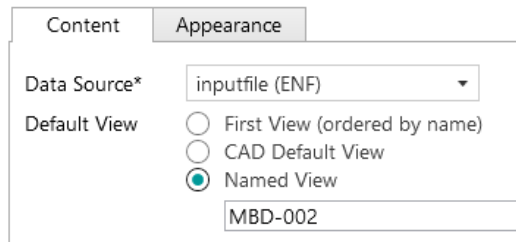


Figure 12. Specifying the contents of the 3D View widget.

1) If a view cannot be found, the fallback is to use the first view in chronological order and then, if not found, a generated ISO view.

5. Customize the 3D View widget's appearance.

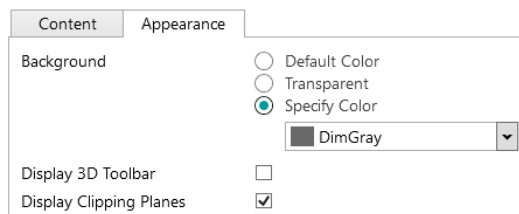


Figure 13. Specifying the appearance of the 3D View widget.

6.1.3. How-to Add and Customize a View Carousel Widget

In this how-to guide you will go through the steps of adding and customizing a View Carousel widget. A View Carousel widget displays the saved views of a 3D model as thumbnails in a grid layout. It takes an **ENF** data source type as input.

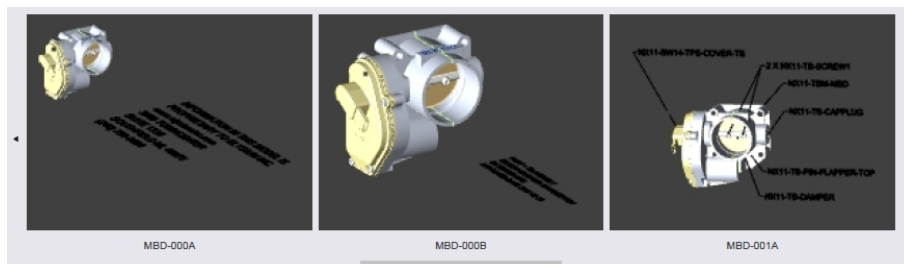


Figure 14. Example of a View carousel.

A View Carousel widget can be used to control which saved view to display in a **3D View** widget. This is enabled automatically when the View Carousel widget, the 3D View widget, and a **BoM Table** widget is connected to the same data source and there is a trigger/action connection between the Bom Table widget (trigger) and the 3D View widget (action).

The video below will take you through the following steps.

Steps:

1. Open your TDP template in Adobe Acrobat and start DPS.

2. Add a ENF data source and name it **inputfile**.
3. Add a View Carousel widget.
 - a. Click the [+ **Add Widget**] button.
 - b. Select **View Carousel** as the type of the new widget.
 - c. Connect the new View Carousel widget with its placeholder in the TDP template.
4. Specify the View Carousel widget's content (select **inputfile (ENF)** as the data source).
5. Add, optionally, **filters** to include/exclude content.
6. Customize the View Carousel widget's appearance.

► <https://vimeo.com/373805643/7c36189138> (Vimeo video)

6.1.4. How-to Add a Button Widget

In this how-to guide you will learn how to add a Button widget. A Button can be used to trigger actions such as printing saved views as described in [How-to Enable Printing of Saved Views](#). It does not take a data source type as input.

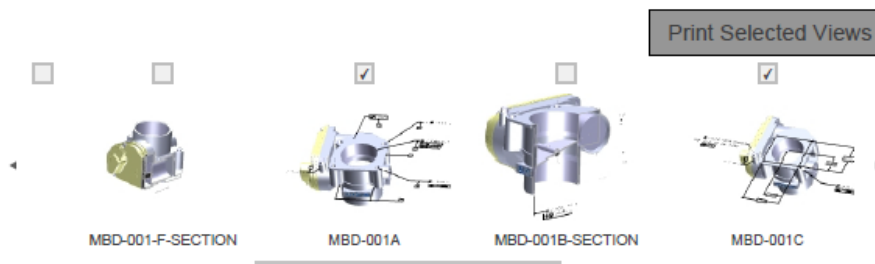


Figure 15. Example of a button to print selected views.

Steps:

1. Open your TDP template in Adobe Acrobat and start DPS.
2. Add a Button widget.
 - a. Click the [+ **Add Widget**] button.
 - b. Select **Button** as the type of the new widget.
 - c. Connect the new Button widget with its placeholder in the TDP template.

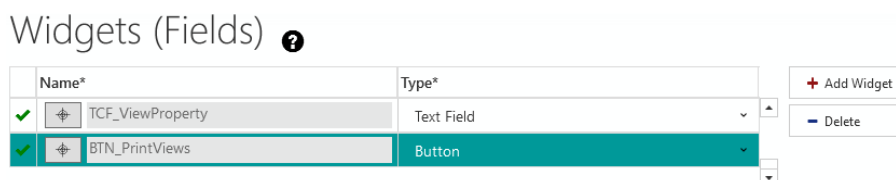


Figure 16. Adding a Button widget.

3. Specify the Button widget's label (e.g., **Print Selected Views**).

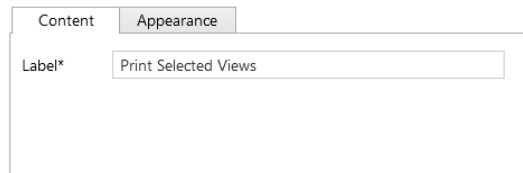


Figure 17. Specifying the Button widget's label.

4. Customize the Button [widget's appearance](#).
5. Add a Trigger/Action (see [How-to Add a Trigger/Action Connection](#) for details).

6.1.5. How-to Add a Text Field Widget

In this how-to guide you will learn how to add a Text Field widget. A Text Field widget presents the content of a data source as plain text in a TDP. It takes a [ENF](#) data source type and the name of the property to present as input.

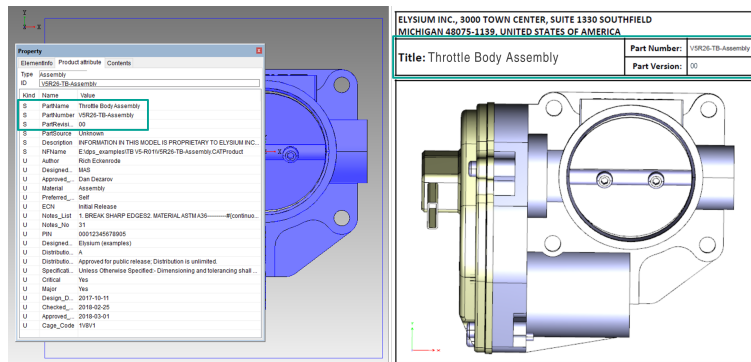


Figure 18. Example of model properties (left) displayed in Text fields (right).

A Text Field widget can also take a [text file](#) data source type as input which can be useful, e.g., to display an EAR/ITAR note. In such cases, you do not need to specify a property name (the content of the whole file will be presented).

The data source of a Text Field widget can also be provided by [Trigger/Action](#). This enables you to control what to display in a Text Field, and when, depending on properties and events available in a triggering widget (see [How-to Add a Trigger/Action Connection](#)).

Steps:

1. Open your TDP template in Adobe Acrobat and start DPS.
2. Add a ENF data source and name it **inputfile**.
3. Add a Text Field widget.
 - a. Click the [+ Add Widget] button.
 - b. Select **Text Field** as the type of the new widget.
 - c. Connect the new Text Field widget with its placeholder in the TDP template.
4. Specify the Text Field widget's content.
 - a. Select **inputfile (ENF)** as the data source.

- b. Specify the property to be displayed in the Text Field widget (e.g., **Part Number**).
 - c. Specify, optionally, that the Text Field widget should be editable by checking the "Text is Editable" checkbox (only available when the data source type is **ENF** or **Text**).
5. Customize the Text Field [widget's appearance](#).

6.1.6. How-to Add a PMI Table Widget

In this how-to guide you will learn how to add a PMI Table widget to a TDP. A PMI Table widget lists PMI items (e.g., datums, dimensions, and geometric tolerances) and specified properties in a table. It takes an **ENF** data source type as input.

Dimension / Geom. Tol.	Unit	Type
0.005 A H D	inch	gdt
0.005 A B	inch	gdt
0.005 A G D-E	inch	gdt
0.005 A G D-E	inch	gdt
0.005 F	inch	gdt
0.01 A	inch	gdt
0.01 A B D-E	inch	gdt
0.01 A B D-E	inch	gdt
0.01 A G D-E	inch	gdt
0.02 J	inch	gdt
1.000	inch	dimension / distance
1.000 -0.005/-0.005	inch	dimension / distance

Figure 19. Example of a PMI table.



[Predefined PMI Properties](#) describes the available PMI types and properties.

The video below will take you through the following steps.

Steps:

1. Open your TDP template in Adobe Acrobat and start DPS.
2. Add a ENF data source and name it **inputfile**.
3. Add a PMI Table widget.
 - a. Click the **[+ Add Widget]** button.
 - b. Select **PMI Table** as the type of the new widget.
 - c. Connect the new PMI Table widget with its placeholder in the TDP template.
4. Specify the PMI Table widget's content.
 - a. Select **inputfile (ENF)** as the data source.
 - b. Specify the PMI types you would like to include in the PMI Table widget.
 - c. Click the **[+ Add Column]** button to add a column to the PMI Table widget.
 - d. Specify the property to be presented in the column (e.g., **Text Rep.**).
 - e. Repeat Steps 4.c and 4.d for each additional property you want to add.

5. Add, optionally, [filters](#) to include/exclude content.
6. Customize the PMI Table [widget's appearance](#).

► <https://vimeo.com/373801203/71d5d891e9> (Vimeo video)

6.1.7. How-to Add a BoM Table Widget

In this how-to guide you will learn how to add a BoM Table widget. A BoM Table lists parts and specified properties in a table. It takes an [ENF](#) data source type as input.

Name	QTY
NX11-TB Assembly	1
NX11-TBM-MBD	1
NX11-TB-Pin-Gear	1

Figure 20. Example of a BoM table.

Steps:

1. Open your TDP template in Adobe Acrobat and start DPS.
2. Add a ENF data source and name it **inputfile**.
3. Add a BoM Table widget.
 - a. Click the [+ Add Widget] button.
 - b. Select **BoM Table** as the type of the new widget.
 - c. Connect the new BoM Table widget with its placeholder in the TDP template.

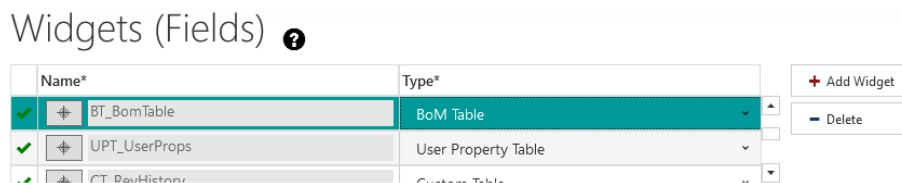


Figure 21. Adding a BoM Table widget.

4. Specify the BoM Table widget's content.
 - a. Select **inputfile (ENF)** as the data source.
 - b. Click the [+ Add Column] button to add a column to the BoM Table widget.
 - c. Specify the property to be presented in the column (e.g., **Part Quantity** or **Part Name**).
 - d. Repeat Steps 4.b and 4.c for each additional property you want to add.

Figure 22. Specifying the BoM Table widget's columns and what properties to link with them.



Flip the switch to **U** (instead of **P**) to add a user-defined property (instead of a predefined property).

5. Add, optionally, [filters](#) to include/exclude content.
6. Customize the BoM Table [widget's appearance](#).

6.1.8. How-to Add a Custom Table Widget

In this how-to guide you will learn how to add a Custom Table widget. A Custom Table widget takes a [CSV](#) file as an input and lists its content in a table.

	A	B	C	D
1	Revision	Description	Date	ECN
2	Rev_000	Initial release.	13/03/2018	-
3	Rev_001	The cap pin location change.	15/03/2018	ECN-001

Revision	Description	Date	ECN
Rev_000	Initial release.	13/03/2018	-
Rev_001	The cap pin location change.	15/03/2018	ECN-001

Figure 23. Example of revision history data in a CSV (top) displayed in a Custom table (bottom).



Refer to the ["example_revision_history.csv"](#) file for an example CSV input.



The first row of your CSV file will be used as a header by Data Package Studio.

Steps:

1. Open your TDP template in Adobe Acrobat and start DPS.
2. Add a CSV data source and name it **revision_history**.
3. Add a Custom Table widget.
 - a. Click the **[+ Add Widget]** button.
 - b. Select **Custom Table** as the type of the new widget.
 - c. Connect the new Custom Table widget with its placeholder in the TDP template.

Figure 24. Adding a Custom Table widget.

4. Specify the Custom Table widget's content.
 - a. Select **revision_history (CSV)** as the data source.
 - b. Click the **[+ Add Column]** button to add a column to the Custom Table widget.
 - c. Specify the CSV column name to be presented in the column (e.g., **Revision**).
 - d. Repeat Steps 4.b and 4.c for each additional property you want to add.

Header Name in CSV*	Header Name in TDP*	Info Popup	Width (%)
Revision	Revision	<input type="checkbox"/>	15
Description	Description	<input type="checkbox"/>	*
Date	Date	<input type="checkbox"/>	25
ECN	ECN	<input type="checkbox"/>	15

Figure 25. Specifying the Custom Table widget's columns and the corresponding CSV columns.

5. Add, optionally, **filters** to include/exclude content.
6. Customize the Custom Table **widget's appearance**.

6.1.9. How-to Add a User Property Table Widget

In this how-to guide you will learn how to add a User Property Table widget. A User Property Table widget lists user-defined properties related to a model's parts/assemblies as key/value pairs in a table.

Property	Value
Author	Lynn Garcia
Designed_By	Michael Reynolds
Approved_By	James Payne
Material	Assembly
Preferred_Vendor	Self
ECN	Initial Release
Notes_List	1. BREAK SHARP EDGES 2. MATERIAL ASTM ...
Notes_No	33
Cage_Code	1VBV1
PIN	00012345678905
Designed_For	Elysium (examples)
Distribution_Code	A

Figure 26. Example of user-defined properties in a model (left) displayed in a User Property table (right).

In contrast to other widgets, the User Property Table widget does not require a data source input. Instead it needs a **Trigger/Action** connection from a **BoM Table widget**, which will enable it to display all the user-defined properties of the part/assembly selected in the connected BoM Table widget.

Steps:

1. Open your TDP template in Adobe Acrobat and start DPS.
2. Add a ENF data source and name it **inputfile**.
3. Add a BoM Table widget and connect it with the **inputfile** data source (see [How-to Add a BoM Table Widget](#) for details).

4. Add a User Property Table widget.
 - a. Click the [+ Add Widget] button.
 - b. Select **User Property Table** as the type of the new widget.
 - c. Connect the new User Property Table widget with its placeholder in the TDP template.

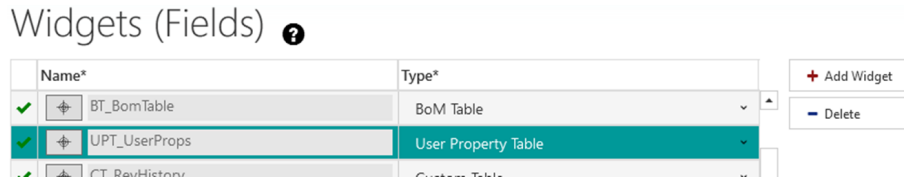


Figure 27. Adding a User Property Table widget.

5. Specify the User Property Table widget's content.
 - a. Specify the Header Name of Property Name (e.g., **Property**).
 - b. Specify the Header Name of Property Value (e.g., **Value**).

The screenshot shows the 'Content' tab of a configuration panel. It has two input fields: 'Header Name of Property Name (in TDP)*' with the value 'Property' and 'Header Name of Property Value (in TDP)*' with the value 'Value'.

Figure 28. Specifying the column header names of the User Property Table widget.

6. Add, optionally, **filters** to include/exclude content.
7. Add a Trigger/Action (see [How-to Add a Trigger/Action Connection](#) for details).
 - a. Select your BoM Table widget as the Trigger Widget.
 - b. Select your User Property Table widget as the Target Widget.

6.1.10. How-to Add a Table Column Filter Widget

In this how-to guide you will learn how to add a Table Column Filter widget, which enables filtering of the table it is connected to. Like the [User Property Table widget](#), it does not take a data source as input but requires a [Trigger/Action](#) connection to the table it is filtering.

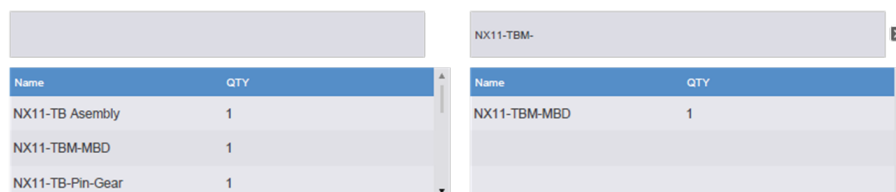


Figure 29. Example of a table column filter (left) when applied to the "Name" column of a table (right).

Steps:

1. Open your TDP template in Adobe Acrobat and start DPS.
2. Add a ENF data source and name it **inputfile**.
3. Add a Table Column Filter widget.

- a. Click the [+ **Add Widget**] button.
- b. Select **Table Column Filter** as the type of the new widget.
- c. Connect the new Table Column Filter widget with its placeholder in the TDP template.
- d. Customize the Table Column Filter [widget's appearance](#).

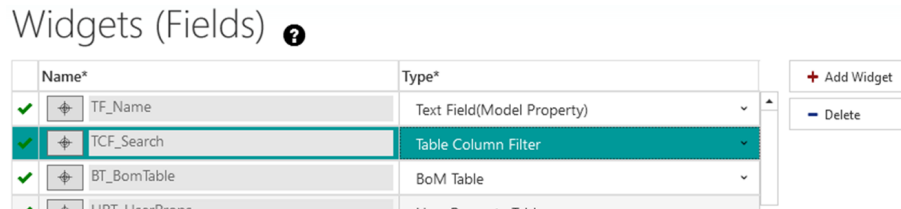


Figure 30. Adding a Table Column Filter widget.

4. Add a BoM Table widget (see [How-to Add a BoM Table Widget](#) for details).
5. Add a Trigger/Action (see [How-to Add a Trigger/Action Connection](#) for details).
 - a. Select your Table Column Filter widget as the Trigger Widget.
 - b. Select your BoM Table widget as the Target Widget.
 - c. Select the column(s) to filter on in your BoM Table widget.

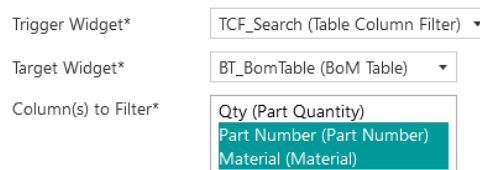


Figure 31. Adding a trigger/action connection between the Table Column Filter and BoM Table widgets.

6.1.11. How-to Customize a Widget

In this how-to guide you will learn how to customize the appearance of a widget by walking through an example using a BoM Table widget. Though tables have some unique appearance options, it should be easy to apply what you learn here on other widgets too.

We recommend that you complete the [How-to Add a BoM Table Widget](#) guide before starting here as it covers the first few steps in more details.

The video below will take you through the following steps.

1. Open your TDP template in Adobe Acrobat and start DPS.
2. Add a data source (e.g., a ENF named **inputfile**)
3. Add a BoM Table widget and customize its content:
 - a. Add a data source (e.g., **inputfile (ENF)**)
 - b. Add columns (e.g., **Part Quantity**, **Part Number**, and **Material**).
 - c. For each column:
 - i. Customize the width (e.g., **15%**, ***¹**, **30%**).

- ii. Specify if the column should enable showing cell content in an information popup.

BoM Table Columns

Property Name*	Header Name in TDP*	Info Popup	Width (%)
Part Quantity	Qty	<input type="checkbox"/>	15
Part Number	Part Number	<input type="checkbox"/>	*
Material	Material	<input checked="" type="checkbox"/>	30

Qty	Part Name	Material
1	VSR26-TB-MSD-RD1	Aluminum Casting
1	V5-R26-TB-PIN-GEAR	Steel ASTM A36
1	VSR26-TB-Pin-Flapper	Steel
1	VSR26-TB-Camper	Brass
1	VSR26-TB-PIN-FLAPPER-TOP	Steel
2	VSR26-TB-SCREW1	Steel
1	VSR26-TB-Caplug	Steel
1	VSR26-TB-Motor	Steel
1	VSR26-TB-SPRING-FLAPPER	Steel
1	VSR26-TB-Gear-Flapper	MC901 CAST NYLC

Info

MC901 CAST NYLC

OK

Figure 32. Enabling column cell content to be shown in an information popup.

4. In the Widgets / Appearance tab, customize the appearance of your table by specifying the:
- Number of table rows to display (e.g., 14) - scrolling is automatically enabled when table items are larger than "rows to display".
 - Scrollbar size (e.g., 15).
 - Customize the table header appearance:
 - Font size (e.g., 11).
 - Text color (e.g., White).
 - Text alignment (e.g., Center).
 - Text case (e.g., UPPERCASE).
 - Background color (e.g., #00A395).
 - Customize the table body appearance:
 - Font size (e.g., 8).
 - Text color (e.g., #00A395).
 - Text alignment (e.g., Center).
 - Text case (e.g., lowercase).
 - Border thickness (e.g., 1) - default is 0.
 - Border color (e.g., #00A395) - applies only if thickness is larger than 0.
 - Background color of odd rows (e.g., White) - applies to the first and third rows, and so on.
 - Background color of even rows (e.g., White) - applies to the second and fourth rows, and so on.
 - Highlight color (e.g., LightGoldenrodYellow) - applies to the selected row.

1: * means "distribute" and results in the available width (100% minus the explicitly specified width) being distributed equally across such columns.



Set different background colors for odd and even rows for a banded table.

► <https://vimeo.com/373805771/6d8e50285c> (Vimeo video)

6.1.12. How-to Add a Trigger/Action Connection

In this how-to guide you will learn how to add [triggers](#) and [actions](#) between [widgets](#).

The video below will take you through the following steps.

Steps:

1. Open your TDP template in Adobe Acrobat and start DPS.
2. Add a ENF data source and name it **inputfile**.
3. Add a BoM Table widget (see [How-to Add a BoM Table Widget](#)).
4. Add a 3D View widget (see [How-to Add and Customize a 3D View Widget](#)).
5. Add a trigger/action connection between the BoM Table widget and the 3D View widget.
 - a. In the Triggers and Actions tab, click [+ Add Trigger/Action].
 - b. Select the BoM Table widget as the trigger.
 - c. Select the 3D View widget as the target.



In Step 5, for some widget combinations, a [+ Add Return Trigger/Action] button is displayed. Click it to also add a trigger/action connection in the other direction.

► <https://vimeo.com/389388993/60be240065> (Vimeo video)

6.2. How-to Guides: Use Cases

6.2.1. How-to Create a TDP Template

This guide walks you through one way of creating your own TDP template. It will take you about one hour and fifteen minutes to complete if you design the template from scratch.

Example Files:

File	Description
"templates\tdp_example_template_design.pptx"	Template design file.

The Role of the Template in TDP Creation

Before starting with the template design, let us first have a look at the different roles that templates, Data Package Studio, and 3DxSUITE play in creating TDPs. By understanding the different roles you will be able to create better templates, faster.

- Template - PDF that:
 - Defines static contents (e.g., logotype and text) of a TDP.
 - Provides placeholders for dynamic contents (provided at the time of TDP generation).
- Data Package Studio - provides a [formula](#) that:
 - Connects dynamic contents (data sources) with their placeholders (via [widgets](#)).
 - Specifies how dynamic contents should be presented (widget type and appearance).
- 3DxSUITE - generates a TDP based on provided template, formula, and data source(s).

From a Data Package Studio point of view, the most important elements of a template are its widget placeholders. Without the widget placeholders Data Package Studio has no means of specifying what, where, and how content should be presented in a TDP.

So, how do you define a widget placeholder in a template? It is easy, just add a text type PDF Form field as in the figure below.

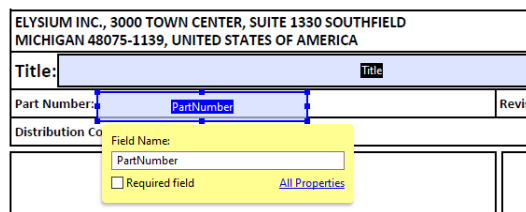


Figure 33. Creating widget placeholders by adding PDF Form fields (Text).

Creating Your Template

Your goal is to design the template (below left) that will be able to generate the TDP (below right).

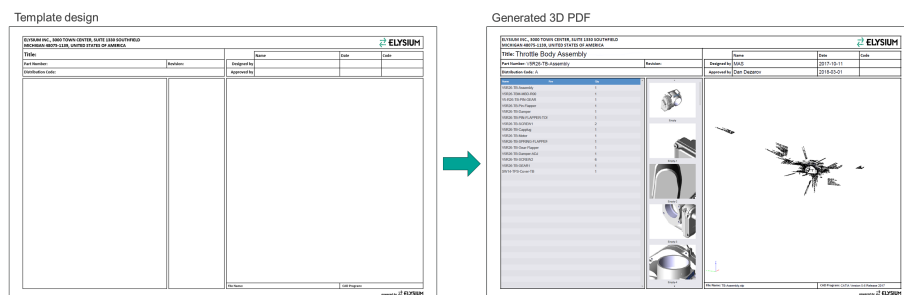


Figure 34. The template design.

The steps to create the template are described below and assumes that you will use PowerPoint and Adobe Acrobat Pro. You will use PowerPoint to design the complex grid layout of the template while you will use Adobe Acrobat to create the template and refine the placeholders.



You can replace PowerPoint with any application that can save a PDF.

Steps:

1. Design the template in PowerPoint.
 - a. Create a new PowerPoint file.
 - b. Insert a table (e.g., size = 1 x 1).
 - c. Adjust the table's width and position, background color(s), and border width and color to your liking.
 - d. Use Split Cell and Merge Cells functions to adjust the design like in the image below.
 - e. Add static text labels like in the image below.
 - f. Add static logotype like in the image below.

