Altova® EBA add-in for Excel, version 2021r2 Enterprise Edition

User & Reference Manual

Altova® EBA add-in for Excel, version 2021r2 Enterprise Edition
User & Reference Manual

Published: 2015-2021

Table of Contents

1 Introduction

The **Altova® European Banking Authority (EBA) XBRL add-in for Excel** enables your organization to prepare XBRL (eXtensible Business Reporting Language) reports conformant with the EBA (European Banking Authority) XBRL taxonomy, or with related country-specific XBRL taxonomies.

The **Altova® European Banking Authority (EBA) XBRL add-in for Excel** enables preparers of XBRL supervisory reports to do the following:

- Enter XBRL data in Microsoft Excel, using a predefined template spreadsheet which maps to the XBRL taxonomy.
- Validate the report data directly from Excel, to ensure it conforms to the XBRL taxonomy.
- Export report data from Excel to XBRL format.
- Import data from existing XBRL reports into Excel.
- Batch convert XBRL files to Excel (.xlsx) format.

The **Altova® European Banking Authority (EBA) XBRL add-in for Excel** supports the EBA XBRL Taxonomy (starting with version 2.0 up to the most recent version), and country-specific taxonomies. The supported country-specific taxonomies are as follows:

- Banque de France (ACPR) COREP SUBCON - COREP, SubConsolidated (Prudential scope)
- Banque de France (ACPR) LCB FT
- Banco de Portugal (BdP) Reporting Framework
- Banco de Portugal (BdP) FINREP ITS
- Bank of England (BOE) Banking Taxonomy
- Bank of England (BOE) Banking Statistics Taxonomy
- Deutsche Bundesbank (BBK)

- De Nederlandsche Bank Financieel Toetsingskader (DNB FTK)
- National Bank of Belgium (NBB) FINREP (Financial Reporting)
- National Bank of Belgium (NBB) TREP (Trading Reporting)
- Single Resolution Board (SRB) LTD (Liability Data Template)
- Single Resolution Board (SRB) RES
- Single Resolution Board (SRB) SRF

Notes:

- The list of supported XBRL taxonomies is periodically updated to include newer versions, independently of Altova add-in releases. To view or install the latest XBRL taxonomy versions, use the XBRL Taxonomy Manager[37] tool included with the add-in.
- The country-specific XBRL taxonomies are not installed by default when you install the add-in. The same applies to the older versions of the EBA XBRL taxonomy. You can view, install, upgrade, or uninstall such taxonomies on demand, using the XBRL Taxonomy Manager[37].

This documentation should be read in conjunction with the supporting documents included with the EBA XBRL Taxonomy, such as:

- "Description of DPM formal model"
- "EBA Architecture for XBRL representation of DPM"
- "EBA XBRL Filing Rules"

Last updated: 25 February 2021

1.1 System Requirements

To install and run the add-in, the following system requirements apply:

- Windows 7 SP1 with Platform Update, Windows 8, Windows 10, Windows Server 2008 R2 SP1 with Platform Update or newer
- Microsoft Excel 2019, 2016, 2013, 2010
- .NET Framework 4.0 or later
- If you use Excel 2010, Visual Studio 2010 Tools for Office Runtime (https://www.microsoft.com/en-us/download/details.aspx?id=48217) must be installed.

Also, note the following important points:

- The add-in is available for both Microsoft Excel 32-bit and 64-bit. Microsoft Excel 64-bit is recommended if you need to load big taxonomies such as COREP CON, COREP IND, FINREP. Otherwise, you may see "out of memory" errors when attempting to load such taxonomies with Microsoft Excel 32-bit.
- The add-in requires full access to the Excel document in order to create, validate, and export XBRL reports. If your organization enforces Information Rights Management (IRM) using the Azure Information Protection or a similar technology, the latter may restrict access to the Excel document, and thus prevent the add-in from working. For information about how to permit code to run behind documents with restricted permissions, see https://docs.microsoft.com/en-us/visualstudio/vsto/how-to-permit-code-to-run-behind-documents-with-restricted-permissions?view=vs-2019.

1.2 Installation and Licensing

To install the **Altova® European Banking Authority (EBA) XBRL add-in for Excel**, download the executable from the Altova Download Center (https://www.altova.com/download) and run it. Follow the wizard steps to complete the installation. You will need to accept the license agreement and privacy policy in order to proceed with the installation.