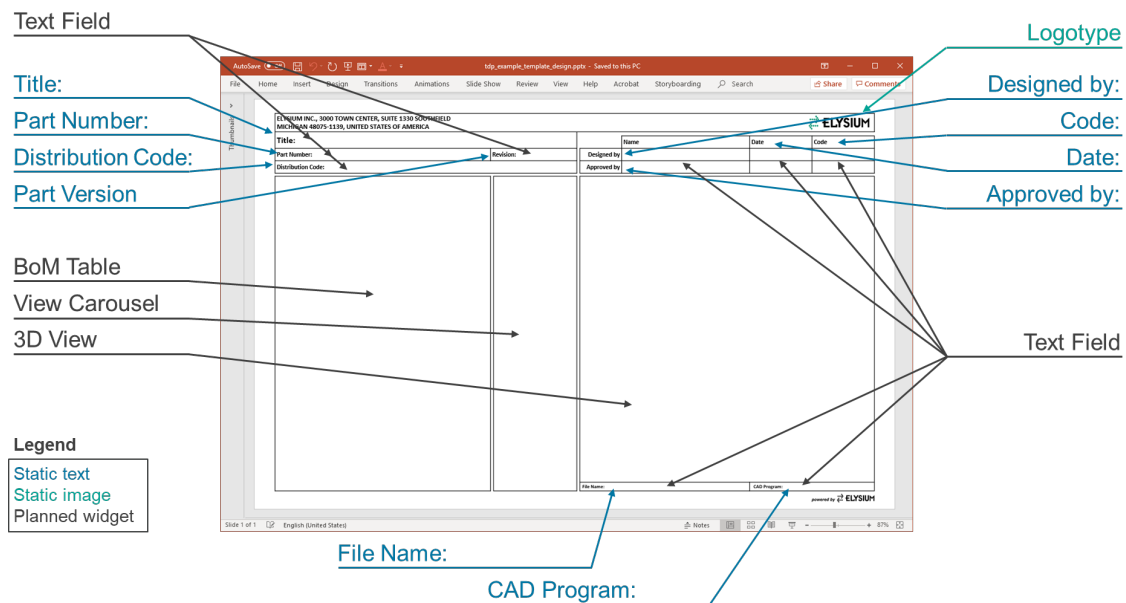


Figure 35. Template design in PowerPoint including planned location of widgets.



You can skip the above steps by using the provided "templates\tdp_example_template_design.pptx" file - this will save you about an hour.

2. Generate a TDP template from the PowerPoint file.
 - a. Click [**Create and Share Adobe PDF**] in PowerPoint's Home Ribbon.
 - b. Save the generated PDF.

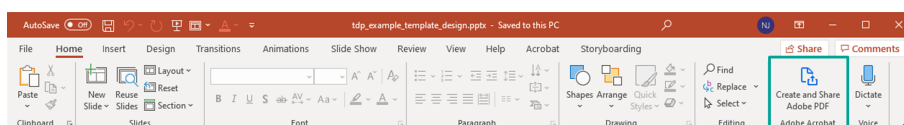


Figure 36. Generating a TDP template from a PowerPoint design.

3. Generate PDF fields in Adobe Acrobat Pro.
 - a. Open the generated PDF in Adobe Acrobat Pro and click **Tools > Prepare Form**.

- b. Select your template file (not Scan a Document) and click **[Start]** to generate PDF fields.

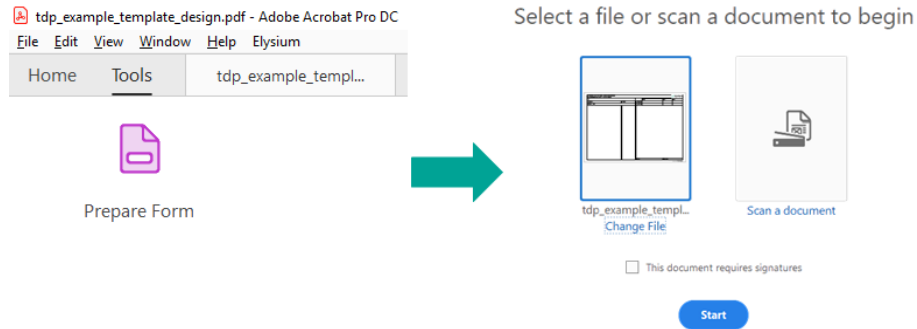


Figure 37. Generating PDF fields in Adobe Acrobat Pro.

4. Optionally (if you are not happy with, e.g., the names or sizes of the generated PDF fields), adjust the template in Adobe Acrobat Pro.
 - a. Adjust sizes and positions of the generated PDF fields to your liking.
 - b. Rename PDF fields to your liking.
 - c. Add PDF fields that were not generated (if you used the provided PowerPoint design template then the Distribution Code PDF field is most likely missing).
 - d. Remove PDF fields that are not needed (if you used the provided PowerPoint design template then no fields should need to be removed).
5. Save the adjusted template.

You now have a template that is ready to be used in Data Package Studio.

6.2.2. How-to Customize a CAD Validator Report

This how-to guide will walk you through the steps of customizing a CAD Validator report. You will use 3DxSUITE SmartLauncher (SLS) to generate your customized TDP.



Read the [Migrating to the New CAD Validator Report Template Format](#) appendix to get an introduction to widgets and the template basics of CAD Validator reports.

Example Files:

File	Description
"data_sources\example_throttle_body_r01.enf"	First model input (changed).
"data_sources\example_throttle_body_r00.enf"	Second model input (original).
"templates\cv_example_template.pdf"	CAD Validator report template.

File	Description
"formulas\cv_formula.rb"	Formula/scenario to generate your customized CAD Validator report.
"scenarios\cv_scenario.esa"	

The next few sections will walk you through the following steps:

1. [Create Your CAD Validator Report Template](#)
2. [Create Your CAD Validator Report Formula](#)
3. [Create Your CAD Validator Report Scenario](#)
4. [Generate Your CAD Validator Report with SLS.](#)

Create Your CAD Validator Report Template

To create your own CAD Validator report template, you need to create a PDF with placeholders (PDF Form fields) for each widget you want to include in the report. See [How-to Create a TDP Template](#) for details on how to create a template and [CAD Validator Widget Types](#) for a list of available CAD Validator widgets.



Avoid naming your customized data placeholders (PDF fields) to anything starting with "ELY_" (see [Reserved PDF Field Names](#)).

Once you are done, you should have a template that looks something like the example in the image below.

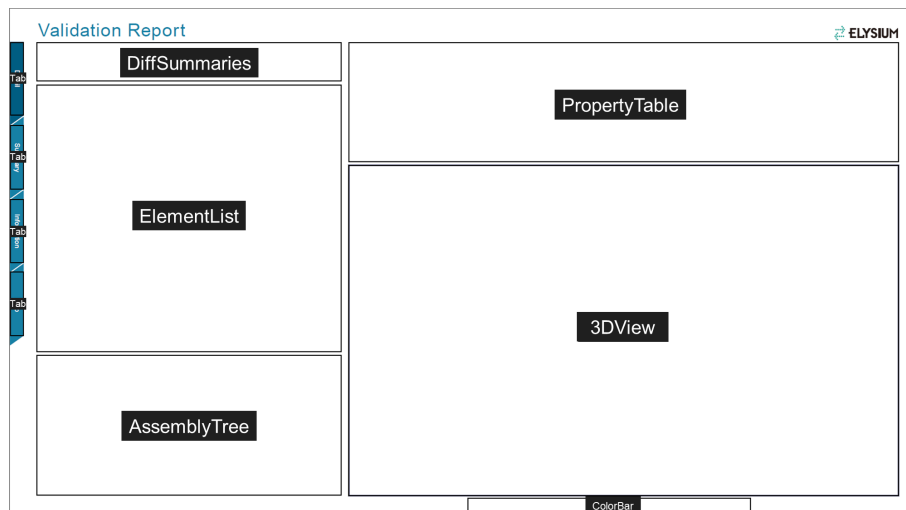


Figure 38. Example CAD Validator report template.

Create Your CAD Validator Report Formula

In this section we will walk you through the steps of creating a formula for your customized CAD Validator report template. We will not cover all the CAD Validator widgets, just the most essential ones, such as the [Difference Summary](#), [Detail List](#), and [3D View](#) widgets.

Steps:

1. Open your template in Adobe Acrobat Pro.
2. Start Data Package Studio (DPS) (click **Elysium** > **Start Data Package Studio**).
3. In the **[Data Sources]** tab/page, add a CAD Validator Result type of data source and name it **cv-result**.



When adding a CAD Validator Result data source, all the CAD Validator widgets are automatically added to the **[Widgets]** tab/page, but you still need to connect them to their respective placeholder.

4. In the **[Widgets]** tab/page:
 - a. Select the 3D View (CAD Validator) widget.
 - b. Uncheck the Show PMI (at startup) to hide all PMI items when the report is opened.
 - c. Uncheck the toolbar items you do not want to include in the report (e.g., Render Mode (Selector), Action Menu / Show Unselected Parts (Checkbox), and Refresh (Button)).

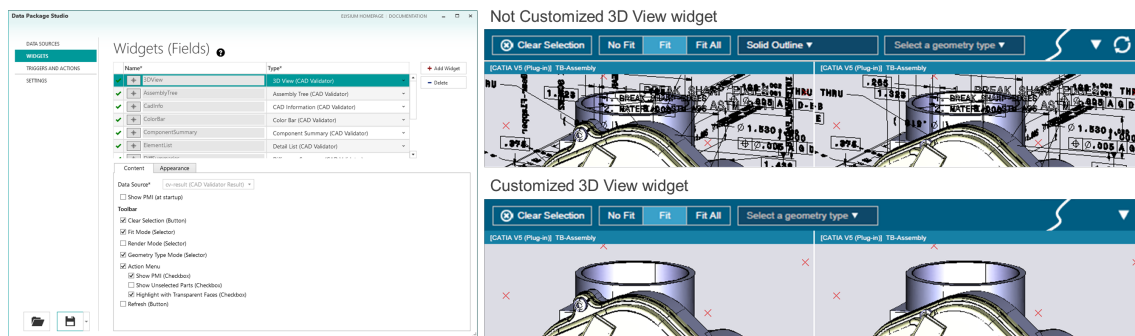


Figure 39. Example of a 3D View widget customization: DPS options (left), report without customization (right top), and a report with customization (right bottom).

5. In the **[Widgets]** tab/page:
 - a. Select the Detail List (CAD Validator) widget.
 - b. Uncheck the items you do not want to include in the report (e.g., PMI (Checkbox), Group List / Action Menu, and Element List / Action Menu / Copy to Group).

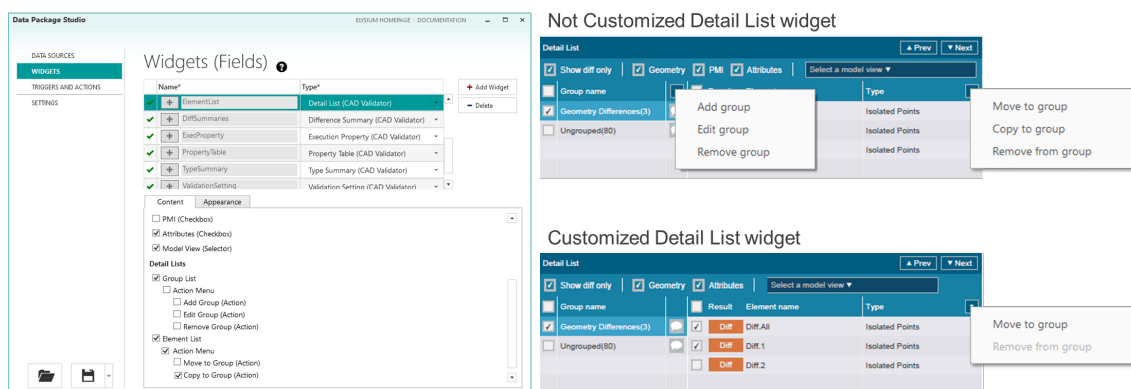


Figure 40. Example of a Detail List widget customization: DPS options (left), report without customization (right top), and a report with customization (right bottom).

6. In the [**Widgets**] tab/page:
 - a. Select the Difference Summary (CAD Validator) widget.
 - b. Uncheck the items you do not want to include in the report (e.g., Passed/Failed (Assessment), PMI (Differences)).

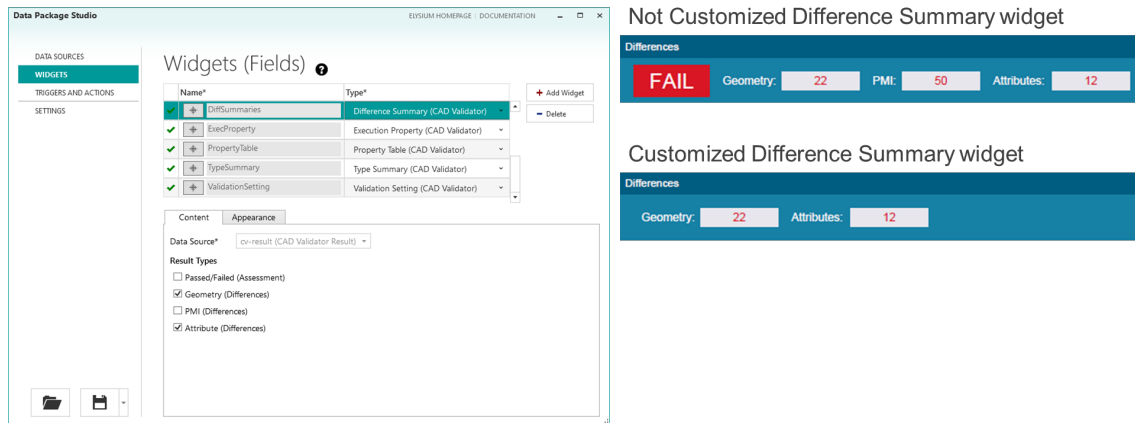


Figure 41. Example of a Difference Summary widget customization: DPS options (left), report without customization (right top), and a report with customization (right bottom).

Create Your CAD Validator Report Scenario

Before you can generate a customized report, you need to set the following parameters (see [Required Parameters for CAD Validator Component](#) for details):

- `Create3DPdfReport=1`
- `3DPdfReportType=Custom`
- `Customized3DPdfReportScript=<PATH_TO_FORMULA>`

Here we will create a scenario using the 3DxSUITE ScenarioEditor. You will then save the scenario to the 3DxSUITE SmartLauncher (SLS) scenario folder before generating your customized CAD Validator report.

Steps:

1. Open the 3DxSUITE ScenarioEditor.
2. Add a CAD Validator component.
 - a. Expand the Optimizer item in the left-hand menu.
 - b. Select the CAD Validator component item (note that it is added between the first Scenario Input and Scenario Output boxes¹).
3. Add a second Scenario Input and connect it to the CAD Validator component.
 - a. Expand the Scenario I/O item in the left-hand menu.
 - b. Select the Scenario Input item (note that a second Scenario Input is added and that it has no connections).
 - c. Connect the second Scenario Input to the CAD Validator component.

- i. Click the output port (grey dot at the bottom) of the second Scenario Input box (and hold the mouse button).
 - ii. "Drag" the output port to the second input port (grey dot to the top right) of the CAD Validator component box (note the arrow that appears when you start dragging).
 - iii. "Drop" the arrow on the second input port of the CAD Validator component box (note that the two items are now connected).
4. Set the CAD Validator parameters required to generate a customized report.
 - a. Select the CAD Validator box
 - b. Set the 3D PDF Report | Script File Type for Report Template parameter (**3DPdfReportType**) to **Custom**.
 - c. Set the Export Validation Result | 3D PDF Report parameter (**Create3DPdfReport**) to **1**.
 - d. Set the 3D PDF Report | Script File for Customized Template parameter (**Customized3DPdfReportScript**).
 - i. Select the External Reference (or Embed File) option.
 - ii. Select the formula script (e.g., **C:\dps\examples\formulas\cv_formula.rb**).
5. Save the scenario file.
 - a. Select 3DxSUITE scenario folder (e.g., **C:\Users\Public Documents\Elysium\3DxSUITE\Scenarios**).
 - b. Name the scenario (e.g., **cv_scenario**).
 - c. Save the scenario.

1 Box(es) refers to the boxes in the visual scenario flow editor at the center of the 3DxSUITE ScenarioEditor.

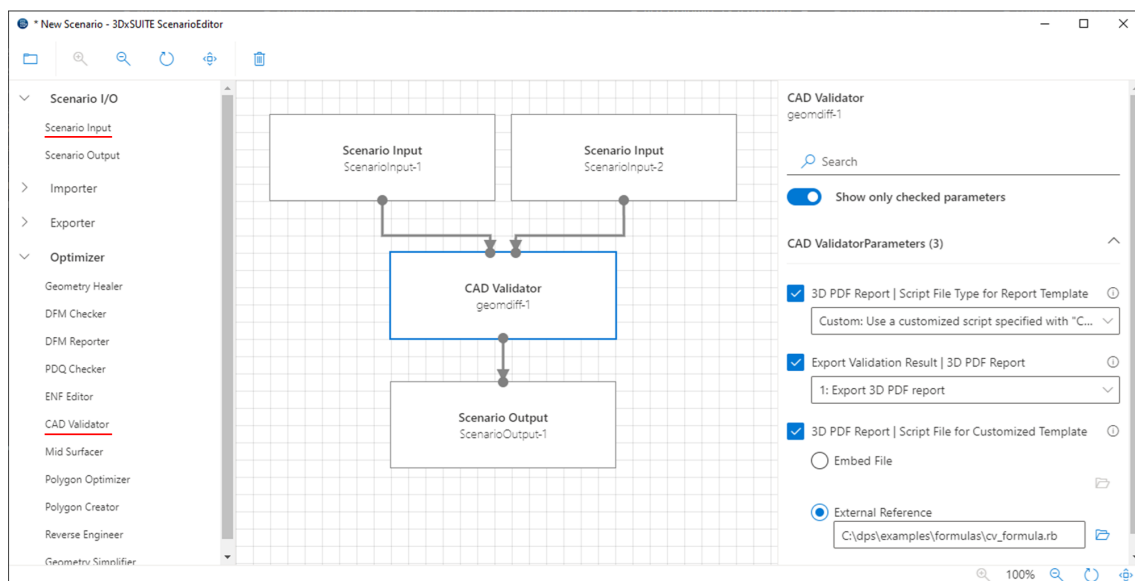


Figure 42. Example scenario created in 3DxSUITE ScenarioEditor.



From EX9.0, the CAD Validator (CV) component in the scenario above and a single template/formula (including CV and non-CV widgets) are all you need to

fully customize a CAD Validator report.



The two ENF files used as CV input can be accessed in Data Package Studio by adding data sources named **inputfile1** (CV's source) and **inputfile2** (CV's target) - see [How-to Name an ENF Data Source](#).



To set a 3D PDF property in the CV component, add it as a Custom Parameter using the **\$ENF23DPDF_** prefix, e.g., to change the 3D View(s) background color to red, add **\$ENF23DPDF_BackgroundColor=(255,0,0)**.

You are now ready to generate your customized CAD Validator report.

Generate Your CAD Validator Report with SLS

The final step is to generate your customized CAD Validator report. This will be simple compared to all the work you have already done. What remains is for you to select the two models to compare, select a location where to save the generated report, and generate the report.

Steps:

1. Open SLS with your first model (right click on, e.g., **example_throttle_body_r00.enf**, and select **Elysium > Run Scenario...**).
2. Select your scenario (e.g., **CAD Validator Customization Scenario (cv_scenario.esa)**).
3. Set the number of input files for the scenario to **2**.
4. Select your second input file (e.g., **example_throttle_body_r01.enf**).
5. Specify your output folder (e.g., **C:\dps\output**).
6. Click **[OK]** to generate your customized CAD Validator report.

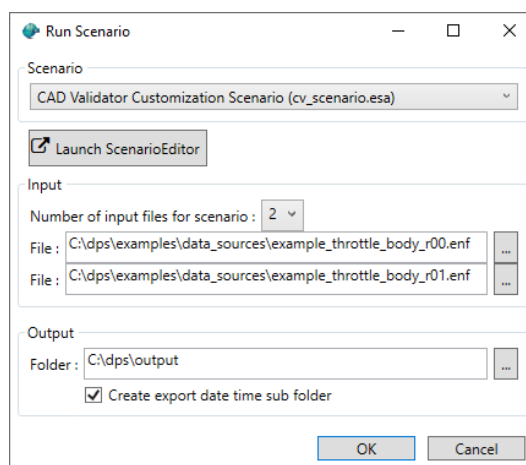


Figure 43. Example of running a scenario in 3DxSUITE SmartLauncher.

6.2.3. How-to Include/Exclude Content in a Table or View Carousel

In this how-to guide you will learn how to include and exclude content in Table and View

Carousel widgets. We will use the BoM Table widget in our example, but the steps are the same for all Tables and the View Carousel.

Before we go through the steps of how to add a table content filter, let us first have a look at how the content filtering works. A content filter has three parts:

1. Property Name - specifies the property that the filter will match against.
2. Filter Type - specifies if a match should **Include** or **Exclude** the item.
3. Filter - specifies a simple text or Regular Expression to match against the specified property.

An **Include** type of filter will include all items that matches the filter whereas an **Exclude** type of filter will include all items that do not match the filter. Only the items that remain after all filters have been applied will be available as the content of a Table or View Carousel widget.

If you add multiple filters, then the filter engine will:

1. First, remove the items that match one or more **Exclude** filters.
2. Then, from the remaining items, keep only those that match one or more **Include** filters.

Internally, the matching uses a Regular Expression (something like: `property =~ /filter/`). You do not need to know Regular Expressions to specify a filter, a simple text is enough. However, simple text filters may match more inclusively and, therefore, require adding more filters that are more explicit to accomplish the same result.

Example:

In the table below, you can see the original BoM content to the left and the wanted BoM table content to the right. Since, this example will only consider filtering on a BoM item's Part Number property, the table only includes part numbers.

Table 8. The original model BoM content (left) and wanted BoM table content (right).

Original	Wanted
TB-Assembly	TB-Assembly
TB-Pin-Flapper	-
TB-Pin-Flapper-Top	TB-Pin-Flapper-Top
TB-Pin-Gear	TB-Pin-Gear
TBM-Body	-

First, consider applying a simple text filter **Flapper**. This filter would match all the items that contain **Flapper**, i.e., all **Flapper** items would be included or excluded depending on the filter type. However, since this filter matches both an item we want to include and one we want to exclude, the filter cannot be used to achieve the wanted content. You could achieve the wanted

content by adding three simple text **Include** filters:

- **Assembly**
- **Pin-Flapper-top**
- **Gear**

This, however, is very explicit and would not be robust against model changes.

If you instead used a Regular Expression, you would be able to express that you want to exclude all items that end with **Flapper**. You could be even more granular by specifying that it should start with **TB-**, have zero or more characters in the middle, and end with **Flapper**.

The Regular Expression **Exclude** filter described above is formally specified as **^TB-.*-Flapper\$**. The **^** matches the beginning of a text. The **.** matches any character except line breaks. The ***** is a quantifier that matches zero or more of the preceding token (an "any character" match in our case). The **\$** matches the end of a text.



You can find many Regular Expression resources on the Internet, including online tools to help you tune your own regular expressions.

In the steps below, you will add the filter in the example above and an additional filter that includes items that contain **TB-**.

Steps:

1. In Data Package Studio, add a BoM Table with Part Name, Number, and Quantity columns (see [How-to Add a BoM Table Widget](#) for details).
2. In the Table Content Filter section, click the **[+ Add Filter]**.
 - a. Specify the property you want to filter on, e.g., **Part Number**.
 - b. Specify the filter type, e.g., **Exclude** to exclude matching items.
 - c. Specify the filter, e.g., **^TB-.*-Flapper\$** to match items that start with **TB-** and ends with **-Flapper**.
3. Add another filter that includes items with a **Part Number** that contain **TB-**.

BoM Table Columns

Property Name*	Header Name in TDP*	Info Popup	Width (%)
<input checked="" type="radio"/> P Part Name	Part Name	<input type="checkbox"/>	*
<input checked="" type="radio"/> P Part Number	Part Number	<input checked="" type="checkbox"/>	35
<input checked="" type="radio"/> P Part Quantity	QTY	<input type="checkbox"/>	15

Table Content Filter

Property Name*	Filter Type*	Filter*
<input checked="" type="radio"/> P Part Number	Exclude	^TB-.*-Flapper\$
<input checked="" type="radio"/> P Part Number	Include	TB-

Figure 44. Example of the filters created in the steps above.

If you apply the filters you have just created to the same model content as in the initial example, then you would get the BoM content listed to the right in the table below. The TB-Pin-Flapper item is excluded because it matches the **Exclude** filter. The TBM-Body item is excluded because it does not match the **Include** filter.

Table 9. Example of how the filters created in the steps above are applied.

Original	Exclude →	Output	Include →	Final Output
TB-Assembly	^TB-.*-Flapper\$	TB-Assembly	^TB-	TB-Assembly
TB-Pin-Flapper		-		-
TB-Pin-Flapper-Top		TB-Pin-Flapper-Top		TB-Pin-Flapper-Top
TB-Pin-Gear		TB-Pin-Gear		TB-Pin-Gear
TBM-Body		TBM-Body		-

6.2.4. How-to Add a Bill of Characteristics Table

In this guide you will learn how to pre-process a 3D model to prepare Bill of Characteristics (BoC) data that can be displayed in the [PMI Table widget](#).

An example use case is to streamline the communication between design and inspection planning by automatically extracting BoC data from your 3D models and including it in TDPs going to your inspection planning team.

You will learn:

- The general anatomy of a pre-processing script.
- How to create your own BoC pre-processing script.
- How to configure 3DxSUITE to run your pre-processing script when generating a TDP.

Example Files:

File	Description
"scripts\pre_process_template.rb"	Template pre-processing script.
"scripts\pre_process_boc.rb"	Example BoC pre-processing script.
"scenarios\dps_scenario.esa"	Scenario file with the BoC Formula parameter set.
"data_sources\TBM-MBD-R00_BoC.enf"	3D model with BoC "balloons".

Anatomy of a Pre-Processing Script

The anatomy of a pre-processing script is straightforward. The script is based on Ruby and has three elements that are required for 3DxSUITE to identify and run your pre-processing script (see template script below).

Listing 1. Template TDP pre-process script.

```
# -*- encoding: UTF-8 -*-
module PdfEditor                                ①
  class CustomizedEnfPreProcess                 ②
    logger.info "Custom File (#{__FILE__}): "

    def on_run_before(ee_session)               ③
    end

  end
end
```

- ① The `module PdfEditor` defines where the script "lives" in the 3DxSUITE "Universe".
- ② The `class CustomizedEnfPreProcess` defines a class that implements the pre-processing script, and resides inside the PdfEditor module.
- ③ The `def on_run_before(ee_session)` defines a method that takes an ENF Editor Session as input and resides inside the CustomizedEnfPreProcess class. It is also where your pre-processing logic needs to be implemented.

If you are not familiar with programming or Ruby then this may seem incomprehensible. Do not worry, you only need to remember two things:

1. Your pre-processing script must follow the structure and naming described above.
2. Your pre-processing logic needs to be implemented in the `on_run_before(ee_session)`.

If you follow these instructions, then 3DxSUITE will be able to identify and run your pre-processing script.

You can do almost any type of pre-processing this way. You can, for example, add properties to model elements provided by an external JSON file or, as you will learn in the next section, prepare BoC data for the PMI Table widget.

How to Create Your Own BoC Pre-Processing Script

You are now ready to create your own pre-processing script to prepare BoC data to be displayed in the PMI Table widget.

Your script will process a model in three steps:

1. Identify annotations (PMI items) that are associated with BoC "balloons".

2. Identify annotations that represent the BoC "balloons".
3. Connect the BoC "balloons" with respective associated PMI item.

All three steps require looping through the annotations in a model, so let us first have a look at how you do that by using the ENF Editor API.



You can use the ENF Editor API to process model data because the original model is translated to [ENF](#) prior to the pre-processing stage.

Listing 2. How to loop through all annotations in the model.

```
# ...

def on_run_before(ee_session)
  ee_session.model.components.each do |compo|           ①
    compo.annotations.each do |anno|                   ②
      # ...
    end
  end
end

# ...
```

① Use the ENF Editor Session object (ee_session) to loop through each component in the model.

② Loop through each annotation of a component.

Next you will identify the annotations (PMI items) that are associated with a BoC "balloon". In the example model the name of such PMI items will include a text like `1_EREF(2;5)` where `EREF` is the key identifying word. The Integer numbers within parenthesis `2;5` are references to two associated "balloons" with identifiers equal to `2` and `5`, respectively.



In your CAD model, do not separate multiple numbers by a comma (,) or the highlighting of the BoC balloons and associated PMI will not work.

Let us have a look at how you can use this knowledge to identify PMI items associated with BoC "balloons".

Listing 3. How to identify PMI items associated with BoC "balloons".

```
# ...

ee_session.model.components.each do |compo|
  # Create mapping from BoC ID to referenced PMI
  id_to_ref = Hash.new
  compo.annotations.each do |anno|
    md = anno.name.match(/EREF\(((\d;)+)\)/)           ①
    next if md.nil? || md.size != 2                   ②
    md[1].split(";").each do |id|                     ③

```



```

        id_to_ref[id] = anno
      end
    end
    # ...
  end
# ...

```

- ① Match a regular expression to capture a BoC-associated PMI item by its **EREF** name pattern.
- ② Skip to the next annotation if the regular expression did not match.
- ③ Map each reference identifier with the PMI item (stored in a Ruby hash for connecting with its respective "balloon" later).

So, now you have a Ruby hash (**id_to_ref**) that maps the BoC identifier with the ENF object that represents the PMI item. Next you will flag BoC "balloons" and use the hash to connect them with their respective PMI item, but first you need to identify the BoC "balloons".

The name of a BoC "balloon" item in the example model will include a text like **9_EBOC2** where **EBOC** is the key word identifying the annotation as a BoC "balloon" item. The last Integer number **2** is the unique identifier that is used by the PMI item (see above) to associate it with its "balloon".

Equipped with this knowledge, let us have a look at how you can use it.

Listing 4. How to identify BoC "balloons".

```

# ...
# Create mapping from BoC ID to referenced PMI
# ...

# Flag BoC and set referenced PMI
compo.annotations.each do |anno|
  md = anno.name.match(/EBOC(\d+)/) ①
  next if md.nil? || md.size != 2    ②
  # ...
end
# ...

```

- ① Match a regular expression to capture a BoC "balloon" by its **EBOC** name pattern.
- ② Skip to the next annotation if the regular expression did not match.

You now have:

- A Rubyhash that maps a BoC identifier with the ENF object of the associated PMI item.
- Identified a BoC "balloon" item and its BoC identifier (the **anno** object and the **md** array).

This is enough to flag the identified annotation as a BoC "balloon" and to connect it with its associated PMI item.

Listing 5. How to flag BoC "balloons" and connect them with their associated BoC items.

```

# ...
# Create mapping from BoC ID to referenced PMI
# ...

# Flag BoC and set referenced PMI
compo.annotations.each do |anno|
  md = anno.name.match(/EBOC(\d+)/)
  next if md.nil? || md.size != 2
  ref_anno = id_to_ref[md[1]]      ①
  next if ref_anno.nil?           ②
  anno.boc_tag = true              ③
  anno.boc_ref = ref_anno          ④
end
# ...

```

- ① Get the PMI item associated with the BoC "balloon".
- ② Skip to the next BoC "balloon" if it does not have an associated PMI item.
- ③ Tag the annotation as a BoC "balloon".
- ④ Connect the BoC "balloon" with its associated PMI item.

You can find the complete example pre-processing script in [Example TDP Pre-Process Script](#).

How to Configure 3DxSUITE to Run Your BoC Pre-Processing Script

To put your BoC pre-processing script to use, or any other pre-processing script for that matter, there is only one thing remaining. You need to configure 3DxSUITE to run your script when generating your TDP.

This is simple enough. You only need to specify where your pre-processing script is located by using the '[CustomizedEnfPreProcessScript](#)' parameter. If you are using 3DxSUITE SmartLauncher you set this parameter via its user interface as in the figure below.

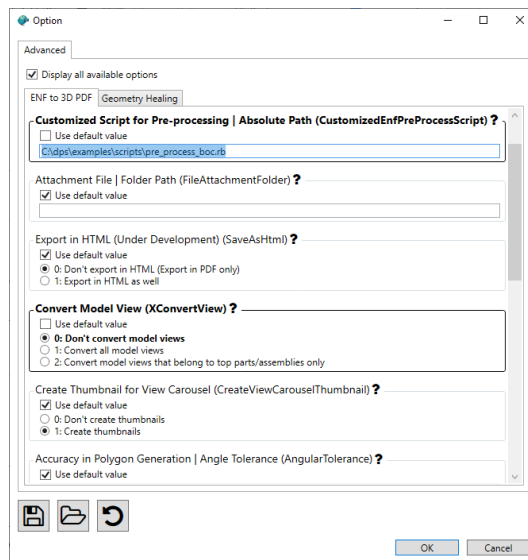


Figure 45. Setting the 'CustomizedEnfPreProcessScript' parameter in 3DxSUITE SmartLauncher.



If you are using a scenario, then you will set the 'CustomizedEnfPreProcessScript' parameter when you create/edit your scenario in 3DxSUITE ScenarioEditor.

You should now be able to run your BoC pre-processing script when generating a TDP. Try it by using the `dps_scenario.esa` scenario, the `BoC Formula` parameter set, and the `TBM-MBD-R00_BoC.enf` example model. Update the 'CustomizedEnfPreProcessScript' scenario parameter to point to your own BoC pre-processing script, or keep it as is to use the provided example script.

6.2.5. How-to Save Table Data using Elysium's TDP JavaScript API

This how-to guide will take you through the steps of creating a button to save a table's data using Elysium's TDP JavaScript API (TDP JS API). See the [Class: ElyTdpTableWidget](#) documentation for details on the Table JS API.

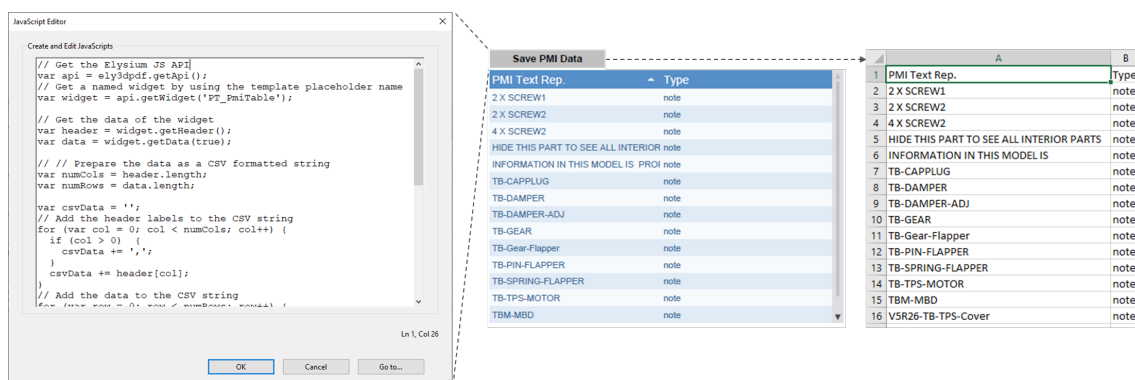


Figure 46. Example of saving table data with the TDP JS API.

We recommend that you complete the [How-to Add a 3D Toolbar using Elysium's TDP JavaScript API](#) guide before starting here as it covers the first few steps in more details.

Example Files:

File	Description
"formulas\basic_how_to_guide_template.pdf/json/rb"	Formula.
"scenarios\dps_scenario.esa"	Scenario with How-To Guide Formula parameter set.
"data_sources\example_throttle_body_r01.enf"	Model input.

Steps:

1. Add a button and name it, e.g., btn_save_pmi.
2. Double click the button to open Button Properties.
3. In the **[Options]** tab:
 - a. Select the "Label only" layout.
 - b. Input "Save PMI Data" as the label.
4. In the **[Actions]** tab:
 - a. Select "Mouse Up" as the trigger.
 - b. Select "Run a JavaScript" as the action.
 - c. Click the **[Add...]** button.
5. In the JavaScript Editor that opens:
 - a. Add your Save PMI Data code - see example code below.
 - b. Save your code and close the editor (just click **[OK]** if you use the built in editor).
6. In the Button Properties window, click **[Close]**.

Listing 6. Example code to save table data as a CSV file.

```

var api = ely3dpdf.getApi();
var widget = api.getWidget('PT_PmiTable');

var header = widget.getHeader(); ①
var data = widget.getData(true); ②

var numCols = header.length;
var numRows = data.length;

var csvDelimiter = ','; ③
var csvData = '';
for (var col = 0; col < numCols; col++) { ④
  if (col > 0) {
    csvData += csvDelimiter;
  }
  csvData += data[col];
}

```

```

    }
    csvData += header[col];
  }
  for (var row = 0; row < numRows; row++) { ⑤
    csvData += '\n';
    for (var col = 0; col < numCols; col++) {
      if (col > 0) {
        csvData += csvDelimiter;
      }
      var cellData = data[row][col];
      if (cellData != '') {
        cellData = '"' + cellData + '"'; ⑥
      }
      csvData += cellData;
    }
  }
  try {
    var dataObjectName = 'PmiCsvFile.csv';
    createDataObject(dataObjectName, csvData); ⑦
    exportDataObject(dataObjectName); ⑧
    removeDataObject(dataObjectName); ⑨
  } catch (err) {
    app.alert({
      cTitle: 'Save CSV File Error',
      cMsg: 'Saving requires Adobe Acrobat Pro.'
    });
  }
}

```

- ① Get the header labels of the table (Array of strings).
- ② Get the filtered table data (Array of Arrays of strings) - provide no input or false to get unfiltered data.
- ③ Decide on what CSV delimiter to use - here we use a comma (",").
- ④ Add header labels to the csvData string.
- ⑤ Add table data to the csvData string.
- ⑥ Wrap the cell data in double apostrophes if your table data includes your chosen CSV delimiter or it will break your CSV structure.
- ⑦ Create a CSV file and attach it to the TDP.
- ⑧ Open a save-file prompt to save the CSV file attachment.
- ⑨ Remove the CSV file attachment from the TDP.



Steps 7-9 use the Adobe JS API to save the CSV file - it requires Adobe Acrobat Pro¹ and may be subject to change.

1: Adobe Acrobat Reader will cause a JS error when calling the `createDataObject` function in Step 7.

6.2.6. How-to Create a Work Instructions TDP

In this how-to guide you will learn how to create a Work Instructions TDP. First, we will describe what we mean with [Work Instructions](#). Then, we will explain how we [conceptually support Work Instructions](#) creation. Finally, we will walk you through the [steps of creating an example Work Instructions formula](#).

Example Files:

File	Description
"scenarios\dps_scenario.esa"	Scenario file with the Work Instructions Formula parameter set.
"data_sources\work_instructions\TB-Assembly-WI.enf"	Model input.
"data_sources\work_instructions\wi_steps.csv"	Work Instructions steps input.
"data_sources\work_instructions\wi_parts.csv"	Input of parts associated with a step.
"data_sources\work_instructions\wi_tools.csv"	Input of tools associated with a step.
"data_sources\work_instructions\wi_safety.csv"	Input of safety instructions associated with a step.

Definition

Work Instructions are a guide, or standard operating procedure, that describes, in detail, how to perform an activity. It helps teams accurately manufacture, assemble, package, and ship products to the required specifications.

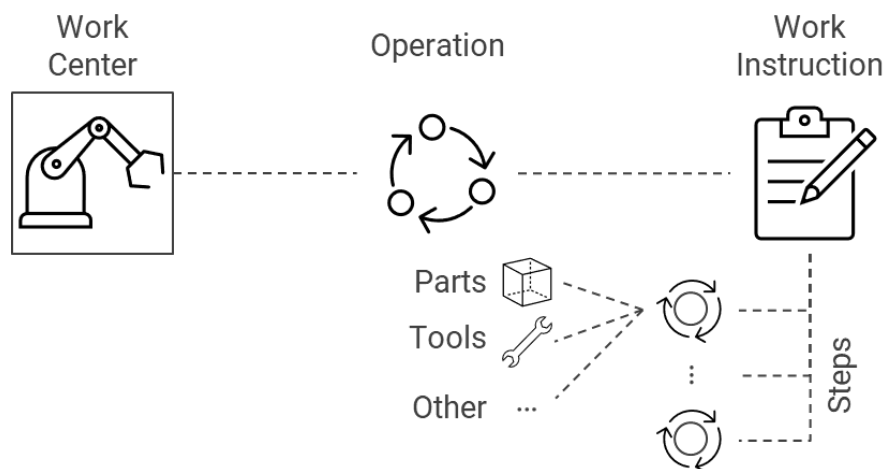


Figure 47. Example of how Work Instructions relate to other manufacturing entities.

Concept

We have enabled Work Instructions by using general data sources, widgets and trigger/actions instead of specific ones. This gives you the means to adapt Work Instructions TDPs to your unique needs instead of the opposite.

In the [Example](#) section, you will create a basic Work Instructions TDP as in [Figure 48](#). When a Work Instructions step is selected in the Step list (top-right), then it is visualized in the 3D View (top-left), its description is displayed below the Step list, its related Parts, Tools, and Safety Instructions are displayed in the respective lists below the Step description.

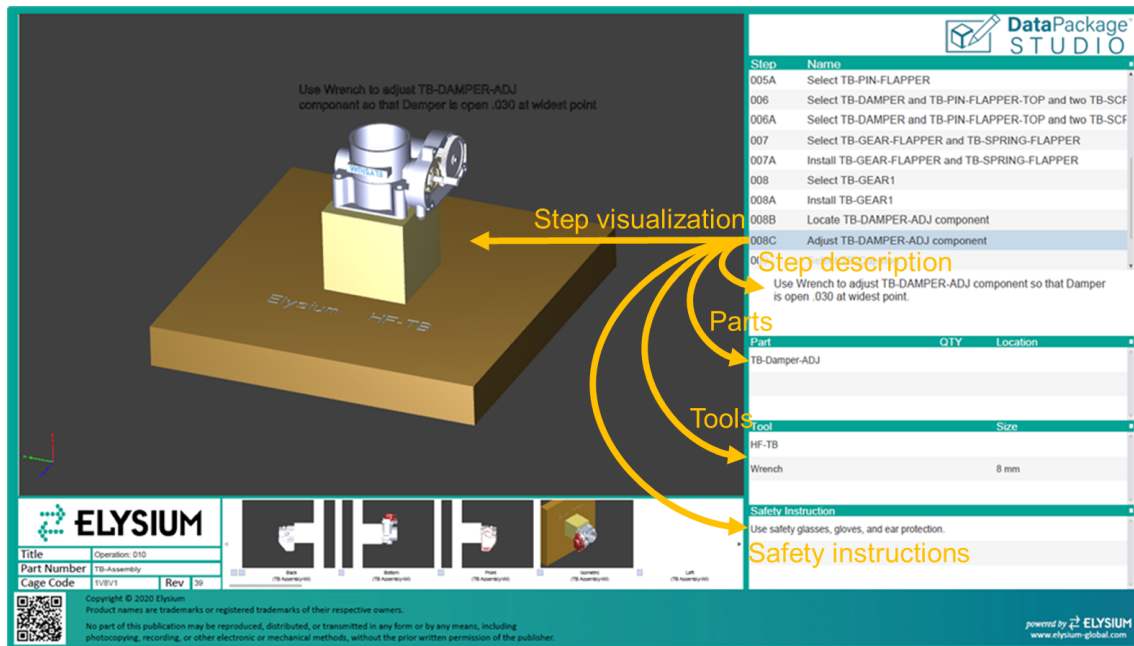


Figure 48. Example Work Instructions TDP.

The data for the Work Instructions TDP in [Figure 48](#) is provided by CSV data sources as depicted in [Figure 49](#). You can include any data required to express and describe your Work Instructions as long as you also include the data required to connect the widgets.

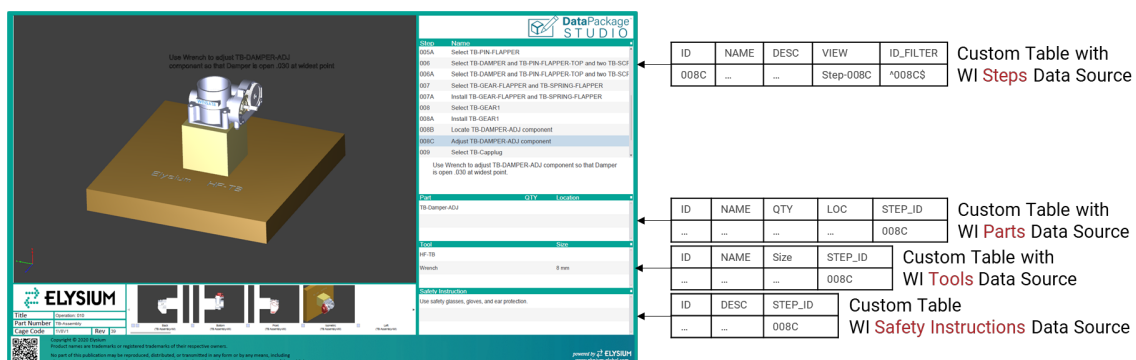


Figure 49. Work Instructions data provided by CSV data sources.

Let us first look at the data required to set a view in the 3D View widget when selecting an item in the Step list. The full view name is required, which is the view name plus the owning part name within parentheses, e.g., **Step-008C (TB Assembly-WI)**. See the **STEP_VIEW** column in the Work Instructions Steps data source (**wi_steps.csv**) for example.

The Trigger/Action that you will create between the Step list (Custom Table widget) and the 3D View widget in [Example](#) will take the **STEP_VIEW** value of the selected item and set the view.

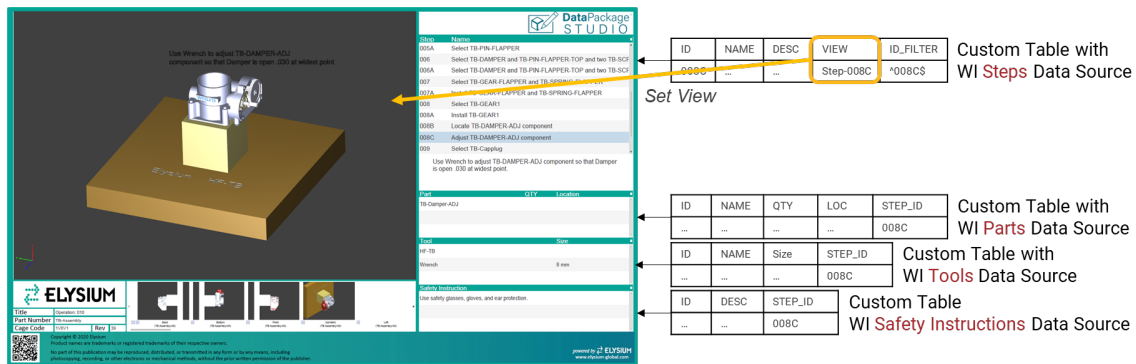


Figure 50. The Step CSV data source contains the view name required to set a view.

Next is the description of a step, which we want to display in a Text Field widget when selecting an item in the Step list. In the Work Instructions Steps data source, the **STEP_DESC** column contains the description of each step.

The Trigger/Action that you will create between the Step list (Custom Table widget) and the Text Field widget in [Example](#) will take the **STEP_DESC** value of the selected item and display it.

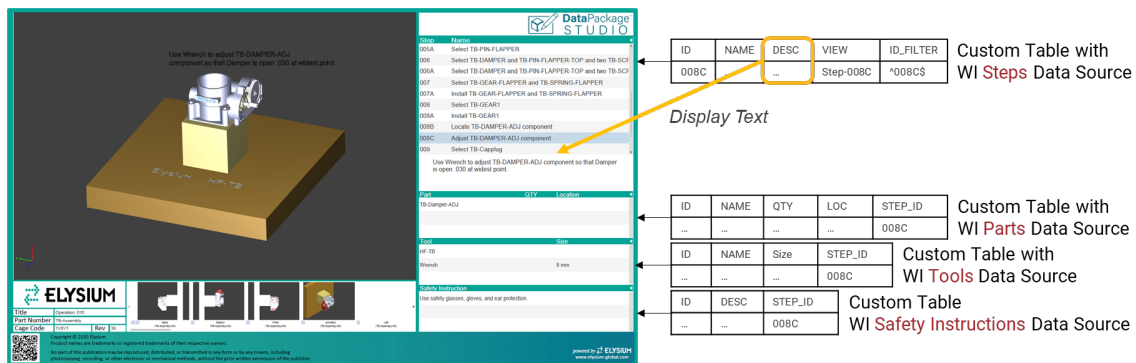


Figure 51. The Step CSV data source contains the description to display.

Finally, we want to display Parts, Tools, and Safety Instructions associated with a step when it is selected in the Step list. We can achieve this by filtering the Parts, Tools, and Safety Instructions tables when a Step item is selected.

Let each item in the Work Instructions Steps data source specify a filter based on its ID, e.g., **^008C\$**. Each item in the Parts, Tools, and Safety Instructions data sources specifies the Step ID it is associated with, e.g., **008C**.

We are using Regular Expression filters to match the ID exactly, i.e., start of text (^) + ID + end of text (008C\$). See the **STEP_ID_FILTER** column in the Work Instructions Steps data source (**wi_steps.csv**) and the **STEP_ID** columns in the other CSV data sources for examples.

The Triggers/Actions that you will create between the Step list (Custom Table widget) and the Parts, Tools, and Safety Instructions lists (Custom Table widgets) in [Example](#) will take the **STEP_ID_FILTER** value and filter the connected tables on their respective **STEP_ID** column.

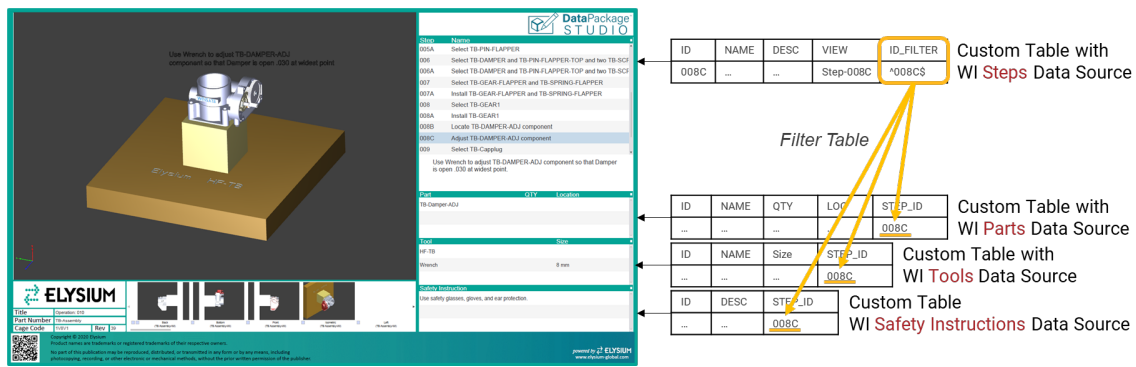


Figure 52. The Step CSV data source contains the filter that will match the associated items in the Parts, Tools, and Safety Instructions CSV data sources.

Now that you understand our Work Instructions concept, you can prepare your own CSV data sources. As mentioned earlier, you can include any data you need to address your specific needs as long as you include the required data.

You may need to extract the data from your MES and/or other systems as depicted in Figure 53. Note, however, that the Work Instructions data extraction steps (greyed) are out of scope of the 3DxSUITE products.

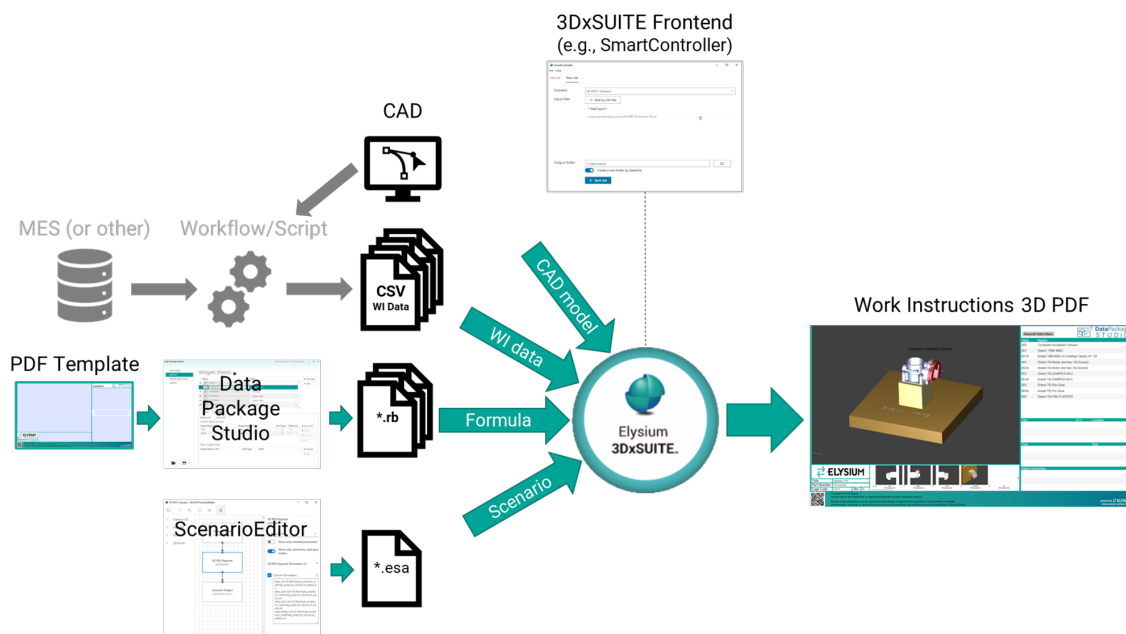


Figure 53. Example of a Work Instructions TDP creation workflow.

Example

In this section you will create the basic Work Instructions TDP depicted in Figure 48.

Steps:

1. Open the `wi_formula.pdf` in Data Package Studio.
2. Create the required data sources (see Figure 54):
 - a. ENF data source named `inputfile`.

- b. CSV data source named **step_list** (UTF-8 encoded).
- c. CSV data source named **step_part_list** (UTF-8 encoded).
- d. CSV data source named **step_tool_list** (UTF-8 encoded).
- e. CSV data source named **step_safety_list** (UTF-8 encoded).

Data Sources






ID*	Type*	Encoding*	Attachment	Description
 inputfile	ENF	-	<input type="checkbox"/>	
 step_list	CSV	Unicode (UTF-8)	<input type="checkbox"/>	
 step_part_list	CSV	Unicode (UTF-8)	<input type="checkbox"/>	
 step_tool_list	CSV	Unicode (UTF-8)	<input type="checkbox"/>	
 step_safety_list	CSV	Unicode (UTF-8)	<input type="checkbox"/>	

Figure 54. Data sources for the Work Instructions TDP.

3. Add widgets:

- a. Add a Custom Table widget for the **CT_StepList** placeholder.
 - i. Specify **step_list** (CSV) as the data source.
 - ii. Add a column with CSV Header: **STEP_ID**, TDP Header: **Step**, and width: **15**.
 - iii. Add a column with CSV Header: **STEP_NAME**, TDP Header: **Name**, and width: *****.

Data Source* step_list (CSV)

☐ Show Empty Table (at startup and unfiltered)

Custom Table Columns

Header Name in CSV*	Header Name in TDP*	Info Popup	Width (%)
STEP_ID	Step	<input type="checkbox"/>	15
STEP_NAME	Name	<input type="checkbox"/>	*

Figure 55. Data source and columns for the Step list (Custom Table widget).

- b. Add a Text Field widget for the **TF_StepDesc** placeholder.
 - i. Specify **By Trigger/Action** as the data source (to get data from a selected Step item).
- c. Add a Custom Table widget for the **CT_PartsList** placeholder.
 - i. Specify **step_part_list** (CSV) as the data source.
 - ii. Check the "Show Empty Table" checkbox (to hide items until a Step item is selected).
 - iii. Add a column with CSV Header: **PART_NAME**, TDP Header: **Part**, and width: **50**.
 - iv. Add a column with CSV Header: **PART_QTY**, TDP Header: **QTY**, and width: *****.
 - v. Add a column with CSV Header: **PART_LOC**, TDP Header: **Location**, and width: **35**.

Data Source* step_part_list (CSV)

☒ Show Empty Table (at startup and unfiltered)

Custom Table Columns

Header Name in CSV*	Header Name in TDP*	Info Popup	Width (%)
PART_NAME	Part	<input checked="" type="checkbox"/>	50
PART_QTY	QTY	<input type="checkbox"/>	*
PART_LOC	Location	<input checked="" type="checkbox"/>	35

Figure 56. Data source and columns for the Parts list (Custom Table widget).

- d. Add a Custom Table widget for the **CT_ToolsList** placeholder.
 - i. Specify **step_tool_list (CSV)** as the data source.
 - ii. Check the "Show Empty Table" checkbox (to hide items until a Step item is selected).
 - iii. Add a column with CSV Header: **TOOL_NAME**, TDP Header: **Tool**, and width: *****.
 - iv. Add a column with CSV Header: **TOOL=SIZE**, TDP Header: **Size**, and width: **35**.
- e. Add a Custom Table for the **CT_SafetyList** placeholder.
 - i. Specify **step_safety_list (CSV)** as the data source.
 - ii. Check the "Show Empty Table" checkbox (to hide items until a Step item is selected).
 - iii. Add a column with CSV Header: **TOOL_NAME**, TDP Header: **Tool**, and width: *****.
 - iv. Add a column with CSV Header: **TOOL=SIZE**, TDP Header: **Size**, and width: **35**.
- f. Add a View Carousel widget for the **VC_Views** placeholder.
 - i. Specify **inputfile (ENF)** the data source.
 - ii. Add a Regular Expression **Exclude** filter (**^Step-\d{1,3}[A-Z]?\$**) to exclude all Step views (see [How-to Include/Exclude Content in a Table or View Carousel](#)).

Data Source*

Table Content Filter

Property Name*	Filter Type*	Filter*
<input checked="" type="radio"/> Name	Exclude	^Step-\d{1,3}[A-Z]?\$

Figure 57. Data source and content filter to exclude Step views for the View Carousel.

- g. Add a 3D View widget for the **3DView** placeholder and the **inputfile (ENF)** data source.
- h. Add a Text Field widget for the **TF_Title** placeholder.
 - i. Specify **inputfile (ENF)** as the data source.
 - ii. Specify **Enter property name...** as the property.
 - iii. Specify **Operation** as the name of the property.
- i. Add a Text Field widget for the **TF_PartNumber** placeholder.
 - i. Specify **inputfile (ENF)** as the data source.
 - ii. Specify **Enter property name...** as the property.
 - iii. Specify **PartNumber** as the name of the property.
- j. Add a Text Field widget for the **TF_CageCode** placeholder.
 - i. Specify **inputfile (ENF)** as the data source.
 - ii. Specify **Enter property name...** as the property.
 - iii. Specify **CageCode** as the name of the property.
- k. Add a Text Field widget for the **TF_Revision** placeholder.
 - i. Specify **inputfile (ENF)** as the data source.

ii. Specify **Part Version** as the property.

4. Add Trigger/Actions:

a. Add a Trigger/Action to visualize a step in the 3D View.

i. Select **CT_StepList (Custom Table)** as the Trigger Widget.

ii. Select **3DView (3D View)** as the Target Widget.

iii. Specify **STEP_VIEW** as the Trigger Property.

Trigger Widget*	CT_StepList (Custom Table) ▼
Target Widget*	3DView (3D View) ▼
Trigger Property*	STEP_VIEW

Figure 58. Connecting the Step list with the 3D View.

b. Add a Trigger/Action to display a step's description.

i. Select **CT_StepList (Custom Table)** as the Trigger Widget.

ii. Select **TF_StepDesc (Text Field)** as the Target Widget.

iii. Specify **STEP_DESC** as the Trigger Property.

Trigger Widget*	CT_StepList (Custom Table) ▼
Target Widget*	TF_StepDesc (Text Field) ▼
Trigger Property*	STEP_DESC

Figure 59. Connecting the Step list with the description Text Field.

c. Add a Trigger/Action to display the Parts associated with a step.

i. Select **CT_StepList (Custom Table)** as the Trigger Widget.

ii. Select **CT_PartsList (Custom Table)** as the Target Widget.

iii. Specify **STEP_ID_FILTER** as the Trigger Property.

iv. Specify **STEP_ID** as the Target Property.

Trigger Widget*	CT_StepList (Custom Table) ▼
Target Widget*	CT_PartsList (Custom Table) ▼
Trigger Property*	STEP_ID_FILTER
Target Property*	STEP_ID

Figure 60. Connecting the Step list with the Parts list.

d. Add a Trigger/Action to display the Tools associated with a step.

i. Select **CT_StepList (Custom Table)** as the Trigger Widget.

ii. Select **CT_ToolsList (Custom Table)** as the Target Widget.

iii. Specify **STEP_ID_FILTER** as the Trigger Property.

iv. Specify **STEP_ID** as the Target Property.

Trigger Widget*	CT_StepList (Custom Table) ▼
Target Widget*	CT_ToolsList (Custom Table) ▼
Trigger Property*	STEP_ID_FILTER
Target Property*	STEP_ID

Figure 61. Connecting the Step list with the Tools list.

- e. Add a Trigger/Action to display the Safety Instructions associated with a step.
 - i. Select **CT_StepList (Custom Table)** as the Trigger Widget.
 - ii. Select **CT_SafetyList (Custom Table)** as the Target Widget.
 - iii. Specify **STEP_ID_FILTER** as the Trigger Property.
 - iv. Specify **STEP_ID** as the Target Property.

Trigger Widget*	CT_StepList (Custom Table) ▼
Target Widget*	CT_SafetyList (Custom Table) ▼
Trigger Property*	STEP_ID_FILTER
Target Property*	STEP_ID

Figure 62. Connecting the Step list with the Safety Instructions list.

- f. Add a Trigger/Action from the View Carousel to the 3D View to enable changing views.
5. Save your formula.
6. Create (or configure) your scenario (see the 3DxSUITE ScenarioEditor manual for details).
 - a. Add your CSV data sources as 3D PDF Exporter Custom Parameters (see [Figure 63](#)).
 - **step_list** - specifies the location of the step_list data source.
 - **step_part_list** - specifies the location of the step_part_list data source.
 - **step_tool_list** - specifies the location of the step_tool_list data source.
 - **step_safety_list** - specifies the location of the step_safety_list data source.

☒ Custom Parameters ⓘ

```

step_list=C:\dps\examples\data_sources\wi\wi_steps.csv
step_part_list=C:\dps\examples\data_sources\wi\wi_parts.csv
step_tool_list=C:\dps\examples\data_sources\wi\wi_tools.csv
step_safety_list=C:\dps\examples\data_sources\wi\wi_safety.csv
    
```

Figure 63. Custom Parameters to specify the location of the CSV data sources.

7. Run your scenario using 3DxSUITE SmartController (see [How-to Generate a TDP with 3DxSUITE SmartController](#) for details).
 - a. Select your Scenario, e.g., **DPS How-To Guide Scenario**.
 - b. Select your Parameter Set, e.g., **Work Instructions Formula**.
 - c. Select your Input File, e.g., **C:\dps\examples\data_sources\work_instructions\TB-Assembly-WI.enf**.
 - d. Start the job.

6.2.7. How-to Enable Printing of Saved Views

In this guide you will learn how to enable printing of saved views.

We recommend that you complete the [Generate Your First TDP](#) guide before starting here.

Example Files:

File	Description
"templates\basic_how_to_guide_template.pdf"	TDP template.
"data_sources\example_throttle_body_r01.enf"	Model input.
"scenarios\dps_scenario.esa"	Scenario with How-To Guide Formula parameter set.

Steps:

1. Open your TDP template in Adobe Acrobat and start DPS.
2. Add a 3D View widget, a View Carousel widget, and a Button widget (e.g., with label **Print Selected Views**).
3. Customize the [widgets' appearances](#).
4. Add a "print saved views" Trigger/Action.
 - a. Click [**+ Add Trigger/Action**].
 - b. Select a button as the trigger widget (e.g., **BTN_PrintViews (Button)**).
 - c. Select **Document** as the target widget.
 - d. Select **Print Saved Views** as the action.
 - e. Select a mode for rendering the views for printing (e.g., **Illustration**).
 - f. Select a 3D View widget to use for displaying the views for printing (e.g., **3D View widget (3D View)**).
 - g. Select a View Carousel widget to use for selecting views for printing (e.g., **VC_Views (View Carousel)**).

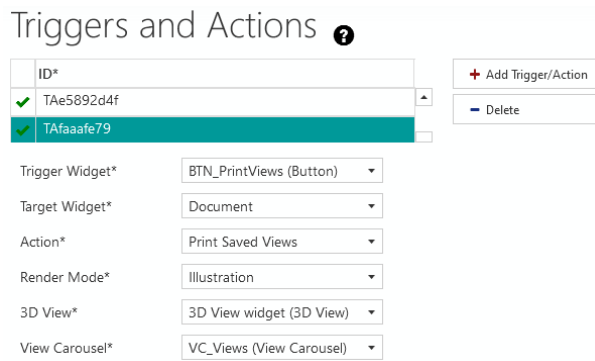


Figure 64. Adding a Trigger/Action between a button and a document's print saved views action.

7. Save your formula (e.g., "C:\dps\my_formula.rb").
8. Generate your customized TDP (see [Generate Your First TDP](#)).

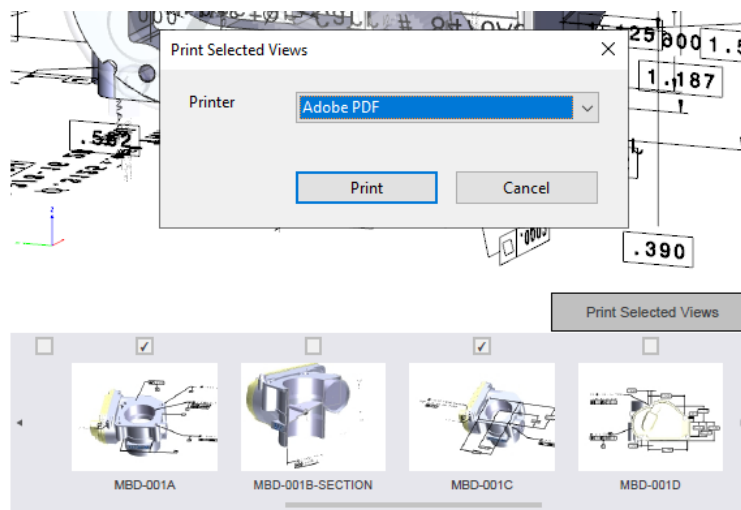


Figure 65. Example of enabling printing of saved views.

6.2.8. How-to Add a 3D Toolbar using Elysium's TDP JavaScript API

In this how-to guide you will go through the steps of creating a 3D toolbar using the [Elysium TDP JavaScript API](#) (TDP JS API). The TDP JS API allows you to add customized behavior to your TDP by providing access to a selected set of functions of the widgets you can customize via Data Package Studio (DPS).

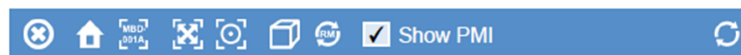


Figure 66. Example of a 3D toolbar created with the TDP JS API.

You will learn:

- The general approach to adding customized functionality via the API.
- How to get the API, and how to use it to get and interact with a named widget.
- How to utilize what you learn from the above to create a simple 3D toolbar.

Example Files:

File	Description
"formulas\basic_how_to_guide_template.pdf/json/rb"	Formula.
"data_sources\example_throttle_body_r01.enf"	Model input.
"scenarios\dps_scenario.esa"	Scenario with How-To Guide Formula parameter set.

General Approach to Adding Customized Functionality Using the TDP JS API

You add customized functionality using the TDP JS API by adding JavaScript code to your TDP.

Before you start to code, consider a few things:

1. When to add your code: before or after generating your TDP?
2. How to add your code: embed it, add it to a field's actions, or a combination of the two?
3. What to name your code: how to avoid naming conflicts?

What approach to take for each of the aspects listed above depends on what you want to achieve. We will go through each aspect and provide simple guidelines below.

When to add your code:

- Add your code to your template (before generating your TDP), if you want the same behavior in all generated TDPs.
- Add your code after you have generated your TDP, if you want the behavior to be specific to the generated TDP.

How to add your code:

- Embed your code, if you want to:
 - Trigger functions from events (e.g., before a document closes).
 - Reuse functions to keep your code DRY¹.
- Add your code to a field's actions, if you want user mouse events to trigger functions.
- Combine the above, if you want to reuse functions across fields.

1: DRY = Don't Repeat Yourself - a coding best practice to, e.g., reduce maintenance.

What to name your code

As you can see in the [Elysium TDP JavaScript API](#) documentation, we have used **Ely** to prefix our TDP JS API classes, e.g., **ElyTdpApi**. However, not all classes, global functions, and global variables/constants embedded in a TDP will have this prefix. So, we suggest that you use a

naming convention, e.g., adding a prefix that is unique to your organization, to avoid naming conflicts.

How-to Use the TDP JS API

Once you have decided your approach, it is time to start coding. One of the first things you want to do is to get hold of the TDP JS API. You can get the API by calling the `getApi` function that is available via the global object `ely3dpdf`.

```
var api = ely3dpdf.getApi();
```

You then, likely, want to get a widget so you can invoke its functions, or, in other words, make it do things. This requires you to know the name of the widget, which you can find in Name column in the DPS Widgets / Widgets (Fields) table.

Widgets (Fields) ?

Name*	Type*
✓ 3D View widget	3D View
✓ VC_VIEWS	View Carousel
✓ PT_PmiTable	PMI Table
✓ TCF_ViewProperty	Text Field
✓ BTN_PrintViews	Button

Figure 67. You can find a widget's name in Name column in the DPS Widgets / Widgets (Fields) table.

You then use the widget name when you call the `getWidget` function to get hold of the widget.

```
var api = ely3dpdf.getApi();
var widget = api.getWidget('3D View widget');
```

With these two steps you now have a widget that you can interact with. Let us try a couple of simple things like checking the type of a widget and, if it is a 3D View widget, reset it.

```
// ...
var type = widget.getType();
if (type === 'ElyTdp3dViewWidget') {
  widget.reset();
}
```

Next you will utilize what you have learned so far to implement a simple 3D toolbar.

How-to Create a 3D Toolbar Using the TDP JS API

In this section you will go through the basic steps that we have used to create a few of the 3D toolbar items you can find in the "[basic_how_to_guide_formula.pdf](#)". The steps cover creating the toolbar items and adding JS code (functionality) to the items, but not the formatting of them.

The toolbar items you will create:

1. Fit All (parts) button.
2. Show/Hide PMI checkbox.
3. Reset (3D View) button.



Figure 68. Example of a 3D toolbar items you will create with the TDP JS API.

First, open one of your templates that include a 3D View widget in Adobe Acrobat Pro, and start the Prepare Form tool. You can see that several new items have appeared in the Adobe Acrobat Pro toolbar, such as Text Field, Checkbox, and Button.



Make a copy of "[basic_how_to_guide_formula.pdf](#)" and delete the existing 3D toolbar by using the Prepare Form tool.

Create a Fit All button

To create a Fit All button, take the following steps:

1. Add a button and name it, e.g., FIT_ALL_BTN.

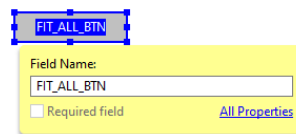


Figure 69. Adding a Fit All button.

2. Double click the button to open Button Properties.
3. In the **[Options]** tab:
 - a. Select the "Label only" layout.
 - b. Input "Fit All" as the label.

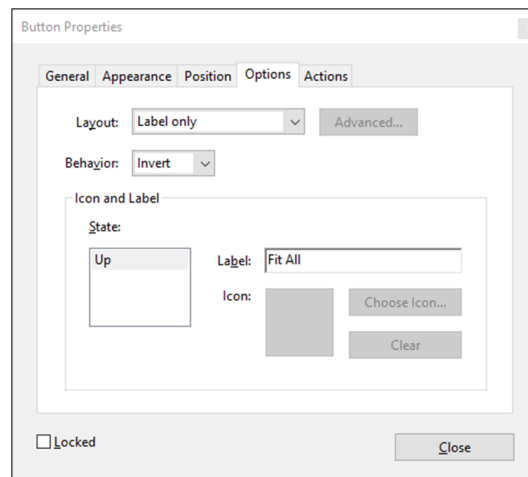


Figure 70. Setting the label for the Fit All button.

4. In the **[Actions]** tab:
 - a. Select "Mouse Up" as the trigger.
 - b. Select "Run a JavaScript" as the action.
 - c. Click the **[Add...]** button.

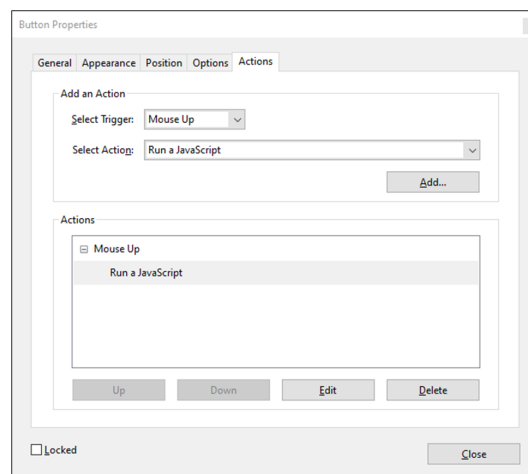


Figure 71. Adding a JavaScript action for the Fit All button.

5. In the JavaScript Editor that opens:
 - a. Add your Fit All code - see code below.
 - b. Save your code and close the editor (just click **[OK]** if you use the built in editor).
6. In the Button Properties window, click **[Close]**.

Listing 7. Code to fit all parts in a 3D View.

```
var api = ely3dpdf.getApi();
var widget = api.getWidget('3D View widget');
if (widget != null) {
    widget.fitAll();
}
```

Create checkbox to show/hide PMIs

To create a checkbox to Show/Hide PMIs, you go through the same steps as when you created the Fit All button. Name the new checkbox differently, e.g., SHOW_PMI_CB, and add the code included below to its JavaScript action. If you want a label for the checkbox, then add a "Text" item next to it and type, e.g., "Show PMI".

Listing 8. Code to show/hide PMIs based on the value of a checkbox.

```
var api = ely3dpdf.getApi();
var widget = api.getWidget('3D View widget');
var checkbox = api.getField('SHOW_PMI_CB');

if (widget != null && checkbox != null) {
    var showPmi = checkbox.isBoxChecked(0);
    widget.setShowPmi(showPmi);
}
```



If you go with "Show PMI", then make sure the checkbox is checked by default because PMIs are, currently, always shown when opening a TDP.

Create a Reset button

To create a button that resets a 3D View widget, you go through the same steps as when you created the Fit All button. Name the new button differently, e.g., RESET_ALL_BTN, set its label to, e.g., "Reset 3D", and add the following code to its JavaScript action.

Listing 9. Code to reset a 3D View widget.

```
var api = ely3dpdf.getApi();
var widget = api.getWidget('3D View widget');
if (widget != null) {
    widget.reset();
}
```

6.3. How-to Guides: TDP Generation

6.3.1. How-to Generate a TDP with 3DxSUITE SmartLauncher

In this guide you will learn how to generate a TDP with 3DxSUITE SmartLauncher (SLS).

Example Files:

File	Description
"scenarios\dps_scenario.esa"	Scenario with the Basic Product Definition Formula parameter set.
"data_sources\example_throttle_body_r01.enf"	Model input.

The video below will take you through the following steps.

Steps:

1. Customize your TDP template with Data Package Studio (see [Generate Your First TDP](#)).
2. Generate your TDP with SLS.
 - a. Open the model input file (right click and select **Elysium > Convert/Change File Type...**).
 - b. Select your scenario ("**DPS How-To Guide Scenario**").
 - c. Select your parameter set ("**Basic Product Definition Formula**")
 - d. Specify your output folder.
 - e. Start your job.

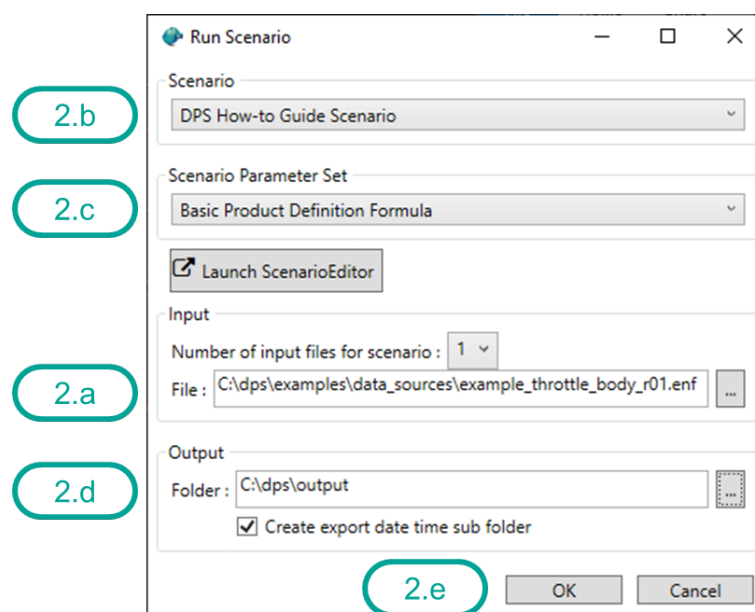


Figure 72. Example of generating a TDP with 3DxSUITE SmartLauncher.

6.3.2. How-to Generate a TDP with 3DxSUITE SmartController

In this guide you will learn how to generate a TDP with 3DxSUITE SmartController (SC).

Example Files:

File	Description
"scenarios\dps_scenario.esa"	Scenario with the Basic Product Definition Formula parameter set.
"data_sources\example_throttle_body_r01.enf"	Model input.

Steps:

1. Customize your TDP with Data Package Studio (see, e.g., [Generate Your First TDP](#)).
2. Generate your TDP with SC.
 - a. Open SC.
 - b. In the New Job tab:
 - i. Select your scenario ("DPS How-To Guide Scenario").
 - ii. Select your parameter set ("Basic Product Definition Formula")
 - iii. Select your input file ("example_throttle_body_r01.enf").
 - iv. Specify your output folder.
 - v. Start your job.

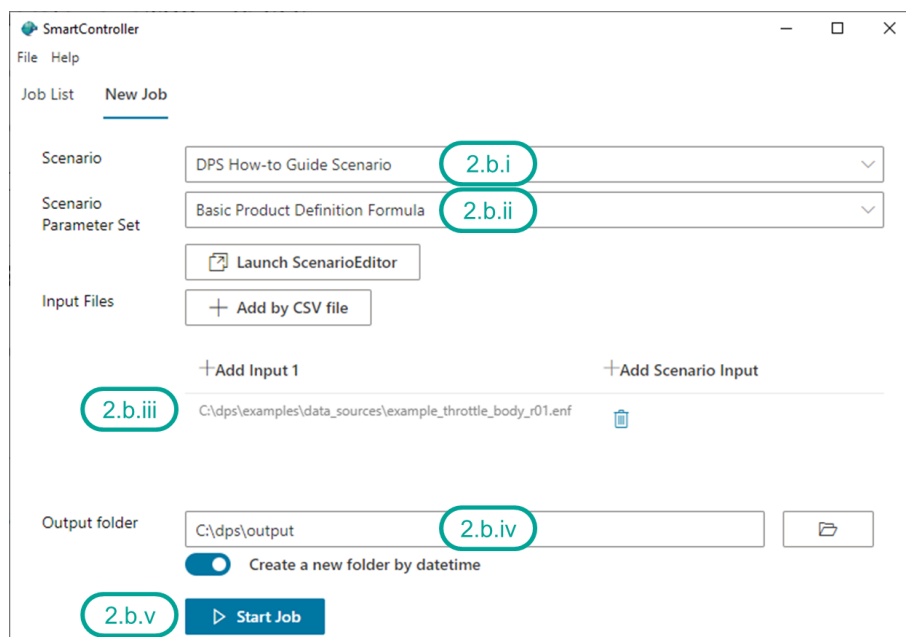


Figure 73. Example of generating a TDP with 3DxSUITE SmartController.

6.3.3. How-to Generate a TDP with 3DxSUITE TransServer

In this guide you will learn how to generate a TDP with 3DxSUITE TransServer (TS).

Prerequisites:

- Access to 3DxSUITE TransServer with the "3DPDF" example scenario imported.



If you cannot find the "3DPDF" scenario, ask your TS administrator to import the "example_ats_scenario.json" from the "examples" folder.

Example Files:

File	Description
"scenarios\example_ats_scenario.json"	3DxSUITE TransServer scenario file.
"formulas\basic_how_to_guide_formula.rb"	Formula file.
"data_sources\example_throttle_body_r01.enf"	Model input.
"data_sources\example_revision_history.csv"	Model revision history input.

The video below will take you through the following steps.

Steps:

1. Customize your TDP with Data Package Studio (see [Generate Your First TDP](#)).
2. Generate your TDP with TS.
 - a. Create a new job.
 - b. Configure the new job.
 - c. Run the configured job.

► <https://vimeo.com/373806155/2e7b02a09c> (Vimeo video)

7. Reserved PDF Field Names



Avoid creating PDF fields with names starting with "ELY_".

When creating your TDP template, avoid naming your fields to something starting with "ELY_" (e.g., "ELY_BoMList"). The reason is that the names of predefined fields in Elysium-provided templates are prefixed "ELY_", and 3DxSUITE requires unique field names to generate a TDP correctly.

8. Frequently Asked Questions

8.1. Definitions

8.1.1. What is DPS?

DPS stands for Data Package Studio.

8.1.2. What is a formula?

A formula is the common name of the three output files of Data Package Studio:

- JSON - file where the data sources, widgets, and actions you define are stored.
- PDF - a copy of your initial TDP template.
- Ruby - script that binds the JSON and TDP template files together.



All three files of a formula are required to generate a TDP.

8.1.3. What is a data source?

A data source represents the files, like ENF or CSV files, that should be used as inputs to 3DxSUITE when generating a TDP. This is enabled by using the data source ID as an 3DxSUITE parameter to specify the file to be used as input.



If you plan to use SC¹, SLS², or TS³ to generate your TDP, then one ENF data source must have "inputfile" as its ID.

1: SC = 3DxSUITE SmartController; 2: SLS = 3DxSUITE SmartLauncher; 3: TS = 3DxSUITE TransServer.

See also [What is a widget?](#) to understand how a data source is presented, e.g., as a BoM table, and [How-to Add a Data Source](#).

The following data sources are supported:

- [ENF \(Elysium Neutral Format\)](#)
- [CSV \(Comma Separated Values\)](#)
- [Text File \(TXT\)](#)
- [Others](#)

ENF (Elysium Neutral Format)

Elysium's proprietary format that is used as a neutral format across Elysium's products.

CSV (Comma Separated Values)

A text file that uses a comma (",") to separate values.

Text File (TXT)

A plain text file that can be used as input for a Text Field widget.

Others

Any other file format not listed above, such as JSON and XML files. These file formats can be added as [attachment](#) to a TDP but cannot be used as input to [widgets](#).

8.1.4. What is a widget?

A widget allows a field in a TDP template to be replaced by more valuable content such as a BoM table or a 3D model to present and interact with product model data. It does so by:

1. Connecting a PDF field with its data source.
2. Specifying how the data in a source should be presented in a PDF field (e.g., as a BoM table).
3. Providing a connection point for a [trigger \(start\)](#) and [action \(end\)](#) pair.

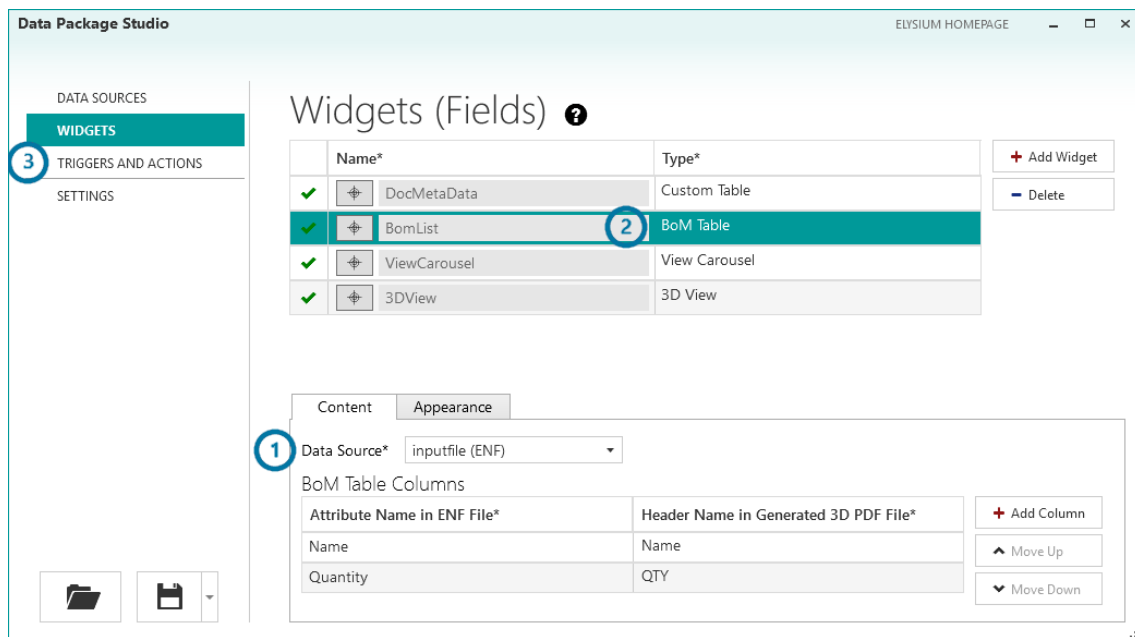


Figure 74. Example of setting a widget.

The following widget types are available:

- [3D View](#)
- [Button](#)
- [BoM Table](#)
- [Custom Table](#)

- [PMI Table](#)
- [Table Column Filter](#)
- [Text Field](#)
- [User Property Table](#)
- [View Carousel](#)

8.1.5. What is a trigger?

A trigger is the event part of a defined interactivity between two widgets (fields) located on the same page within a TDP, e.g., the "on click" event of a BoM table that results in a part in a 3D view being highlighted.

A trigger initiates the execution of an [action](#).

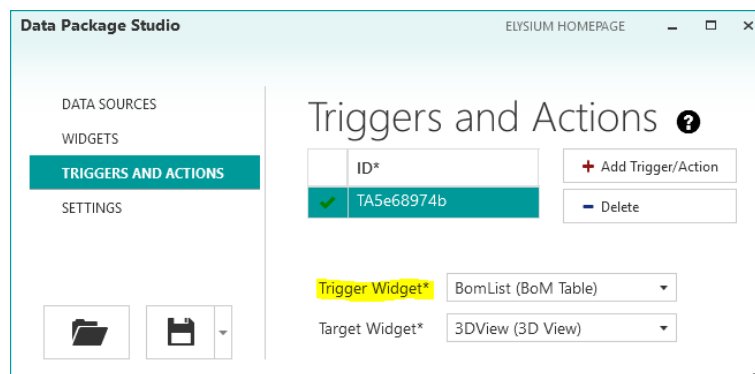


Figure 75. Example of setting a trigger.

See also [What triggers and actions are supported?](#) and [How-to Add a Trigger/Action Connection](#).

8.1.6. What is an action?

An action is the executing part of a defined interactivity between two widgets (fields) located on the same page within a TDP, e.g., the highlighting of a part in a 3D view when a row in a BoM table has been clicked.

An action needs to be [triggered](#) to execute.

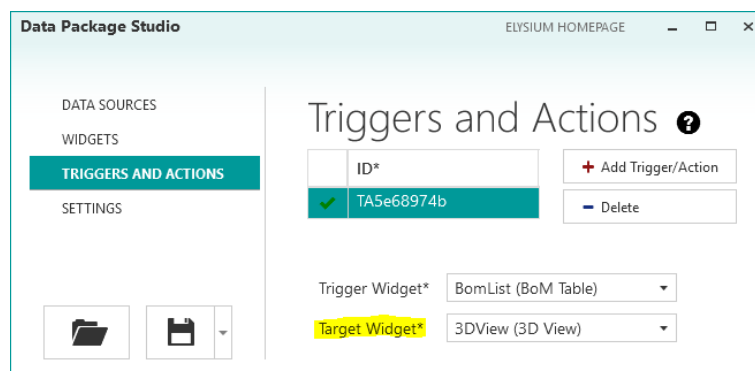


Figure 76. Example of setting an action.

See also [What triggers and actions are supported?](#) and [How-to Add a Trigger/Action Connection](#).

8.1.7. What triggers and actions are supported?

The table below lists the combinations of widgets that can, when located on the same PDF page, be connected by a trigger/action pair in Data Package Studio (DPS).



DPS adds some trigger/action pairs automatically, and these pairs are not shown in the list on the DPS's "Triggers and Actions" page.

Table 10. Supported widget combinations that can be connected via a trigger/action pair.

Trigger	Action	Description	
BoM table	3D view	Event	Part selected
		Result	Part zoomed
		The selected part is zoomed in the 3D view (others are transparent).	
	PMI Table	Event	Part selected
		Result	PMI items listed
		The selected part's PMI items (e.g., geometric tolerances, datums, and notes - depending on the PMI Table widget customization) are filtered in the PMI table. <ul style="list-style-type: none"> • If no part is selected, then all PMI items in the model that matches the PMI Table widget customization are listed. • If the part has no PMI items, then the PMI table is empty. 	
	User Property Table	Event	Part selected
		Result	Part properties listed
		The selected part's user-defined properties are listed as a key/value pair in the user property table, with the property name in the first column and the property value in the second column.	

Trigger	Action	Description	
BoM table	View Carousel	Event	Part selected
		Result	Saved views listed
		<p>The selected part's saved views are filtered in the view carousel.</p> <ul style="list-style-type: none"> • If no part is selected, then all saved views in the model are listed. • If the part has no saved views, then the view carousel is empty. <p>The trigger/action pair is automatically added when adding a trigger/action pair between a BoM Table widget and a 3D View widget that share the same data source.</p>	
Button	Document	Event	Clicked (on mouse up)
		Result	Print dialog is opened
		The print dialog is opened to enable printing of selected views.	
Custom Table	Custom Table	Event	Table item selected
		Result	Table filtered
		The specified property of the selected table item is used as a filter of the connected table. The property can be a Regular Expression.	
	3D View	Event	Table item selected
		Result	View selected
		The specified property of the selected table item is used as the view name to select a saved view.	
CV ¹ Property Table	Document	Event	Table item selected
		Result	URL opened
		<p>If the selected Property Table item is a valid URL, then the URL is opened.</p> <p>1: CAD Validator.</p>	

Trigger	Action	Description	
PMI Table	3D View	Event	PMI item selected
		Result	PMI item highlighted and displayed
		The selected PMI item is highlighted and displayed in the 3D view. Other PMI items referred to from the selected item are also highlighted and displayed.	
Table	Text Field	Event	Table item selected
		Result	Specified property displayed
		The specified property of the selected table item is displayed in the Text Field.	
Table Column Filter	Table	Event	User text input
		Result	Table filtered
		The table is filtered based on the user input matching the content in one or more of the specified columns.	
View Carousel	3D View	Event	Saved view selected
		Result	Saved view displayed
		The selected saved view is displayed in the 3D view. The trigger/action pair is automatically added when adding a View Carousel widget and a 3D View widget that share the same data source.	
	Text Field	Event	View selected
		Result	Specified view property presented
		The selected view's property, as specified in the trigger/action connection, is presented in the text field.	
3D View	PMI Table	Event	PMI clicked (on mouse up)
		Result	PMI selected/highlighted
		The PMI clicked in the 3D View is selected/highlighted in the PMI Table (unless filtered out).	

8.1.8. Why does Data Package Studio save three files?

The three files that Data Package Studio saves make up what we call a *formula* to generate a TDP. See [What is a formula?](#) for more on the formula and the formula files.

8.1.9. What is a predefined property?

A predefined property is defined by a system (e.g., a CAD system or Data Package Studio) and cannot be changed when using the system.

Examples of predefined properties are:

- Part Name in [ENF](#)
- Text Rep. of a PMI item



See also [Predefined Model Properties](#) and [Predefined PMI Properties](#).

8.1.10. What is a user-defined property?

A user-defined property is a property that you as a user has created in, e.g., a CAD system. You can add, delete, and rename these types of properties in any way you like.

8.1.11. What is the difference between pre- and user-defined properties?

See [What is a predefined property?](#) and [What is a user-defined property?](#).

8.2. Extending Functionality

8.2.1. Can I extend the functionality of Data Package Studio?

Currently this is not supported. We may support this in a future release.

8.2.2. Where do I find Data Package Studio API documentation?

Currently Data Package Studio API is not open (see [Can I extend the functionality of Data Package Studio?](#)). Therefore, no documentation is provided.

8.3. Formatting and Customization

8.3.1. Can I add attachments to a TDP in batch?

Yes, you can do this in 3DxSUITE by using the '[FileAttachmentFolder](#)' parameter to specify the location of the files you want to attach.



All files in the specified folder will be attached, but not files in sub-folders.

8.3.2. Can I change the background color of the tables?

You can change the background color of tables starting from Data Package Studio EX8.2. See [How-to Customize a Widget](#) for details on how to do this.

8.3.3. Can I change the formatting of elements/widgets in a CAD Validator report?

You can change formatting of [non-CAD Validator widgets](#) added to a CAD Validator report. You can also customize what to display in a [CAD Validator widget](#), e.g., display only the Passed/Failed assessment of a Difference Summary widget.



See [How-to Customize a CAD Validator Report](#) for details.

However, changing formatting (e.g., font and background colors) of CAD Validator widgets is currently not supported. We may support this in a future release.

8.3.4. Can I change the size of table columns and rows in a TDP?

You can change the width of table columns starting from Data Package Studio EX8.2, but currently you cannot explicitly set the height of rows. You can, however, implicitly set the row height by specifying the number of rows to be displayed in a table.

See [How-to Customize a Widget](#) for details on how to set a table's column width and number of rows to display.

8.3.5. Can I remove columns of a table in the CAD Validator report?

Currently this is not supported. We may support this in a future release.

8.3.6. Can I scroll in the tables generated by Elysium?

Yes, you scroll through a table using the up and down arrows on the right side of the table.

8.3.7. Where can I find the "fit all" and "clear selection" buttons that the CAD Validator report 3D view has?

The 3D view widget in Data Package Studio does not support "fit all" and "clear selection" out-of-the-box. However, you can add [Fit All] and [Clear Selection] buttons by using Elysium's TDP JavaScript API - see [How-to Add a 3D Toolbar using Elysium's TDP JavaScript API](#).

8.3.8. Why is the View Carousel missing thumbnail images?

Confirm that generating thumbnails is enabled, which you do by checking that the 3DxSUITE

enf23dpf 'CreateViewCarouselThumbnail' parameter is not set to '0'. If it is set to '0' then thumbnails are not generated (it is enabled by setting it to '1', default).

If you have enabled generating thumbnails and you still cannot see any thumbnails, then it may be because:

1. The input model (ENF) contains no saved views - confirm that the 'XConvertView' parameter is not set to '0' when converting your original model to ENF.
2. The selected part in the connected [BoM Table](#) widget has no saved views - see [How-to Add and Customize a View Carousel Widget](#) for an explanation.

8.3.9. Will the file name of the attachment be imported automatically?

Yes, the file name of an attachment is automatically imported. It will be used as the name of the attachment in the TDP's list of attachments. The full file path is not imported.

8.4. Installation and License

8.4.1. Do I need a license to use Data Package Studio?

No, since 3DxSUITE EX9.0 you do not need a license to use Data Package Studio.

8.4.2. Where can I find the Adobe Acrobat plug_ins Folder?

You can find the Adobe Acrobat plug_ins folder in the Adobe Acrobat application directory, which is typically found at: "C:\Program Files\Adobe\Acrobat <version>\Acrobat\plug_ins" or "C:\Program Files(x86)\Adobe\Acrobat <version>\Acrobat\plug_ins".

8.5. PDF

8.5.1. How do I add an attachment to a PDF?

Please refer to your application's documentation, e.g., if you are using Acrobat you can find its User Guide linked below.

https://helpx.adobe.com/acrobat/using/links-attachments-pdfs.html#open_save_or_delete_an_attachment

8.5.2. How do I add a date field to a PDF?

Please also refer to your application's documentation, e.g., if you are using Acrobat you can find its User Guide linked below.

<https://helpx.adobe.com/acrobat/kb/support-for-image-and-date-fields-in-acroforms.html>



Adding a date field to your template may cause issues when generating your

TDP (see [Known Issues](#) for details).

8.5.3. How do I add missing fonts to a PDF?

If a PDF does not look like you expected or you cannot interact with its content properly, e.g., add comments, then it may be because the fonts used in the PDF are missing (not embedded in the PDF or missing from your system, e.g., Windows).

If you are using Adobe Acrobat then you may resolve this by installing the Adobe Reader Font Pack on your system. You can download the Adobe Reader Font Pack from the "ADD-ONS" section at the bottom of the pages linked below.

Windows: <https://supportdownloads.adobe.com/product.jsp?product=10&platform=Windows>

Mac: <https://supportdownloads.adobe.com/product.jsp?product=10&platform=Mac>

If you are not using Adobe Acrobat, or if installing the Adobe Reader Font Pack does not resolve the problem, then please refer to your system's documentation for help on how to install missing fonts.

If you are the author of the PDF then have a look at the linked page below to know how you can ensure the fonts you have used are embedded in the PDF.

<https://helpx.adobe.com/acrobat/using/analyzing-documents-preflight-tool-acrobat.html>

8.5.4. How do I add or organize pages in a PDF?

Please refer to your application's documentation, e.g., if you are using Acrobat you can find its User Guide linked below.

<https://helpx.adobe.com/acrobat/how-to/organize-add-delete-pages-pdf.html?playlist=/ccx/v1/collection/product/acrobat-dc/topics/acrobat-practice-tutorials/collection.ccx.js&ref=helpx.adobe.com>

8.5.5. How do I add tabs/buttons to a PDF?

The tabs that are used in Elysium's validation report are each created by three elements:

1. A page that acts as the tab page.
2. A visual element that looks like a tab.
3. An invisible button on top of the visual element that functions like a tab (or more correctly, like a link to the page of the tab).

Please refer to your application's documentation for more details, e.g., if you are using Acrobat you can find its User Guide linked below.

<https://helpx.adobe.com/acrobat/using/setting-action-buttons-pdf-forms.html>

8.5.6. How do I add a watermark to a PDF?

Please refer to your application's documentation, e.g., if you are using Acrobat you can find its User Guide linked below.

<https://helpx.adobe.com/acrobat/using/add-watermarks-pdfs.html>

8.5.7. How do I modify or create editable fields in PDF?

Please refer to your application's documentation, e.g., if you are using Acrobat you can find its User Guide linked below.

<https://helpx.adobe.com/acrobat/user-guide.html?topic=/acrobat/morehelp/forms.ug.js>

8.5.8. How do I create a template?

See [How-to Create a TDP Template](#).

8.5.9. How do I define/set static text and images in PDF?

Please refer to your application's documentation, e.g., if you are using Acrobat you can find its User Guide linked below.

<https://helpx.adobe.com/acrobat/using/edit-text-pdfs.html>

8.5.10. How do I digitally sign a PDF?

Please refer to your application's documentation, e.g., if you are using Acrobat you can find its User Guide linked below.

<https://helpx.adobe.com/acrobat/how-to/fill-and-sign-forms-anywhere.html>

<https://helpx.adobe.com/acrobat/using/certificate-based-signatures.html>



Add digital signatures after a TDP has been generated (see [Known Issues](#)).



If you use a certificate-based signature, then the 3D content may not work as expected. The reason is that 3D content requires changes to the TDP to be enabled while the certificate-based signature does not allow changes.

8.5.11. How do I enable 3D content in a PDF?

Please refer to your application's documentation, e.g., if you are using Acrobat you can find its User Guide linked below.

<https://helpx.adobe.com/acrobat/using/enable-3d-content-pdf.html>

8.5.12. How do I protect a PDF with a password?

Please refer to your application's documentation, e.g., if you are using Acrobat you can find its User Guide linked below.

https://helpx.adobe.com/acrobat/using/securing-pdfs-passwords.html#id_48581

8.5.13. Where do I find attachments in a PDF, and how do I open them?

Please refer to your application's documentation, e.g., if you are using Acrobat you can find its User Guide linked below.

https://helpx.adobe.com/acrobat/using/links-attachments-pdfs.html#open_save_or_delete_an_attachment

8.6. User Interface

8.6.1. Where do I find more information about the UI?

For your convenience, help regarding the User Interface is just a click away when you are using the Data Package Studio (click the [?] button).

9. Example Templates and Data Sources

We have provided a few example files (templates, data sources, formulas, and scenarios) to get you up to speed quickly (see [Example Files](#) for details). You can find them in the "examples" folder located next to the documentation that was provided with Data Package Studio.

You may use the example templates, formulas, and scenarios as-is, or modified versions, to generate your own TDPs that you may use/distribute without limitations.



We kindly ask you to:

1. Remove the "Elysium" logotype from modified templates.
2. Keep the "Empowered by Elysium" logotype in modified templates.

See the Data Package Studio End User License Agreement for legal details.

9.1. Example Files Setup

We recommend that you take the following steps to enable example formulas and scenarios to work out-of-the-box:

1. Copy the "examples" folder to "C:\dps".
2. Copy example scenarios to your "scenario" folder.
3. Configure 3DxSUITE SmartLauncher to run imported example scenarios.

9.1.1. Copy Example Files

Copy the "examples" folder to "C:\dps" so that you will have the following folder structure:

```
c:
|-dps
  |-examples
    |-data_sources
    |-formulas
    |-output
    |-scenarios
    |-templates
```

9.1.2. Copy Example Scenarios

We recommend that you copy the files you find in the "examples\scenarios" folder to your preferred "scenario" folder, e.g., "C:\Users\Public Documents\Elysium\3DxSUITE\Scenarios". This will enable you to use the example scenarios to run repetitive TDP generation tasks from 3DxSUITE SmartLauncher.

9.1.3. Configure 3DxSUITE SmartLauncher to Run Example Scenarios

Next, configure 3DxSUITE SmartLauncher to use the folder where the example scenarios are located as its scenario folder.

To configure 3DxSUITE SmartLauncher:

1. Enable the scenario execution feature in 3DxSUITE SmartLauncher.
 - a. Right click any file and select **Elysium > Preferences**.
 - b. Click the [**Yes**] button (if asked if it is OK to make changes to your device).
 - c. Click the [**Feature**] tab/button.
 - d. Confirm that the "Use scenario execution feature" checkbox is checked.
 - e. Click the [**OK**] button.

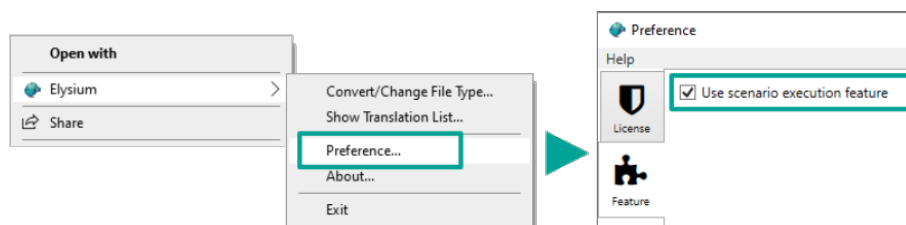


Figure 77. Enable 3DxSUITE SmartLauncher to run scenarios.

2. Specify the location of your scenarios in 3DxSUITE Common Setting.
 - a. From the Windows start menu, select **Elysium 3DxSUITE > 3DxSUITE Common Setting**.
 - b. Click the [**Yes**] button (if asked if it is OK to make changes to your device).
 - c. Click the [**Feature**] tab/button.

- d. Specify the "Scenario Folder" where you stored the example scenario files.
- e. Click the [OK] button.

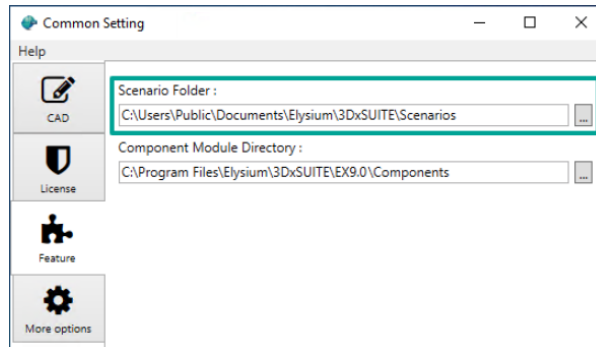


Figure 78. Specify the location of scenario files.

9.2. Example Files

9.2.1. Data Sources

The 3D model data sources (*.enf and *.stp) are provided for your internal use only.

File	Description
example_revision_history.csv	Example revision history for usage in our how-to guides.
example_throttle_body_r00.enf	3D model for usage in our how-to guides - slightly different from ENF file (V2) below.
example_throttle_body_r01.enf	3D model for usage in our how-to guides - slightly different from ENF file (V1) above.
TBM-MBD-R00_BoC.enf	3D model with a few example Bill of Characteristics "balloons".
work_instructions\TB-Assembly-WI.enf	3D model with views that captures Work Instructions steps.
work_instructions\wi_steps.csv	Work Instructions step description, including the name of the corresponding view.
work_instructions\wi_parts.csv	Parts data associated with a Work Instruction step.
work_instructions\wi_tools.csv	Tools data associated with a Work Instruction step.

File	Description
work_instructions\wi_safety.csv	Safety instructions associated with a Work Instruction step.

9.2.2. Formulas

File	Description
basic_how_to_guide_formula.json	Based on the basic_how_to_guide_template.pdf. Used in How-to Guides: Basic Functionality .
basic_how_to_guide_formula.pdf	
basic_how_to_guide_formula.rb	
basic_product_definition.json	Based on the basic_product_definition_template.pdf. Generates a basic single-page product definition TDP.
basic_product_definition.pdf	
basic_product_definition.rb	
boc_formula.json	Based on the boc_example_template.pdf. Used in How-to Add a Bill of Characteristics Table .
boc_formula.pdf	
boc_formula.rb	
cv_formula.json	Based on the cv_example_template.pdf. Generates a customized CAD Validator report.
cv_formula.pdf	
cv_formula.rb	
wi_formula.json	Based on the wi_example_template.pdf. Used in How-to Create a Work Instructions TDP .
wi_formula.pdf	
wi_formula.rb	

9.2.3. Output

For your reference, TDP files generated by example formulas.

File	Description
basic_how_to_guide_output.pdf	Generated by the dps_scenario.esa scenario (Basic How-To Guide Formula parameter set), the example_throttle_body_r01.enf, and example_revision_history.csv files.
basic_product_definition.pdf	Generated by the dps_scenario.esa scenario (Basic Product Definition Formula parameter set), and the example_throttle_body_r01.enf files.
boc_output.pdf	Generated by the dps_scenario.esa scenario (BoC Formula parameter set) and the TBM-MBD-R00_BoC.enf files.
cv_output.pdf	Generated by the cv_scenario.esa scenario, and the example_throttle_body_r00.enf and example_throttle_body_r01.enf files.
wi_output.pdf	Generated by the dps_scenario.esa scenario (Work Instructions Formula parameter set), and the TB-Assembly-WI.enf files.

9.2.4. Scenarios

A scenario specifies what components (Adapters and Optimizers) 3DxSUITE should run, and in what order. It also specifies parameter sets that 3DxSUITE will use when running the components.

File	Description
cv_scenario.esa	Scenario that generates a CAD Validator report with Data Package Studio customizations.
example_ats_scenario.json	3DxSUITE TransServer scenario used in How-to Generate a TDP with 3DxSUITE TransServer .

File	Description
dps_scenario.esa	<p>Scenario used in most of the how-to guides.</p> <p>It includes the following parameter sets that all converts PMIs, incl. flat-to-screen PMIs:</p> <ul style="list-style-type: none"> • Basic How-To Guide Formula <ul style="list-style-type: none"> ◦ Formula: basic_how_to_guide_formula.rb ◦ Data source: example_revision_history.csv ◦ Converts all model views • Basic Product Definition Formula <ul style="list-style-type: none"> ◦ Formula: basic_product_definition.rb ◦ Converts all model views • BoC Formula <ul style="list-style-type: none"> ◦ Formula: boc_formula.rb ◦ Converts no model views • Work Instructions Formula <ul style="list-style-type: none"> ◦ Formula: wi_formula.rb ◦ Data source: <ul style="list-style-type: none"> ▪ wi_steps.csv ▪ wi_parts.csv ▪ wi_tools.csv ▪ wi_safety.csv ◦ Converts top model views

9.2.5. Scripts

We have provided an example pre-processing script to help you structure Bill of Characteristics (BoC) data in ENF so that it can be presented properly in the PMI Table widget. See [How-to Add a Bill of Characteristics Table](#) for details on the script and how to use it.

File	Description
pre_process_boc.rb	Script to prepare BoC data from Elysium generated BoC "balloons" as part of the TDP pre-process.

File	Description
pre_process_template.rb	Template for script to run as part of the TDP pre-process.

9.2.6. Templates

File	Description
basic_how_to_guide_template.pdf	Generates a single-page TDP. Used in Generate Your First TDP and How-to Guides: Basic Functionality .
basic_product_definition_template.pdf	Generates a single-page product definition TDP.
cv_example_template.pdf	Generates a TDP with "Cover Sheet", "Product Definition", "Comparison", and "Help" pages.
pd_example_template.pdf	Generates a TDP with "Cover Sheet", "Product Definition", and "Help" pages.
tdp_example_template.pdf	Generates a single-page product definition TDP.
tdp_example_template_design.pptx	PowerPoint file with the tdp_example_template.pdf template design.
wi_example_template.pdf	Generates a single-page Work Instructions TDP.

Appendix A: 3DxSUITE Parameters

A.1. Required Parameters for the 3D PDF Component

'CustomizedScript' / Customized Script File | Absolute Path

Specify the absolute path to the customized formula (Ruby script) to use when generating a TDP (i.e., the formula generated by Data Package Studio).

'Script' / Script File for 3D PDF Export

Set the parameter to 'Custom', or the 'CustomizedScript' parameter is ignored.

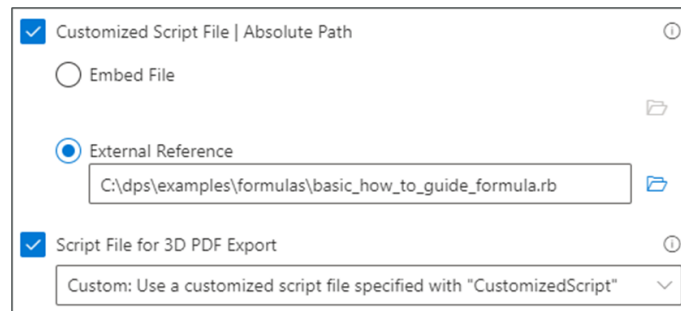


Figure 79. Example usage of required parameters for 3D PDF component.

A.2. Optional Parameters for the 3D PDF Component

'CreateViewCarouselThumbnail' / Create Thumbnail for View Carousel

Specify if thumbnails for View Carousels should be generated ('1', default) or not ('0').

'CustomizedEnfPreProcessScript' / Customized Script for Pre-processing | Absolute Path

Specify the absolute path to the customized script to use in the pre-processing stage when generating a TDP.

'SaveAsHtml' / Export in HTML [DEPRECATED]

Specifies if a TDP should be saved, in addition to PDF, as HTML ('1') or not ('0', default). The HTML file will be saved to the work folder. Note that this function requires Python 3.4 or later (64-bit version is recommended). Note also that it is currently only experimental. So, use it for evaluation, not for production.



[DEPRECATED] means you can still use 'SaveAsHtml', but support will be removed in a future release without further notice.

'SectionCapping' / Display Cross Section with Cap

Specifies if cross sections should be displayed with ('1', default) or without ('0') section caps.

'XConvertFlatToScreenPMI' / How to Convert Flat-to-screen PMI

Specifies how flat-to-screen PMIs are converted - as flat-to-screen ('1', default) or as normal

('0') PMIs. The parameter will only be considered when 'XConvertPMI' is set to '1'.

'XConvertPMI' / Convert PMI

Specifies if PMIs should be converted ('1', default) or not ('0').

'<CustomDataSource>' / Custom Parameters

Any custom data source added in Data Package Studio will be available as a parameter, e.g., 'revision_history'. Specify the absolute path to the file you want to bind to the data source.

Figure 80. Example usage of optional parameters for the 3D PDF component.

A.3. Required Parameters for CAD Validator Component

'3DPdfReportType' / 3D PDF Report | Script File Type for Report Template

Set the parameter to 'Custom', or the 'Customized3DPdfReportScript' parameter is ignored.

'Create3DPdfReport' / Export Validation Result | 3D PDF Report

Set the parameter to '1' to specify that you want the validation report as a 3D PDF.

'Customized3DPdfReportScript' / 3D PDF Report | Script File for Customized Template

Set the parameter to the formula file you want to use to customize your report.

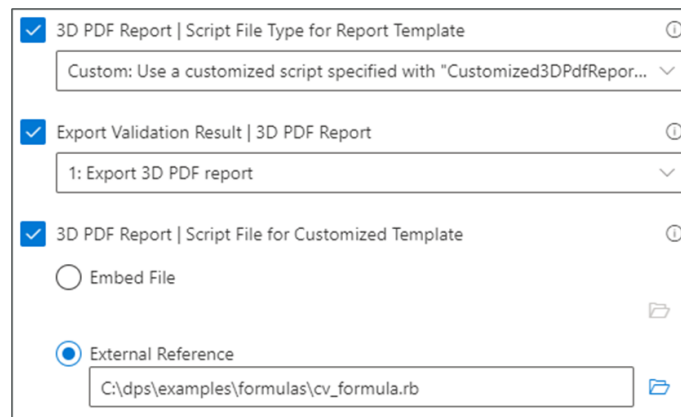


Figure 81. Example usage of required parameters for the CAD Validator component.

A.4. Optional Parameters for CAD Validator Component

'PdfTableMode' / Viewing Performance | 3D PDF Report

Set the parameter to '1' to generate a report with simplified comparison tables for higher performance (default is '0' for normal tables).

'ReportLanguage' / HTML / 3D PDF Report | Display Language

Set the parameter to 'en' to use the English CAD Validator template and to 'ja' to use the Japanese one. Use ISO 639-1 language codes.

'Custom Parameters'

Specify any custom parameter by adding a KEY/VALUE pair separated by an equal sign, i.e., KEY=VALUE. Add one KEY/VALUE pair per line. To set a 3D PDF property, add a KEY/VALUE pair with the KEY prefixed by \$ENF23DPDF_. Example: To set the 3D View(s) background color to red, add \$ENF23DPDF_BackgroundColor=(255,0,0).

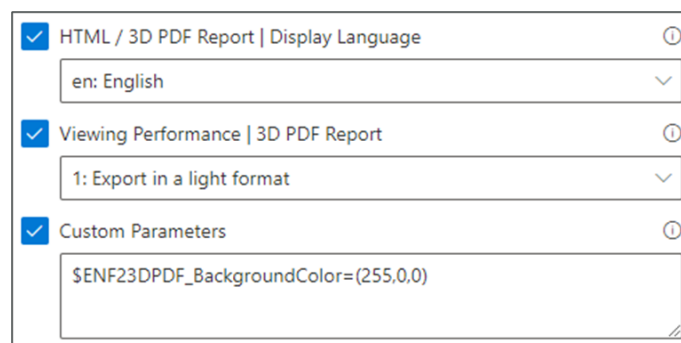


Figure 82. Example usage of optional parameters for the CAD Validator component.

A.5. Optional 3DxSUITE Parameters for CAD Translation

'XConvertDimension' / Convert PMI | Dimension

Set the parameter to '1', or dimensions will not be available in ENF.

'XConvertGDT' / Convert PMI | Geometric Tolerance

Set the parameter to '1', or geometric dimensions and tolerances will not be available in ENF.

'XConvertMaterial' / Convert Attribute | Material

Set the parameter to '1', or material properties will not be available in ENF.

'XConvertNote' / Convert PMI | Note

Set the parameter to '1', or notes will not be available in ENF.

'XConvertSystemProperty' / Convert Attribute | System Property

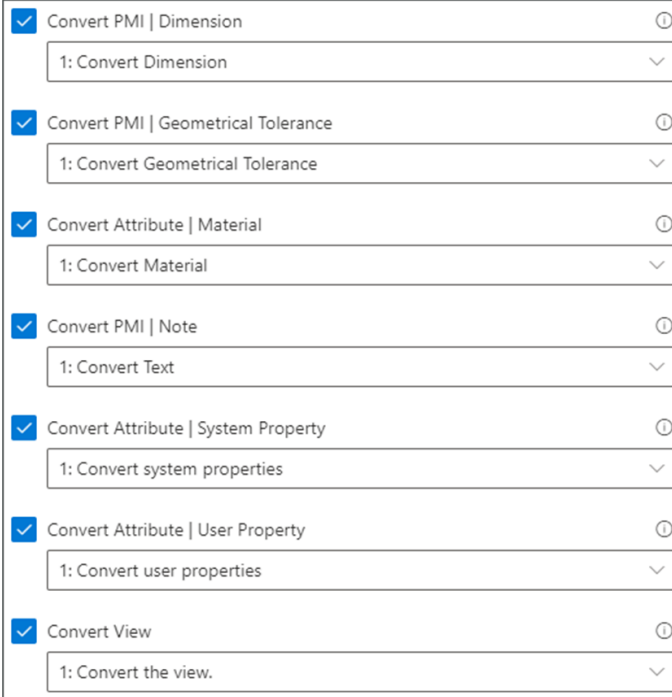
Set the parameter to '1', or predefined properties will not be available in ENF.

'XConvertUserProperty' / Convert Attribute | User Property

Set the parameter to '1', or user-defined properties will not be available in ENF.

'XConvertView' / Convert View

Set the parameter to '1', or saved views will not be available in ENF.



The screenshot shows a settings window with seven rows, each representing a different parameter for CAD translation. Each row has a checked checkbox, a label, a value field, and an information icon. The labels and values are as follows:

Parameter	Value
Convert PMI Dimension	1: Convert Dimension
Convert PMI Geometrical Tolerance	1: Convert Geometrical Tolerance
Convert Attribute Material	1: Convert Material
Convert PMI Note	1: Convert Text
Convert Attribute System Property	1: Convert system properties
Convert Attribute User Property	1: Convert user properties
Convert View	1: Convert the view.

Figure 83. Examples usage of some of the parameters available for CAD translation.

Appendix B: Predefined Model Properties

The table below lists all predefined properties available in Data Package Studio (DPS) and their corresponding properties in ENF. The properties of some major CAD systems/formats that are converted during translation from CAD to ENF are also listed (see [Optional 3DxSUITE Parameters for CAD Translation](#) for details on parameter settings to enable property conversion).

DPS Property	ENF Property	CAD System	CAD Property	Description
Executed CAD Version	ExecutedCadVersion	-	-	CAD version used to convert the native CAD file to ENF.
Native ¹ CAD Version	NativeCadVersion	-	-	CAD system used to create the native CAD file.
Native File Base Name	NFName	-	-	Model file name (including file extension and excluding path).
Native File Modified Date	NativeFileLastWrite	3DEXPERIENCE / CATIA V6	Last modification	Date and time when the native file was last modified.
		NX	Input file last write time	
		STEP	Input file last write time	
		STEP BOM	PartVersion / DateTimeAssignment (Role is update)	
		XPDMXML	TimeModified (Product)	

DPS Property	ENF Property	CAD System	CAD Property	Description
Native File Name	NFName	-	-	Model file path and name (including file extension).
Native File Size	NativeFileSize	-	-	Size of the native file.
Part ² Change History	ChangeHistory	NX I-deas	Change History	Change notes.
Part Comment	PartComment	Inventor	Description	Note/comment attached to the part.
		STEP	PRODUCT_DEFINITION_FORMATION. description	
Part Definition	PartDefinition	CATIA V5	Definition	Definition of the part.
Part Density	Density	Calculated		Density of the part.
Part Description	Description	3DEXPERIENCE / CATIA V6	Description	Description of the part.
		CATIA V5	Description	
		NX I-deas	Description	
		Inventor	Comments	
		MDT	Description	
		STEP	PRODUCT. description	

DPS Property	ENF Property	CAD System	CAD Property	Description
Part Identifier	Name	CATIA V4	Work Space Name	Text that identifies the part.
		CATIA V5	PartNumber	
		CIFO	Object Name	
		NX I-deas	Name	
		Inventor	Name	
		MDT	Name	
		NX	Part Name	
		ProE	Name	
		SolidWorks	Component Name	
		STEP	PRODUCT.id or PRODUCT.name (if no id)	
Part Layer	PartLayer	CATIA V5	Part Layer	Layer where the part is located.
Part Mass	Mass	Calculated		Mass of the part.

DPS Property	ENF Property	CAD System	CAD Property	Description
Part Material	Material	CATIA V5	Material Name	Material of the part.
		NX I-deas	Material Name	
		Inventor	Material	
		MDT	Material	
		ProE	Material	
		SolidWorks	Material	
Part Name	PartName	3DEXPERIENCE / CATIA V6	Name	Name of the part.
		CATIA V5	Nomenclature	
		NX I-deas	Name	
		STEP	PRODUCT.id	
		STEP BOM	Part / Name / CharacterString	

DPS Property	ENF Property	CAD System	CAD Property	Description
Part Number	PartNumber	CATIA V5	PartNumber	Identifier of the part.
		NX I-deas	Part Number	
		Inventor	Part Number	
		STEP	PRODUCT.id	
		STEP BOM	Part / Id / Identifier.id	
Part Revision	PartRevision	3DEXPERIENCE / CATIA V6	Revision	Revision of the part.
		CATIA V5	Revision	
		NX I-deas	Revision	
		Inventor	Revision Number	
		ProE	Revision	
		STEP BOM	PartVersion / Id / Identifier.id	
Part Source	PartSource	CATIA V5	Source	Source of the part (e.g., build, buy).
		STEP	PRODUCT_DEFINITION.make_or_buy	
Part Surface Area	SurfaceArea	Calculated		Surface area of the part.

DPS Property	ENF Property	CAD System	CAD Property	Description
Part Version	PartVersion	NX I-deas	Version	Version of the part.
		STEP	PRODUCT_DEFINITION_FORMATION.id	
Part Volume	Volume	Calculated		Volume of the part.

1: Native refers to the file/system before translation to ENF.

2: Part refers to part or assembly.

Appendix C: Predefined PMI Properties

The tables below list all predefined PMI properties available in Data Package Studio (DPS). Tables are organized with common properties first and then per PMI type in the same order as in the DPS UI.

- [Common](#)
- [Datum](#)
- [Datum Target](#)
- [Dimension](#)
- [Geometric Tolerance](#)
- [Line Weld](#)
- [Locator](#)
- Note (not listed because it only includes properties common to all PMI types)
- [Roughness](#)
- [Spot Weld](#)
- [Bill of Characteristic](#)
- Other (not listed because it only includes properties common to all PMI types)

C.1. Common Properties

Property	Description
Name	The name of a PMI item.

Property	Description
Text Rep.	A text representation of a PMI item (e.g., 1.100 A B D-E).
Type	The type of a PMI item (e.g. note, dimension/distance).

C.2. Datum Properties

Property	Description
Datum Label	The label of a Datum (e.g., A).

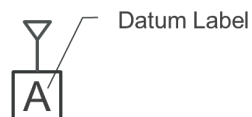


Figure 84. Example of Datum's Datum Label property.

C.3. Datum Target Properties

Property	Description
Lower String (1st)	The datum identifying letter and target number (e.g., A1).
Unit System (1st)	The unit of a Datum Target (e.g., millimeter or inch).
Upper String (1st)	The target area size (e.g., 5x5)



Figure 85. Example of Datum Target properties.

C.4. Dimension Properties

Property	Description
Lower String (1st)	The text below the main Dimension element (e.g., Tapped .875-14 UNF-3B).
Lower String (2nd)	The text below the value/tolerance of a Dimension.
Lower String (3rd)	The text below the value/tolerance (secondary unit) of a Dimension.
Prefix (1st)	The text before the main element of a Dimension (e.g., 4X)
Prefix (2nd)	The prefix of the value/tolerance of a Dimension.
Prefix (3rd)	The prefix of the value/tolerance (secondary unit) of a Dimension.
Subtype	The type of Dimension (e.g., diameter and distance).
Suffix (1st)	The text after the main element of a Dimension (e.g., THRU)
Suffix (2nd)	The suffix of the value/tolerance of a Dimension.
Suffix (3rd)	The suffix of the value/tolerance (secondary unit) of a Dimension.

Property	Description
Tolerance (1st)	The tolerance of a Dimension (e.g., -.100 / +.100).
Tolerance (2nd)	The tolerance (secondary unit) of a Dimension (e.g., -2.540 / +2.540).
Tolerance Class (1st)	The tolerance class of a Dimension (e.g., H7 or h7 - hole fit or shaft fit).
Tolerance Class (2nd)	The tolerance class (secondary unit) of a Dimension (e.g., H7 or h7 - hole fit or shaft fit).
Unit System (1st)	A Dimension's primary unit (e.g., inch).
Unit System (2nd)	A Dimension's secondary unit (e.g., millimeter).
Upper String (1st)	The text above the main Dimension element (e.g., Tapped .875-14 UNF-3B).
Upper String (2nd)	The text above the value/tolerance of a Dimension.
Upper String (3rd)	The text above the value/tolerance (secondary unit) of a Dimension.
Value (1st)	The value of a Dimension (e.g., 2.500).
Value (2nd)	The value (secondary unit) of a Dimension (e.g., 63.500).

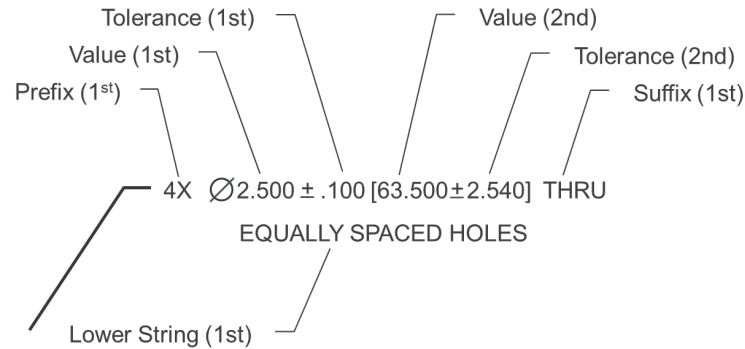


Figure 86. Example of Dimension properties.

C.5. Geometric Tolerance Properties

Property	Description
All Around	Specifies if a Geometric Tolerance is applied "all around" (true) or not (false).
Datum (1st/2nd/3rd, Cell 1)	The primary datum reference of a Geometric Tolerance's first/second/third row (e.g., A).
Datum (1st/2nd/3rd, Cell 2)	The secondary datum reference of a Geometric Tolerance's first/second/third row (e.g., B).
Datum (1st/2nd/3rd, Cell 3)	The tertiary datum reference of a Geometric Tolerance's first/second/third row (e.g., C).
Geom. Tol. Characteristic (1st/2nd/3rd)	The characteristic of a Geometric Tolerance's first/second/third row (e.g., position).
Lower String (1st)	The text below a Geometric Tolerance.
Prefix (1st/2nd/3rd)	The text before a Geometric Tolerance's first/second/third row.
Projected Value (1st/2nd/3rd)	The projected value of a Geometric Tolerance's first/second/third row.

Property	Description
Suffix (1st/2nd/3rd)	The text after a Geometric Tolerance's first/second/third row.
Tolerance (1st/2nd/3rd)	The tolerance of a Geometric Tolerance's first/second/third row (e.g., 1.100).
Unit System (1st)	The unit of a Geometric Tolerance (e.g., millimeter or inch).
Upper String (1st)	The text above a Geometric Tolerance.

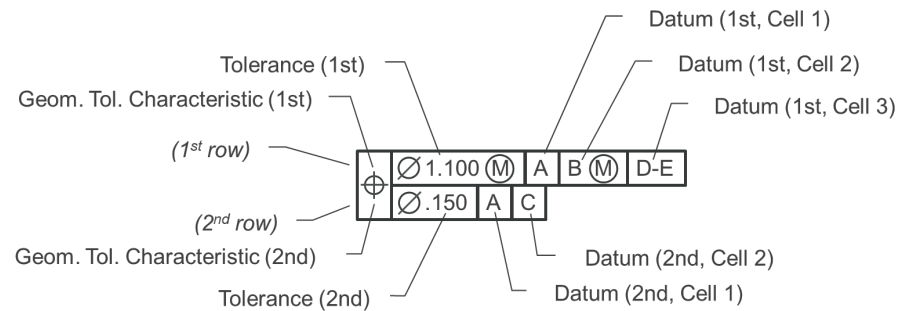


Figure 87. Example of Geometric Tolerance properties.

C.6. Line Weld Properties

Property	Description
All Around	Specifies if a Line Weld is applied "all around" (true) or not (false).
Coordinated Entity	Specifies non-associative labels to, e.g., identify parts that are being welded together.
Field Symbol	Specifies if a field symbol is visible (true) or not (false).

Property	Description
Groove Angle (Lower/Upper)	The lower/upper groove angle of a Line Weld.
Longitudinal Size (Lower/Upper)	The longitudinal size of a Line Weld.
Reference Flag	Specifies if the Line Weld is used as a reference (true) or not (false).
Root Radius (Lower/Upper)	The root radius of a Line Weld.
Root Spacing (Lower/Upper)	The root spacing of a Line Weld.
Special Directive	Specifies the special directive of a Line Weld.
Staggered Flag	Specifies if the Line Weld is staggered (true) or not (false).
Staggered Size (Lower/Upper)	The lower/upper staggered size of a Line Weld.
Surface Condition (Lower/Upper)	The lower/upper surface condition of a Line Weld (e.g., convex or flat).
Surface Finish (Lower/Upper)	The lower/upper surface finish of a Line Weld (e.g., F or G).
Test Method	The test method of a Line Weld (e.g., mt or pt_f).
Total Size (Lower/Upper)	The total size of a Line Weld.
Weld Size (Lower/Upper)	The weld size of a Line Weld.
Weld Type (Lower/Upper)	The weld type of a Line Weld.

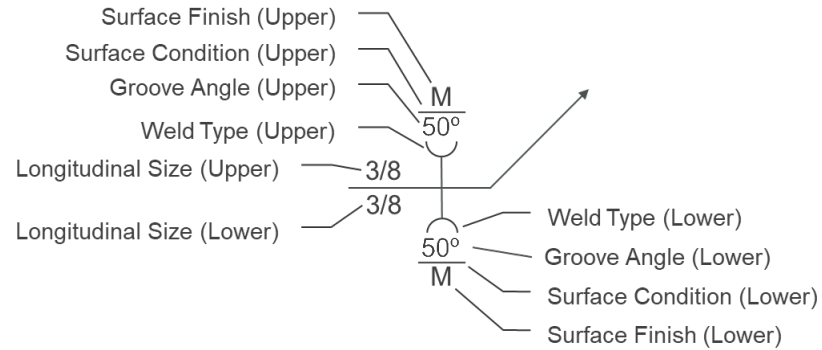


Figure 88. Example of Line Weld properties.

C.7. Locator Properties

Property	Description
Coordinated Entity	Specifies non-associative labels to, e.g., provides context for the locator.
Locator Note	The note of a Locator.
Locator Part Number	The part number a Locator refers to.
Locator Reference	Specifies if Locator is a reference (true) or not (false).
Locator Symbol Type	The symbol type of a Locator (e.g., edge and rectangle).
Locator Type	The type of a Locator.
Subtype	The subtype of a Locator (e.g., fixing and molds).

C.8. Roughness Properties

Property	Description
All Around	Specifies if a Roughness is applied "all around" (true) or not (false).
Field 5	The text/value of Field 5.
Field 6	The text/value of Field 6.
Field 7	The text/value of Field 7.
Field 8	The text/value of Field 8.
Field 9	The text/value of Field 9.
Field 10	The text/value of Field 10.

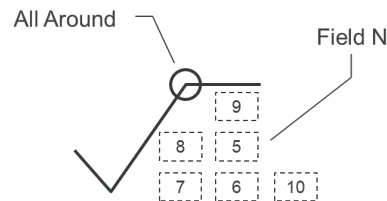


Figure 89. Example of an All Around Roughness property.

C.9. Spot Weld Properties

Property	Description
Coordinated Entity	Specifies non-associative labels to, e.g., identify points on parts or mating panels for alignment.
Critical	Specifies if a Spot Weld is critical (yes) or not (no) or unknown.
Diameter	The diameter of a Spot Weld.
Geometry Flag	Specifies if the Spot Weld is geometry (true) or not (false).
Group ID	The ID of the group that a Spot Weld belongs to.
Inspection Flag	Specifies if the Spot Weld is inspected (true) or not (false).
Joint ID	The ID of the join that a Spot Weld belongs to.
Life-Cycle Standard	The life-cycle standard of a Spot Weld.
Manufacturing Code	The manufacturing code of a Spot Weld.
Process	The process of a Spot Weld.
Shape Type	The shape type of a Spot Weld (e.g., hemisphere or shank).
Subtype	The subtype of a Spot Weld (e.g., resistance_2 or projection_3).
Weld Type	The weld type of a Spot Weld (e.g., edge or seam).

C.10. Bill of Characteristic Properties

Property	Description
Ref. PMI Type	The PMI type that the Bill of Characteristic item refers to (e.g., note, dimension).

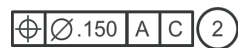


Figure 90. Example of a Bill of Characteristics balloon for a Geometric Tolerance type of PMI.

Appendix D: Minimum Widget Sizes

A widget with a placeholder size less than its minimum size (width or height) listed in [Table 11](#) may fail to be created.



The minimum size of a widget only guarantees that it can be created. It does not guarantee that it makes a widget useful for any intended purpose.

The minimum size listed for a widget has been measured by:

- Including all optional items for the widget, e.g., checkboxes.
- Setting its border thickness to the minimum value **0** (no border).
- Setting its scrollbar size to the minimum value **5** (Tables only).

Therefore, if you change the border or scrollbar size of a widget, then you need to adjust the widget placeholder size in your template accordingly. In addition, for Tables, you need to consider the number of columns and rows to display as both will impact the minimum width and height respectively. For the View Carousel, you need to consider the number of thumbnails (visible views) that it will display.



All listed sizes are rounded up, which may result in sizes not always fully matching, e.g., $2.118 \text{ mm} = 0.083385827 \text{ inch}$ but $2.2 \text{ mm} \neq 0.084 \text{ inch}$.

The sizes in Data Package Studio (DPS) are given in the point unit. **1** point is equivalent to around **0.353 mm**, or **0.014 inch**. Consequently, you have to add $2 \times 0.353 \approx 2.2 \text{ mm}$ to a widget's listed minimum size if you set its border thickness to **2**.

For more examples on minimum size adjustments, see [Example 1](#) through [Example 6](#).

Table 11. Minimum width (W) and height (H) for widget placeholders.

Widget	W/H	Size mm (inch)	Note
Non-CAD Validator Widgets			
3D View	W	1.0 (0.040)	-
	H	1.0 (0.040)	
Button	W	1.0 (0.040)	-
	H	1.0 (0.040)	
Table Column Filter	W	7.5 (0.296)	-
	H	2.5 (0.099)	

Widget	W/H	Size mm (inch)	Note
Tables	W	7.5/col (0.296/col)	Depends on the number of columns, see Example 1 .
	H	5.0 + 1.0/row (0.200 + 0.040/row)	Depends on the number of rows, see Example 1 .
Text Field	W	2.5 (0.099)	-
	H	2.5 (0.099)	
View Carousel (no printing)	W	3.6 + 5.5/view (0.138 + 0.217/view)	Depends on the scrollbar size and the number of visible views, see Example 2 .
	H	16.8 (0.659)	Depends on the scrollbar size, see Example 2 .
View Carousel (printing)	W	7.6 + 7.5/view (0.296 + 0.296/view)	Depends on the scrollbar size and the number of visible views, see Example 3 .
	H	16.8 (0.659)	Depends on the scrollbar size, see Example 3 .
CAD Validator Widgets			
3D View	W	20.0 - 131.5 (0.788 = 5.186)	Depends on customization, see Example 4 .
	H	20.0 (0.788)	-
Assembly Tree	W	30.0 (1.182)	-
	H	10.0 (0.394)	-
CAD Information	W	20.0 (0.788)	-
	H	17.5 (0.689)	-
Color Bar	W	98.3 (3.871)	-
	H	6.3 (0.249)	-
Component Summary	W	140.0 (5.512)	-
	H	23.0 (0.906)	-

Widget	W/H	Size mm (inch)	Note
Detail List	W	12.1 - 113.5 (0.477 - 4.469)	Depends on customization, see Example 5 .
	H	27.0 (1.063)	-
Difference Summary	W	23.0 - 114.0 (0.906 - 4.490)	Depends on customization, see Example 6 .
	H	14.0 (0.552)	-
Execution Property	W	20.0 (0.788)	-
	H	12.5 (0.493)	-
Property Table	W	180.0 (7.087)	-
	H	12.5 (0.493)	-
Type Summary	W	80.0 (3.150)	-
	H	23.0 (0.906)	-
Validation Setting	W	40.0 (1.575)	-
	H	12.5 (0.493)	-

Example 1. Minimum size of Table with 3 columns and 6 rows.

Width:

$$22.5 \text{ mm} = 3^A \times 7.5^B$$

$$0.888 \text{ inch} = 3^A \times 0.296^B$$

Height:

$$11.0 \text{ mm} = 5.0^C + 6 \times 1.0^D$$

$$0.437 \text{ inch} = 0.197^C + 6 \times 0.040^D$$

A: Number of columns. B: Width per column. C: Fixed (header height). D: Height per row.

Example 2. Minimum size of View Carousel with 9 points scrollbar size and 5 visible views.

Width:

$$30.7 \text{ mm} = 9^A \times 0.353^B + 5^C \times 5.5^D$$

$$1.211 \text{ inch} = 9^A \times 0.014^E + 5^C \times 0.217^D$$

Height:

$$19.9 \text{ mm} = 9^A \times 0.353^B + 15.0^F$$

$$0.716 \text{ inch} = 9^A \times 0.014^E + 0.590^F$$

A: Scrollbar in points. B: Conversion rate from point to mm. C: Number of visible view thumbnails. D: Width of a view thumbnail.

E: Conversion rate from point to inch. F: Fixed size (view and view name).

Example 3. Minimum size of View Carousel with printing, 9 points scrollbar size, and 5 visible views.

Width:

$$34.7 \text{ mm} = 4.0^A + 9^B \times 0.353^C + 5^D \times 5.5^E$$

$$1.349 \text{ inch} = 0.138^A + 9^B \times 0.014^F + 5^D \times 0.217^E$$

Height:

$$19.9 \text{ mm} = 9^B \times 0.353^C + 15.0^G$$

$$0.716 \text{ inch} = 9^B \times 0.014^F + 0.590^G$$

A: "Check All" Checkbox. B: Scrollbar in points. C: Conversion rate from point to mm. D: Number of visible view thumbnails. E: Width of a view thumbnail.

F: Conversion rate from point to inch. G: Fixed size (view, view name, and checkbox).

Example 4. Size of CAD Validator 3D View with all options.

Width:

$$131.5 \text{ mm} = 2 \times 1.8^A + 4 \times 1.5^B + 21.2^C + 28.3^D + 30.0^E + 30.0^F + 5.7^G + 5.7^H + 1.0^I$$

$$5.186 \text{ inch} = 2 \times 0.071^A + 3 \times 0.060^B + 0.835^C + 1.115^D + 0.182^E + 1.182^F + 0.225^G + 0.225^H + 0.040^I$$

A: Widget padding (left/right). B: Margins between fields (excl. Action and Refresh). C: Clear selection button. D: Fit option selector.

E: Renderer mode selector. F: Geometry type selector. G: Action menu. H: Refresh button. I: 3D View width.

Example 5. Size of CAD Validator Detail List with all options.

Width:

Minimum Width of Title/Navigation Area:

$$20.0 \text{ mm} = 9.6^A + 9.6^B$$

$$0.756 \text{ inch} = 0.378^A + 0.378^B$$

Minimum Width of Toolbar Area:

$$113.5 \text{ mm} = 2 \times 1.1^C + 17.6^D + 5.3^E + 15.2^F + 9.9^G + 13.4^H + 5.8^E + 44.1^I$$

$$4.473 \text{ inch} = 2 \times 0.044^C + 0.693^D + 0.209^E + 0.599^F + 0.390^G + 0.528^H + 0.229^E + 1.737^I$$

Minimum Width of List Area:

$$45.0 \text{ mm} = 15.0^J + 30.0^K$$

$$1.773 \text{ inch} = 0.591^J + 1.182^K$$

Minimum Width:

$$113.5 \text{ mm} = \max(20.0^L, 113.5^M, 45.0^N)$$

$$4.473 \text{ inch} = \max(0.756^L, 4.473^M, 1.773^N)$$

A: Previous button. B: Next button. C: Toolbar padding (left/right) . D: Show Diff Only checkbox. E: Margin between fields. F: Geometry checkbox.
G: PMI checkbox. H: Attributes checkbox. I: Model View selector. J: Group list. K: Element list. L: Title/Navigation area. M: Toolbar area. N: List area.

Example 6. Size of CAD Validator Difference Summary with all options.

Width:

$$114.0 \text{ mm} = 2 \times 3.2^A + 3 \times 2.2^B + 16.6^C + 30.2^D + 24.0^E + 30.2^F$$

$$4.490 \text{ inch} = 2 \times 0.126^A + 3 \times 0.087^B + 0.654^C + 1.189^D + 0.945^E + 1.189^F$$

A: Widget padding (left/right). B: Margins between fields. C: Assessment field. D: Geometry differences summary. E: PMI differences summary.
F: Attribute differences summary.

Appendix E: Elysium TDP JavaScript API

We introduced the Elysium TDP JavaScript API (TDP JS API for short), in EX8.3 to help you make more advanced customization of a TDP, such as adding a 3D toolbar. You can get hold of the API via the global object named `ely3dpdf` and its function `getApi` - see [Function: getApi](#).

The API depends on, and is implemented by, a handful of JavaScript classes. These classes and their respective functions are described in the sections below.

We have used the following terms to describe the classes and functions:

- **Description** - a brief description of a class/function.
- **Since** - the version when the class/function was introduced in the API.
- **Input** - a brief description of a function's named input parameters and types.
- **Returns** - a brief description of a function's return value and type.
- **Example** - example code to call a function.



We are using `var` instead of `const` and `let` in the code examples to be compatible with Adobe Acrobat's JavaScript engine.

E.1. Class: Ely3dpdf

Description	Class for the global object <code>ely3dpdf</code> to access the TDP JS API.
Since	EX8.3

E.1.1. Function: getApi

Description	Returns the TDP JS API.
Since	EX8.3
Returns	Returns an instance of the TDP JS API (ElyTdpApi).

Example

```
var api = ely3dpdf.getApi();
```

E.2. Class: ElyTdpApi

Description	Class that implements a TDP general API.
Since	EX8.3

E.2.1. Function: getField

Description	Returns a named field (widget placeholder) of your template.		
Since	EX8.3		
Input	<table> <tr> <td>name</td><td>A string that specifies the name of a field in your template (same as the template widget placeholder name).</td></tr> </table>	name	A string that specifies the name of a field in your template (same as the template widget placeholder name).
name	A string that specifies the name of a field in your template (same as the template widget placeholder name).		
Returns	Returns a field object matching the input name, or null if there are no fields matching the name.		

Example

```
var api = ely3dpdf.getApi();
var field = api.getField('3D View widget'); ①

console.println('The ' + field.name + ' field is located on Page ' field.page);
```

① You can find more about the returned **field** object and its properties in the [Adobe Acrobat SDK documentation](#).

E.2.2. Function: getWidget

Description	Returns a named widget.		
Since	EX8.3		
Input	<table> <tr> <td>name</td><td>A string that specifies the name of a widget in your template (same as the template widget placeholder name).</td></tr> </table>	name	A string that specifies the name of a widget in your template (same as the template widget placeholder name).
name	A string that specifies the name of a widget in your template (same as the template widget placeholder name).		
Returns	Returns a widget (ElyTdpWidget) matching the input name, or null if there are no widgets matching the name.		

Example

```
var api = ely3dpdf.getApi();
var widget = api.getWidget('3D View widget');
```

E.3. Class: ElyTdpWidget

Description	Class that implements an API that is common among widgets.
Since	EX8.3

E.3.1. Function: getType

Description	Returns the widget type.
Since	EX8.3
Returns	Returns the widget's type name as a string .

Example

```
var api = ely3dpdf.getApi();
var widget = api.getWidget('3D View widget');
var type = widget.getType();

if (type === 'ElyTdp3dViewWidget') {
  // Do something with a 3D View widget
}
```

E.4. Class: ElyTdp3dViewWidget



For an instance of ElyTdp3dViewWidget to work properly, the corresponding 3D view must be active.

Description	Class that implements a 3D View widget API. Extends ElyTdpWidget .
Since	EX8.3

E.4.1. Function: reset

Description	Resets a 3D View widget (part, pmi, view, and render mode).
Since	EX8.3

Example

```
var api = ely3dpdf.getApi();
var widget = api.getWidget('3D View widget');
// ...
widget.reset();
```

E.4.2. Function: fitAll

Description	Fits all parts in a 3D View widget's viewport (all parts visible in the selected saved view).
Since	EX8.3

Example

```
var api = ely3dpdf.getApi();
var widget = api.getWidget('3D View widget');
// ...
widget.fitAll();
```

E.4.3. Function: fitSelection

Description	Fits selected part/PMI in a 3D View widget's viewport.
Since	EX8.3

Example

```
var api = ely3dpdf.getApi();
var widget = api.getWidget('3D View widget');
// ...
widget.fitSelection();
```

E.4.4. Function: clearSelection

Description	Clears selected part/PMI in a 3D View widget.
Since	EX8.3

Example

```
var api = ely3dpdf.getApi();
var widget = api.getWidget('3D View widget');
// ...
widget.clearSelection();
```

E.4.5. Function: getRenderModes

Description	Gets a list of a 3D View widget's valid render modes.
Since	EX8.3
Returns	Returns a list of valid render modes as an array of string .

Example

```
var api = ely3dpdf.getApi();
var widget = api.getWidget('3D View widget');
// ...
var modes = widget.getRenderModes(); ①
modes.forEach(mode => {
  console.println(mode);
});
```

① See [Function: setRenderMode](#) below for details on valid render modes.

E.4.6. Function: setRenderMode

Description	Sets the render mode of a 3D View widget.	
Since	EX8.3	
Input	name	<p>A string that specifies the render mode name, which must be a valid name or the render mode will remain unchanged.</p> <p>See renderMode Scene property in the Adobe Acrobat SDK documentation for a list of valid render modes.</p>

Example

```
var api = ely3dpdf.getApi();
var widget = api.getWidget('3D View widget');
// ...
widget.setRenderMode('illustration');
```

E.4.7. Function: resetRenderMode

Description	Resets a 3D View widget's render mode to its initial value.
Since	EX8.3

Example

```
var api = ely3dpdf.getApi();
var widget = api.getWidget('3D View widget');
// ...
widget.resetRenderMode();
```

E.4.8. Function: isValidRenderMode

Description	Checks if a name is a valid render mode name.	
Since	EX8.3	
Input	name	A string that specifies the render mode name.
Returns	Returns true if the input name is a valid render mode, otherwise false .	

Example

```
var api = ely3dpdf.getApi();
var widget = api.getWidget('3D View widget');
// ...
if (widget.isValidRenderMode('bounding box')) {
    // Do something
};
```

E.4.9. Function: getViews

Description	Returns a list of names of the 3D View widget's views.
Since	EX8.3
Returns	Returns a list of view names as an array of string .

Example

```
var api = ely3dpdf.getApi();
var widget = api.getWidget('3D View widget');
// ...
var views = widget.getViews();
views.forEach(view => {
  console.println(view);
});
```

E.4.10. Function: setView

Description	Sets the view of a 3D View widget.	
Since	EX8.3	
Input	name	A string that specifies the view name, which must be a valid name or the 3D View will remain unchanged.
	animate	An optional, Boolean input that specifies if the 3D View widget should animate the transition to the new view (true) or not (false , default).

Example

```
var api = ely3dpdf.getApi();
var widget = api.getWidget('3D View widget');
// ...
widget.setView('MBD-001D', true);
```

E.4.11. Function: resetView

Description		Resets the view of a 3D View widget to its initial view.
Since		EX8.3
Input	animate	An optional, Boolean input that specifies if the 3D View widget should animate the transition to the initial view (true) or not (false , default).

Example

```
var api = ely3dpdf.getApi();
var widget = api.getWidget('3D View widget');
// ...
widget.resetView();
```

E.4.12. Function: isValidView

Description		Checks if a name is a valid view name.
Since		EX8.3
Input	name	A string that specifies the view name.
Returns		Returns true if the input name is a valid view name, otherwise false .

Example

```
var api = ely3dpdf.getApi();
var widget = api.getWidget('3D View widget');
// ...
if (widget.isValidView('MBD-001D')) {
    // Do something
};
```

E.4.13. Function: getShowPmi

Description		Gets a 3D View widget's show/hide PMI value.
Since		EX8.3

Returns	Returns true if the current setting of a 3D View widget is to show PMIs, otherwise false .
----------------	--

Example

```
var api = ely3dpdf.getApi();
var widget = api.getWidget('3D View widget');
// ...
if (widget.getShowPmi()) {
  console.println('PMIs are shown!');
}
```

E.4.14. Function: setShowPmi

Description		Sets show/hide PMI.
Since		EX8.3
Input	value	A Boolean that specifies if PMIs should be shown (true) or hidden (false).

Example

```
var api = ely3dpdf.getApi();
var widget = api.getWidget('3D View widget');
// ...
widget.setShowPmi(false)
```

E.5. Class: ElyTdpTableWidget

Description	Class that implements a Table widget API. Extends ElyTdpWidget .
Since	EX8.3.2

E.5.1. Function: clearFiltering

Description	Clears the filtering of a Table widget.
Since	EX8.3.2

Example

```
var api = ely3dpdf.getApi();
var widget = api.getWidget('PT_PmiTable');
// ...
widget.clearFiltering();
```

E.5.2. Function: clearSelection

Description	Clears the item selection of a Table widget.
Since	EX8.3.2

Example

```
var api = ely3dpdf.getApi();
var widget = api.getWidget('PT_PmiTable');
// ...
widget.clearSelection();
```

E.5.3. Function: getData

Description	Returns the data (content) of a Table widget. See also Function: getHeader .	
Since	EX8.3.2	
Input	filtered	<p>An optional Boolean that specifies if the data should be returned filtered (true) or unfiltered (false, default).</p> <p>If the table is not filtered, then the unfiltered data will be returned even if filtered is set to true.</p>
Returns	Returns an Array of Arrays of strings where each of the inner arrays represent a table row and the array elements represent the content of the cells in that row.	

Example

```

var api = ely3dpdf.getApi();
var widget = api.getWidget('BT_BomTable');
// ...
var data = widget.getData(); ①
var numRows = data.length;
for (var row = 0; row < numRows; row++) {
    var rowData = data[row];
    var numCols = rowData.length;
    for (var col = 0; col < numCols; col++) {
        var cellData = rowData[col];
        // ...
    }
}
// ...

```

① Input **true** to get the data filtered.

E.5.4. Function: getHeader

Description	Returns the header labels of a Table widget. See also Function: getData .
Since	EX8.3.2
Returns	Returns an Array of strings of the header labels (i.e., what you see in the table header in the TDP).

Example

```

var api = ely3dpdf.getApi();
var widget = api.getWidget('BT_BomTable');
// ...
var header = widget.getHeader();
var numCols = header.length;
for (var col = 0; col < numCols; col++) {
    var label = header[col];
    // ...
}
// ...

```

E.5.5. Function: reset

Description	Resets a Table widget (filtering, selection, sorting).
Since	EX8.3.2

Example

```
var api = ely3dpdf.getApi();
var widget = api.getWidget('BT_BomTable');
// ...
widget.reset();
```

E.5.6. Function: resetSorting

Description	Resets the sorting of a Table widget to its original sort order.
Since	EX8.3.2

Example

```
var api = ely3dpdf.getApi();
var widget = api.getWidget('PT_PmiTable');
// ...
widget.resetSorting();
```

E.6. Class: ElyTdpTextWidget

Description	Class that implements a Text widget API. Extends ElyTdpWidget .
Since	EX9.1.0

E.6.1. Function: getReadOnly

Description	Returns a Text widget's read-only value. See also Function: setReadOnly .
Since	EX9.1.0
Returns	Returns true if a Text widget is read-only (cannot be edited), otherwise false (can be edited).

Example

```

var api = ely3dpdf.getApi();
var widget = api.getWidget('TF_PartName');
// ...
widget.setReadOnly(false);
assert(widget.getReadOnly() === false);
// ...

```

E.6.2. Function: setReadOnly

Description		Sets a Text widget's read-only value. See also Function: getReadOnly .
Since		EX9.1.0
Input	value	A Boolean that specifies if the Text widget is read-only (true) or read-and-write (false).

Example

```

var api = ely3dpdf.getApi();
var widget = api.getWidget('TF_PartName');
// ...
widget.setReadOnly(true);
assert(widget.getReadOnly() === true);
// ...

```

E.6.3. Function: toggleReadOnly

Description		Changes a Text widget's read-only value to the opposite value (from read-only to read-and-write, and vice versa) and returns the new value. See also Function: getReadOnly and Function: setReadOnly .
Since		EX9.1.0
Returns		Returns the new value, i.e., true if the Text widget is read-only (cannot be edited) after its been toggled, otherwise false (can be edited).

Example

```
var api = ely3dpdf.getApi();
var widget = api.getWidget('TF_PartName');
// ...
assert(widget.getReadOnly() === !widget.toggleReadOnly());
// ...
```

Appendix F: Migrating to the New CAD Validator Report Template Format

This appendix will explain how to migrate your existing, customized CAD Validator report templates to the new format introduced with Data Package Studio (DPS) EX9.0. You are required to migrate to the new format if you want to take advantage of DPS's ability to customize CAD Validator widgets, such as hiding the Failed/Passed assessment in the Difference Summary table (see image below).

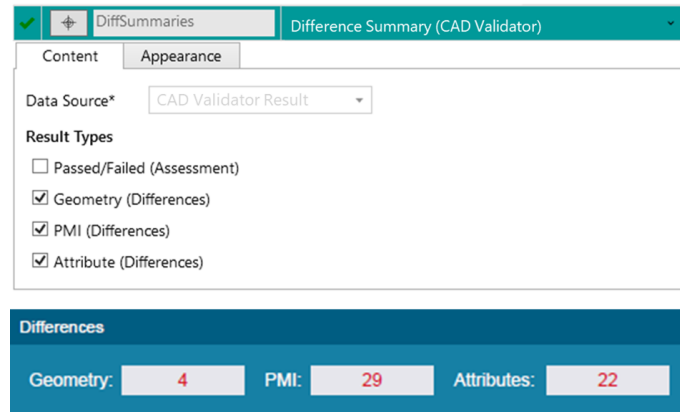


Figure 91. Example of hiding the Failed/Passed assessment in the Difference Summary table (DPS option top and the generated table without the Failed/Passed assessment bottom).

However, you do not need to migrate to the new format if you just want to generate a TDP without widget customizations. You can use your pre-EX9.0 templates as-is and generate a TDP in 3DxSUITE EX9.0 too.

Of course, this appendix will also provide information for how you create a CAD Validator report template in the new format from scratch.



Some widgets, like the [Property Table](#), cannot be customized. However, you can always exclude widgets by removing them from your template/formula.

First, to provide some context for the migration steps, we will cover the [CAD Validator widget types](#) that are supported in Data Package Studio since EX9.0. Then we will walk you through the steps required to migrate a CAD Validator report template to the new format:

1. [Merge the CAD Validator 3D View Widget Related Placeholders.](#)
2. [Rename the CAD Validator Widget Placeholders.](#)
3. [Create a Formula in Data Package Studio.](#)

F.1. CAD Validator Widget Types

There are 11 CAD Validator widgets. We will briefly introduce them and the placeholder they correspond to in the old CAD Validator report template below (including the page you find them

on in the original template).

F.1.1. 3D View (CAD Validator)

The 3D View widget in CAD Validator differs from the DPS standard 3D View widget. It visualizes the geometry of, and differences between, two models instead of the geometry of a single model.

As you can see in the image below, the old CAD Validator report template actually have four placeholders for one widget (all on Page 1):

- ELY_CV_3DController
- ELY_CV_3DHeader
- ELY_CV_ViewSource
- ELY_CV_ViewTarget

However, to simplify, only one placeholder will be needed starting from DPS EX9.0. This is also the reason for the [first migration step](#) - these four placeholder must be merged into one.

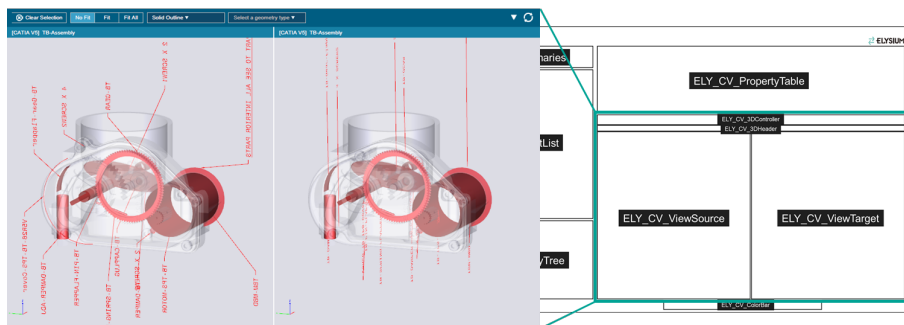


Figure 92. The 3D View widget and the placeholders it corresponds to in the template.

You can customize what to display (show/hide) in the toolbar (from left to right):

- Clear Selection button
- Fit Mode selector
- Render Mode selector
- Geometry Type Mode selector
- Action Menu (the down arrow to the right)
 - Show PMI checkbox
 - Show Unselected Parts checkbox
 - Highlight Transparent Faces checkbox
- Refresh button

All are shown by default.

F.1.2. Assembly Tree (CAD Validator)

The Assembly Tree corresponds to the placeholder on Page 1 named "ELY_CV_AssemblyTree" (see image below). It shows the assembly trees for the two models being compared, and is currently not customizable.

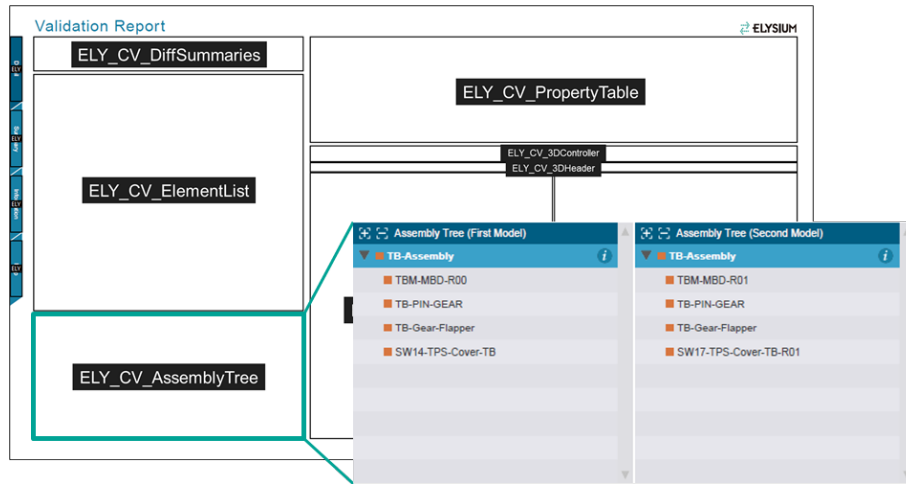


Figure 93. The Assembly Tree widget and the placeholder it corresponds to in the template.

F.1.3. CAD Information (CAD Validator)

The CAD Information corresponds to the placeholder on Page 3 named "ELY_CV_CadInfo" (see image below). It lists CAD information for the two models being compared, such as the CAD version of each model. It is currently not customizable.

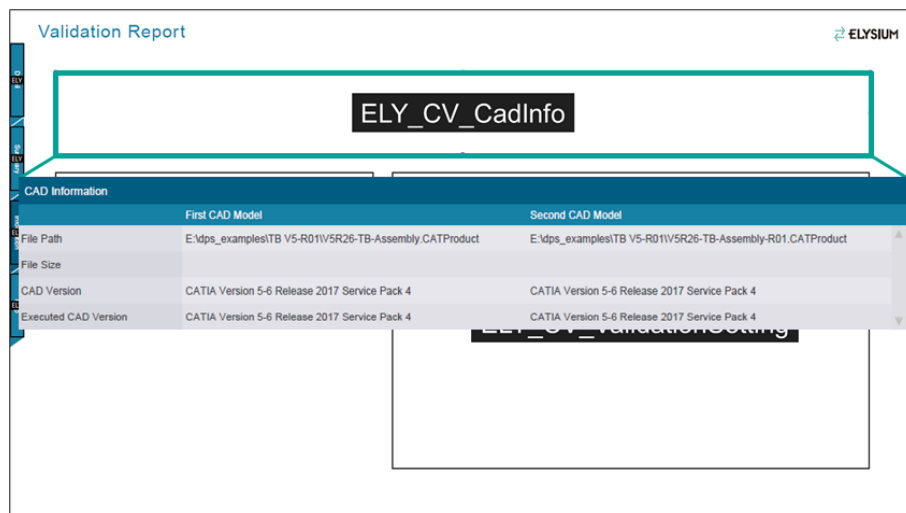


Figure 94. The CAD Information widget and the placeholder it corresponds to in the template.

F.1.4. Color Bar (CAD Validator)

The Color Bar corresponds to the placeholder on Page 1 named "ELY_CV_ColorBar" (see image below). It shows the color coding scale used when visualizing differences in a 3D View widget. It is currently not customizable via DPS but you can use the CAD Validator parameter 'PdfEmbedColorMap' to turn it off (0) / on (1, default).

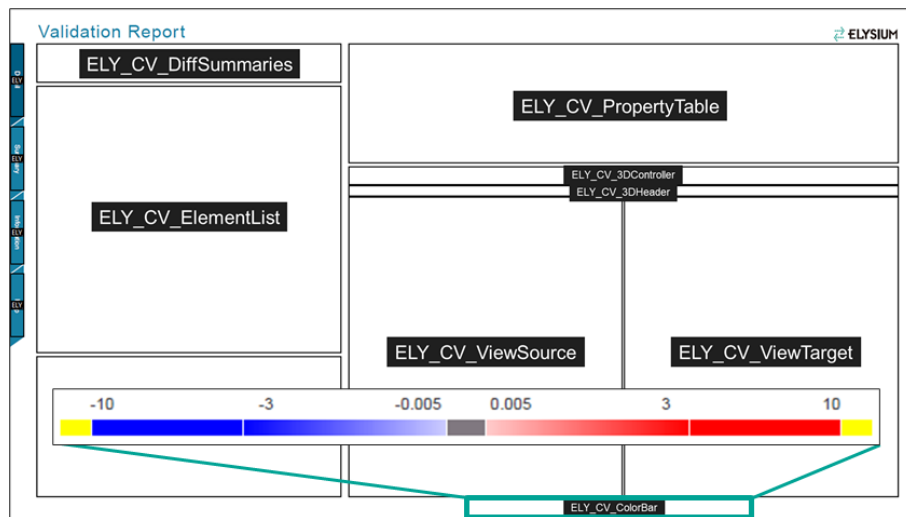


Figure 95. The Color Bar widget and the placeholder it corresponds to in the template.

F.1.5. Component Summary (CAD Validator)

The Component Summary corresponds to the placeholder on Page 2 named "ELY_CV_ComponentSummary" (see image below). It summarizes the results of the comparison of the two models per component.

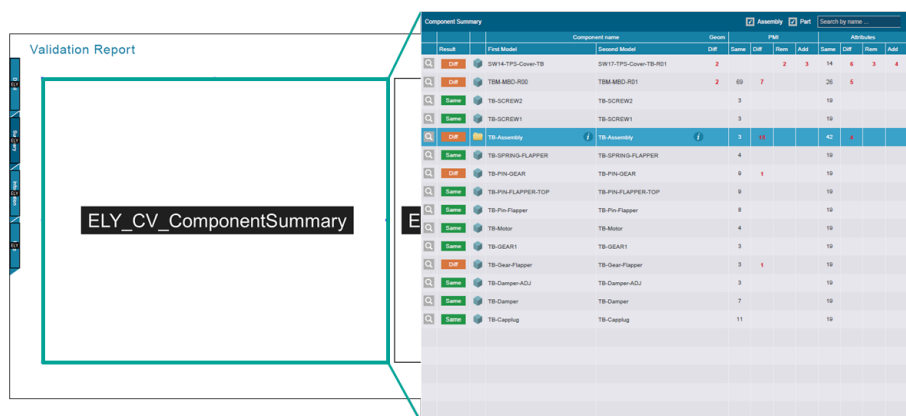


Figure 96. The Component Summary widget and the placeholder it corresponds to in the template.

You can customize what to display (show/hide) in the title bar (from left to right):

- Assembly checkbox
- Part checkbox
- Search field

All are shown by default.

F.1.6. Detail List (CAD Validator)

The Detail List corresponds to the placeholder on Page 1 named "ELY_CV_ElementList" (see image below). It shows the results of the comparison of the two models from two perspectives:

1. Group List: Manually or automatically created groups of results

2. Element List: Elements and their respective result

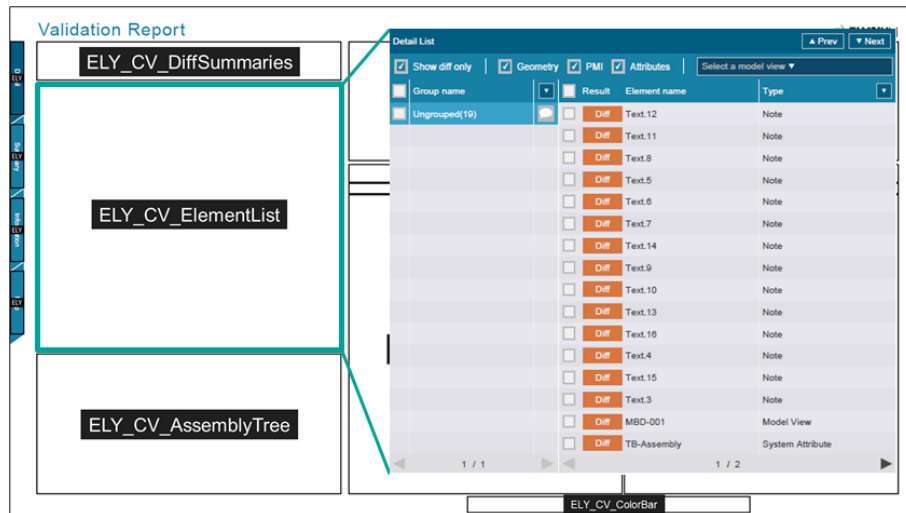


Figure 97. The Detail List widget and the placeholder it corresponds to in the template.

You can customize what to display (show/hide) in the title bar, toolbar, and detail list (from left to right):

- Title bar:
 - Prev button
 - Next button
- Toolbar:
 - Show Diff Only checkbox
 - Geometry checkbox
 - PMI checkbox
 - Attributes checkbox
 - Model View selector
- Detail Lists:
 - Group List
 - Action Menu
 - Add Group action
 - Edit Group action
 - Remove Group action
 - Element List
 - Action Menu
 - Move to Group action
 - Copy to Group action
 - Remove from Group action

All are shown by default.

F.1.7. Difference Summary (CAD Validator)

The Difference Summary corresponds to the placeholder on Page 1 named "ELY_CV_DiffSummaries" (see image below). It shows a summary of the result of comparing two models in terms of Failed/Passed, and the number of differences in each category (Geometry, PMI, Attributes).

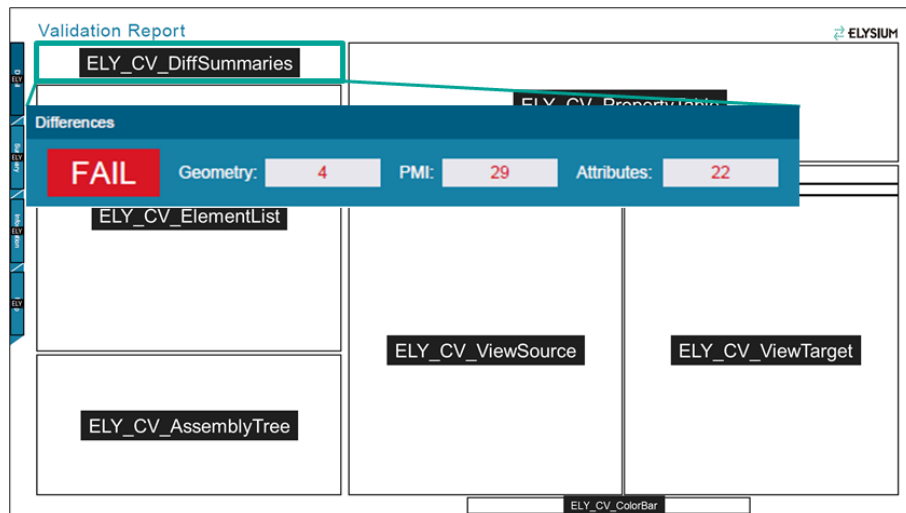


Figure 98. The Difference Summary widget and the placeholder it corresponds to in the template.

You can customize what result types to display (show/hide) in the widget (from left to right):

- Passed/Failed assessment
- Geometry differences
- PMI differences
- Attribute differences

All are shown by default.

F.1.8. Execution Property (CAD Validator)

The Execution Property corresponds to the placeholder on Page 3 named "ELY_CV_ExecProperty" (see image below). It lists the properties related to running the CAD Validator component (e.g., processed date/time, processing time). It is currently not customizable.

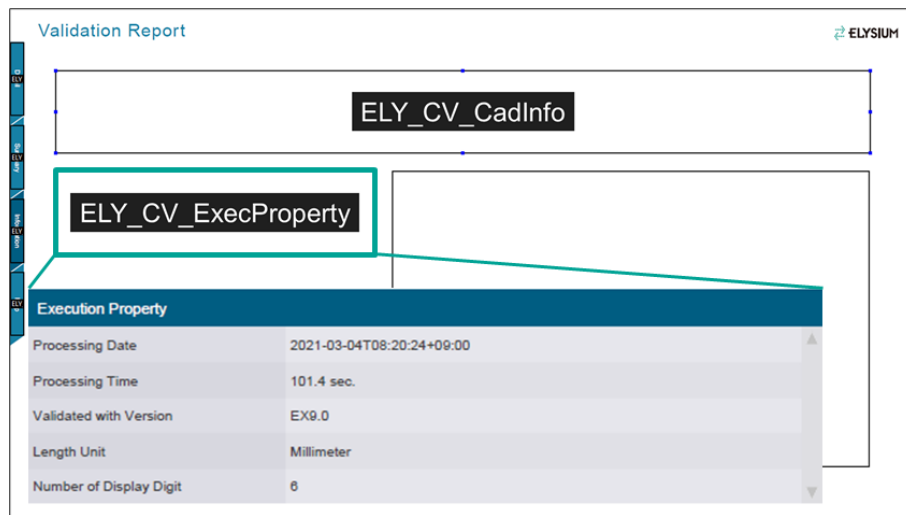


Figure 99. The Execution Property widget and the placeholder it corresponds to in the template.

F.1.9. Property Table (CAD Validator)

The Property Table corresponds to the placeholder on Page 1 named "ELY_CV_PropertyTable" (see image below). It lists the properties related to the difference(s) selected in the Detail List, and is currently not customizable.

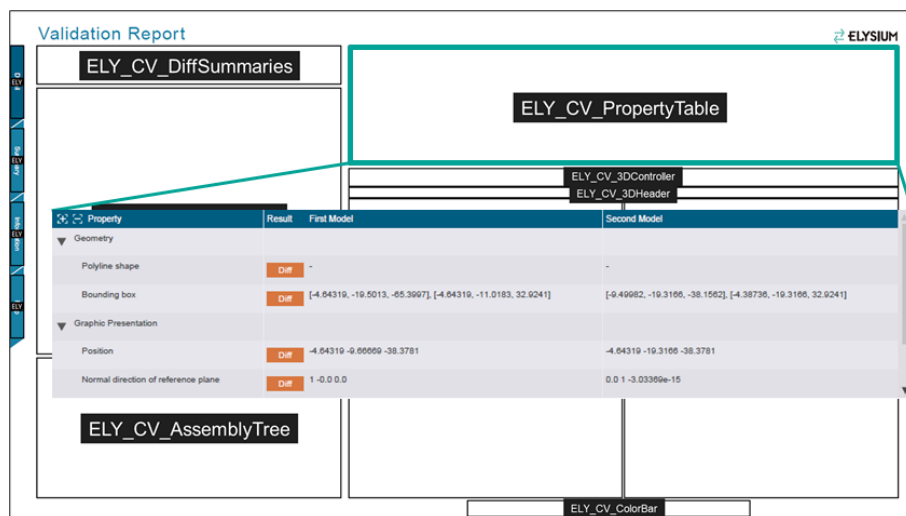


Figure 100. The Property Table widget and the placeholder it corresponds to in the template.

F.1.10. Type Summary (CAD Validator)

The Type Summary corresponds to the placeholder on Page 2 named "ELY_CV_TypeSummary" (see image below). It summarizes the differences per type of element for the component selected in the Assembly Tree widget.

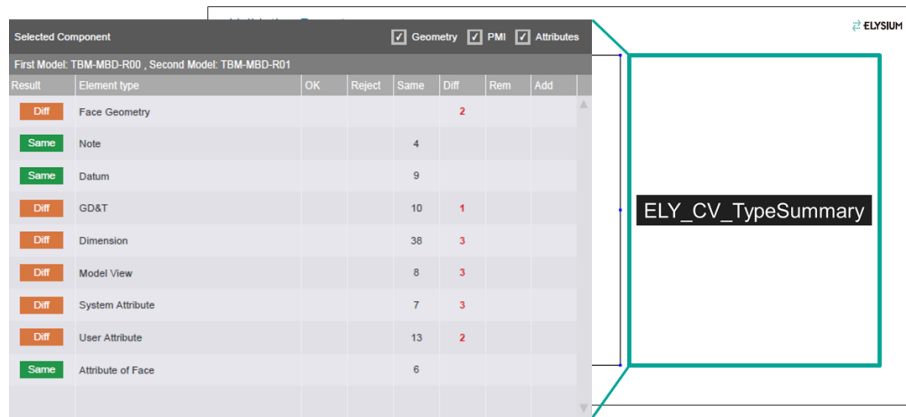


Figure 101. The Type Summary widget and the placeholder it corresponds to in the template.

You can customize which controls to display (show/hide) in the title bar (from left to right):

- Geometry checkbox
- PMI checkbox
- Attributes checkbox

All are shown by default.

F.1.11. Validation Setting (CAD Validator)

The Validation Setting corresponds to the placeholder on Page 3 named "ELY_CV_ValidationSetting" (see image below). It lists the settings used when comparing the two models, and is currently not customizable.

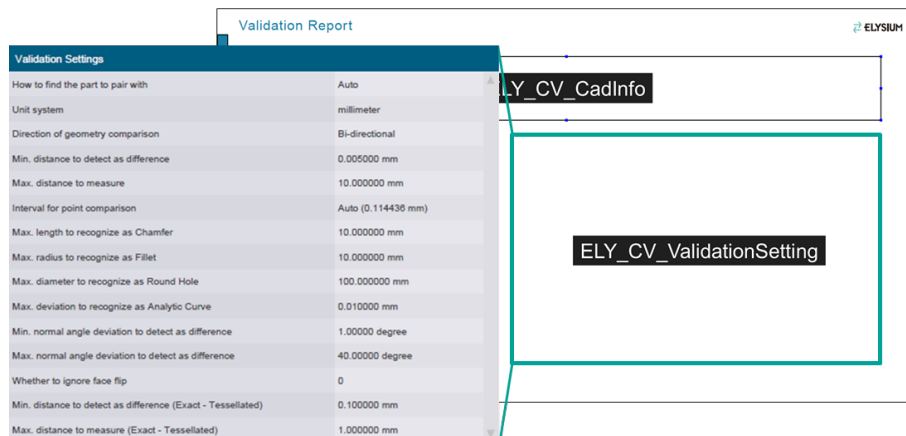


Figure 102. The Validation Setting widget and the placeholder it corresponds to in the template.

F.2. Merge the CAD Validator 3D View Widget Related Placeholders

Now that you know about the different CAD Validator widgets, it is time to start the migration process. First you need to merge the four placeholders for the [3D View \(CAD Validator\)](#) to one.

Open your template in your favorite editor and enable form editing ("Prepare Form" in Adobe

Acrobat Pro). Delete three of the placeholders (your choice). Then move and resize the one placeholder you decided to keep (see the image below for an example).

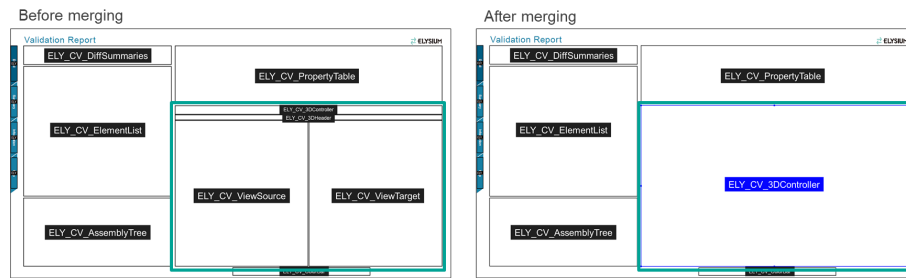


Figure 103. Example of merging the four 3D View related placeholders to one.

Save the template (use a new name if you want to keep the old template for backup).

F.3. Rename the CAD Validator Widget Placeholders

The next step in the migration process is to rename the placeholders to eliminate the risk of name conflicts. You need to rename all the placeholders that starts with 'ELY_'. The reason is that Elysium uses this as a prefix for elements that we have created/generated (see [Reserved PDF Field Names](#)).



You can name your placeholders anything as long as the names do not start with 'ELY_'.

So, the set of placeholders you need to rename includes at least all the placeholders remaining from the original template. You also need to rename the invisible buttons of the navigation pane, unless you have already removed them from your customized template. After renaming, you should have something looking like the example in the image below.

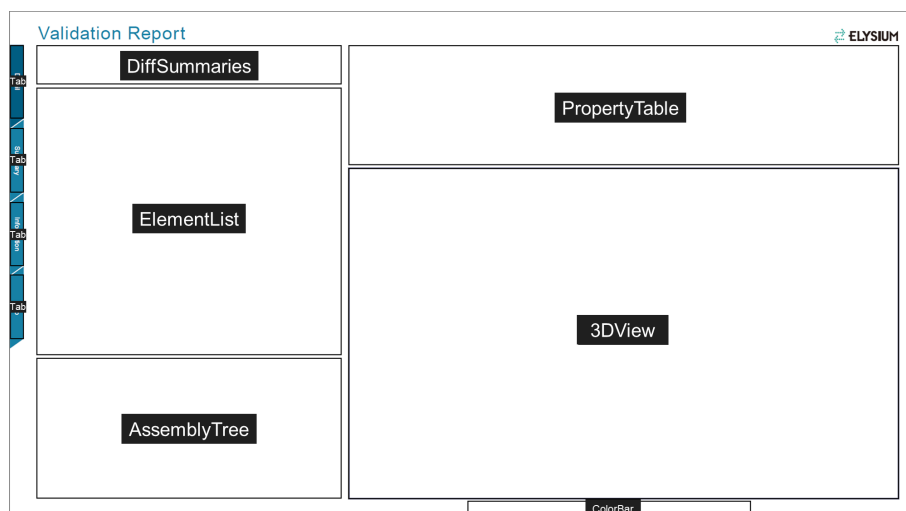


Figure 104. Example of renaming the placeholders in a template.

Save the template.

F.4. Create a Formula in Data Package Studio

The final step in your migration process is to create a formula. Please refer to the [How-to Customize a CAD Validator Report](#) section for how to customize a CAD Validator report.

Appendix G: Example TDP Pre-Process Script

Listing 10. Example TDP pre-process script to prepare Bill of Characteristics data in ENF.

```
# -*- encoding: UTF-8 -*-
module PdfEditor ①
  class CustomizedEnfPreProcess ②
    logger.info "Custom File ({_FILE_}): "

    def on_run_before(ee_session) ③
      ee_session.model.components.each do |compo| ④
        # Create mapping from BoC ID to referenced PMI
        id_to_ref = Hash.new
        compo.annotations.each do |anno| ⑤
          md = anno.name.match(/EREF\(([\\d;]+)\\)/) ⑥
          next if md.nil? || md.size != 2
          md[1].split(";").each do |id|
            id_to_ref[id] = anno ⑦
          end
        end

        # Flag BoC and set referenced PMI
        compo.annotations.each do |anno| ⑧
          md = anno.name.match(/EBOC(\\d+)/) ⑨
          next if md.nil? || md.size != 2
          ref_anno = id_to_ref[md[1]] ⑩
          next if ref_anno.nil?
          anno.boc_tag = true ⑪
          anno.boc_ref = ref_anno ⑫
        end
      end
    end
  end
end
```

- ① The script "lives" in the PdfEditor module,
- ② and the CustomizedEnfPreProcess class.
- ③ The on_run_before method is where the script is defined.
- ④ For each component/part, repeat Steps 5-12.
- ⑤ For each annotation of a component/part, repeat Steps 6-7.
- ⑥ Find all PMIs that have a name that contains EREF(<number>[;<number>;...]), where number is the reference to a BoC identifier.

- ⑦ Store the found PMIs in a hash map with the corresponding BoC's ID as key (to enable lookup in the next steps).
- ⑧ For each annotation of a component/part, repeat Steps 9-12.
- ⑨ Find all annotations (BoCs) with a name that contains EBOC<number>, where number is the unique identifier of the BoC.
- ⑩ Get the corresponding PMI stored in Step 7.
- ⑪ Tag the annotation (BoC) by setting its boc_tag property to **true** (this is what turns a "normal" annotation into a BoC).
- ⑫ Connect the BoC with its PMI by the boc_ref property (this enables highlight/zoom also of the PMI in a 3D view when a BoC is selected in a PMI table).

Appendix H: Known Issues

Table 12. Known Issues - New in Data Package Studio EX9.1.

Issue	Description
Performance of 3D PDF reports (incl. CAD Validator and PDQ) in Adobe Acrobat (Software renderer)	<p>The Software renderer sometimes results in a significantly slower response time as compared to DirectX rendering . Occasionally reports may become unresponsive.</p> <p>Workaround: Acrobat 32bit: Use the DirectX renderer. Acrobat 64bit: No workaround.</p>
Incorrect element (e.g., face) color for Adobe Acrobat (Software renderer)	<p>Changing the element color requires the DirectX renderer. Consequently, cases like highlighting the related elements of a PMI or changing the color of an element in a view will not be rendered as expected using the Software renderer option. An element will render with its original color instead of the highlight/specified color.</p> <p>Workaround: Acrobat 32bit: Use the DirectX renderer. Acrobat 64bit: No workaround.</p>
Adobe Acrobat crashes when opening generated TDP	<p>In recent releases, Adobe Acrobat sometimes crashes when opening a generated TDP.</p> <p>Workaround: Fixed in EX9.1. For reports generated prior to EX9.1, once opened in Adobe Acrobat, save the TDP to eliminate crashes when opening the TDP the next time.</p>
Incorrect background color of icons	<p>Icons are not displayed with transparent background. Temporarily, icon background colors have been set to match the color scheme of the default CAD Validator report template.</p> <p>Workaround: No workaround.</p>
Missing geometry in PDQ report	<p>In some exceptional cases, errors in a PDQ report may be missing some of the geometry in the original model.</p> <p>Workaround: No workaround.</p>

Issue	Description
Incorrect visibility in views	<p>In some exceptional cases in very complex models, the visibility of some elements may not be rendered as expected.</p> <p>Workaround: No workaround.</p>
Regression in tessellation accuracy	<p>In some cases, the result of tessellation is less accurate.</p> <p>Workaround: No workaround.</p>
Running out of memory	<p>In cases with exceptional large models, thousands of instances, Adobe Acrobat may run out of memory when opening a TDP, and some/all widgets may not be rendered.</p> <p>Workaround: No workaround.</p>
Incorrect element color	<p>In some exceptional cases, the color of some elements may not be rendered as expected.</p> <p>Workaround: No workaround.</p>
Incorrect rendering of characters in Text Fields	<p>When displaying content from a text file in a Text Field, some characters may be rendered as "□".</p> <p>Workaround: Ensure that the template, or the machine where the report is opened, has the correct fonts embedded/installed.</p>
Incorrect rendering when opening a TDP on an OS in another language	<p>When opening a TDP on an OS in a language other than the language of the OS it was generated on (and report language), the TDP may not be rendered correctly. Ex. Opening a Japanese CAD Validator report in a German OS.</p> <p>Workaround: Avoid generating a TDP in OS and report language combinations other than where it will be used.</p>
Incorrect HTML report rendering (SaveAsHtml)	<p>Zooming in/out when loading a HTML report (generated by SaveAsHtml) may cause it to be rendered incorrectly.</p> <p>Workaround: Do not zoom in/out when loading the HTML. Reload/refresh the HTML report.</p>

Issue	Description
3D View in CAD Validator HTML report stops working (SaveAsHtml)	<p>Under some conditions, e.g., a combination of selecting items in a table and the 3D View, the 3D View may stop working with a "TypeError" displaying in a red banner at the top of the report.</p> <p>Workaround: Reload/refresh the HTML report.</p>

Table 13. Known Issues.

Issue	Description
At least one ENF input file is required to generate a TDP	<p>The 3DxSUITE frontends require at least one ENF input file to be provided or it will not generate a TDP.</p> <p>Workaround: Provide a ENF file as a “dummy” (even if not used in the formula).</p>
Adobe Acrobat no longer checks for missing fonts	<p>Recent versions of Adobe Acrobat no longer checks and warns about missing fonts. As a result, it will not automatically ask if you want to download missing fonts.</p> <p>Workaround: Check and download missing fonts manually - see FAQ How do I add missing fonts to a PDF?.</p>
File size increase when saving a TDP	<p>Known Adobe Acrobat issue that results in an increase of the TDP's file size every time you save it. Adobe Acrobat adds data to track changes.</p> <p>Workaround: Save your TDP with File › Save As....</p>
Inconsistent PMI highlighting	<p>Issue occurs when a trigger/action connection from a PMI Table widget to a 3D View widget exists but is missing in the other direction.</p> <p>Results in PMI items selected in the 3D view to be highlighted differently from when selected from the PMI table.</p> <p>Workaround: Add a trigger/action connection from the 3D View widget to the PMI Table widget.</p>

Issue	Description
Incorrect formatting of text including Asian language characters	<p>Results in text including Asian language characters to be displayed slightly below the expected location, and is caused by the font used in the TDP.</p> <p>Workaround: Not available.</p>
Incorrect handling of non-ASCII characters in formula file path	<p>Results in an error in the 3DxSUITE enf23dpdf component that prevents the generation of a TDP. The error occurs when the path to the formula includes non-ASCII characters in the folders or formula names.</p> <p>Note that the path separator character (e.g., "\" or "¥") can be a non-ASCII character without causing an error.</p> <p>Workaround: Use only ASCII characters when naming folders and formula files.</p>
Incorrect PMI rendering in views	<p>In some exceptional cases, e.g., when a cross section and a PMI's reference plane are on the same plane, the PMI is not rendered properly in views.</p> <p>Workaround: Offset the cross section and the PMI's reference plane.</p>
Limited handling of rotated pages	<p>Results in content not showing properly or missing.</p> <p>Workaround: Do not rotate pages.</p>
Limited handling of the 3D background color	<p>Results in 3D background color not showing as expected, when multiple 3D views are used. The background color of the 3D view listed last in the formula (JSON) will be used for all 3D views.</p> <p>Workaround: Not available.</p>
Limited handling of CSV formatting	<p>Results in CSV content not showing properly or missing.</p> <p>Workaround: Use the following format:</p> <ul style="list-style-type: none"> • Separator: Comma (,) • Header row: Required (first row in CSV is interpreted as header row)

Issue	Description
Limited handling of parallel on screen annotations	<ol style="list-style-type: none"> 1. Displayed as a normal annotation when we cannot determine it contains text only (e.g., disabled when it includes a leader). 2. Not visible in View Carousel thumbnails. <p>Workaround: Not available.</p>
Limited handling of PDF field formats	<p>Results in content not showing properly or missing.</p> <p>Workaround: Do not format fields in your TDP template. That is, keep the default "None" format.</p> <p>Note that this includes "Date" fields, because they are fields formatted as "date".</p>
Limited handling of sign and barcode fields	<p>Results in content not showing properly or missing.</p> <p>Workaround: Add a sign/barcode field to a:</p> <ul style="list-style-type: none"> • Template before generating a TDP. • TDP after its been generated. <p>In both cases, you must open the generated TDP, save it (with the field included), and then relaunch Adobe Acrobat.</p>
Limited widget support for triggers and actions	<p>Results in some widgets not enabled to be selected as trigger or action, i.e., they are not shown in the Trigger Widget or Action Widget combo boxes. See What triggers and actions are supported? for a list of supported combinations of widgets, triggers, and actions.</p> <p>Workaround: Not available.</p>
Slow CAD Validator response time when selecting groups	<p>CAD Validator's response time can be very slow when selecting a group if the number of differences within that group is large, such as in the default group "Ungrouped."</p> <p>Workaround: Split large groups into multiple smaller groups. This can be achieved by applying grouping logic in a CAD Validator customization script.</p>

Issue	Description
Some views do not display/fit properly	<p>Under exceptional conditions (e.g., view properties), some views do not display/fit properly.</p> <p>Workaround: Not available.</p>
View thumbnails do not display the color of cross sections	<p>View thumbnails in view carousels do not show the color of cross sections.</p> <p>Workaround: Not available.</p>
View thumbnails do not display "explosions" properly	<p>In some cases, view thumbnails in view carousels do not display "exploded" views as expected.</p> <p>Workaround: Not available.</p>
Widget is not created when its placeholder name includes a period (".")	<p>When a widget's placeholder name includes a period, e.g., "Text.1", then the widget is not created, i.e., it will be missing from the generated TDP.</p> <p>Workaround: Rename widget placeholders to something that do not include a period, e.g., "Text_1".</p>

All rights reserved by Elysium or the original author of this material. The content may not be edited, reproduced, distributed, transmitted, displayed, published, broadcast, sold or lent without the prior permission of the author.