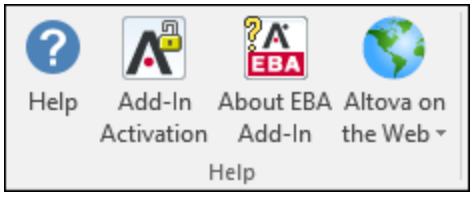
> Make sure to download the executable corresponding to your operating system's and Excel platform (32-bit or 64-bit) The 32-bit executable can be installed on both 32-bit and 64-bit Windows; however, it supports only Excel 32-bit. Note that If you have Excel 32-bit and install the 64-bit version of the add-in, you will still be running the 32-bit version.

After installation, a new tab called **EBA** becomes available in the Excel ribbon.

Licensing

To use **Altova® European Banking Authority (EBA) XBRL add-in for Excel**, you need a valid license key code. To purchase a new key code, or request a free evaluation from the Altova website, take the following steps:

1. In the Excel ribbon, click the **EBA** tab.
2. Click **Add-In Activation**.



A dialog box appears with instructions for getting a new license from Altova, or managing an existing license.

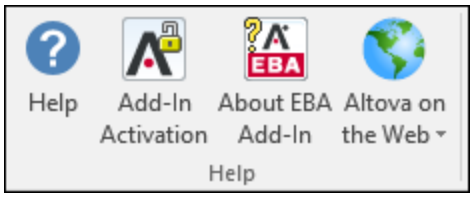
After you purchase a license from Altova, follow the same steps as above to open the activation dialog box and upload the license file.

Alternatively, you can upload purchased licenses to an Altova LicenseServer running on your organization's network. Altova LicenseServer is a free product that helps organizations manage all Altova licenses in a centralized place. For more information about LicenseServer, see https://www.altova.com/licenseserver.

See also License Information 97.

How to view the current version of the add-in

1. In the Excel ribbon, click the **EBA** tab.
2. Click **About EBA Add-In**.


Help
Add-In Activation
About EBA Add-In
Altova on the Web
Help

1.3 Installing XBRL Taxonomies

When you install the add-in for the first time on your computer, only the most recent version of the EBA XBRL Taxonomy is installed by default.

If you need support for other EBA XBRL Taxonomy versions, or country-specific XBRL taxonomies, these have to be installed separately. Specifically, you can install, upgrade, or uninstall XBRL taxonomies on demand, using the XBRL Taxonomy Manager 37 tool included with the add-in.

To run the XBRL Taxonomy Manager, do one of the following:

- In the **EBA** ribbon, click **Manage Taxonomies**.
- Run (double-click) a file with .altova_taxonomies extension downloaded from the Altova website. To open the XBRL Taxonomy Download Center in your browser, select the **EBA** ribbon, and then click **Altova on the Web > Taxonomy Download**.
- From the Windows Control Panel, right-click the **Altova Taxonomy Manager** entry and select **Change** or **Uninstall** from the context menu.

In addition, the check box **Invoke Altova Taxonomy Manager** is available on the last page of the installation wizard, after you complete the installation of Altova® European Banking Authority (EBA) XBRL add-in for Excel.

Note the following:

- Installing or uninstalling a taxonomy from XBRL Taxonomy Manager takes effect for all users accounts on the same computer.
- Installing or uninstalling a taxonomy from XBRL Taxonomy Manager takes effect in all Altova XBRL-enabled applications installed on the same computer.
- If the current taxonomy has dependencies on other taxonomies, the dependent taxonomies are also installed (or uninstalled, as applicable).

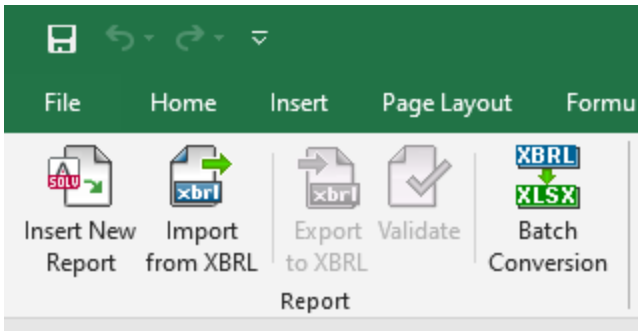
For further information, see Managing XBRL Taxonomies 37.

2 Creating a New Report

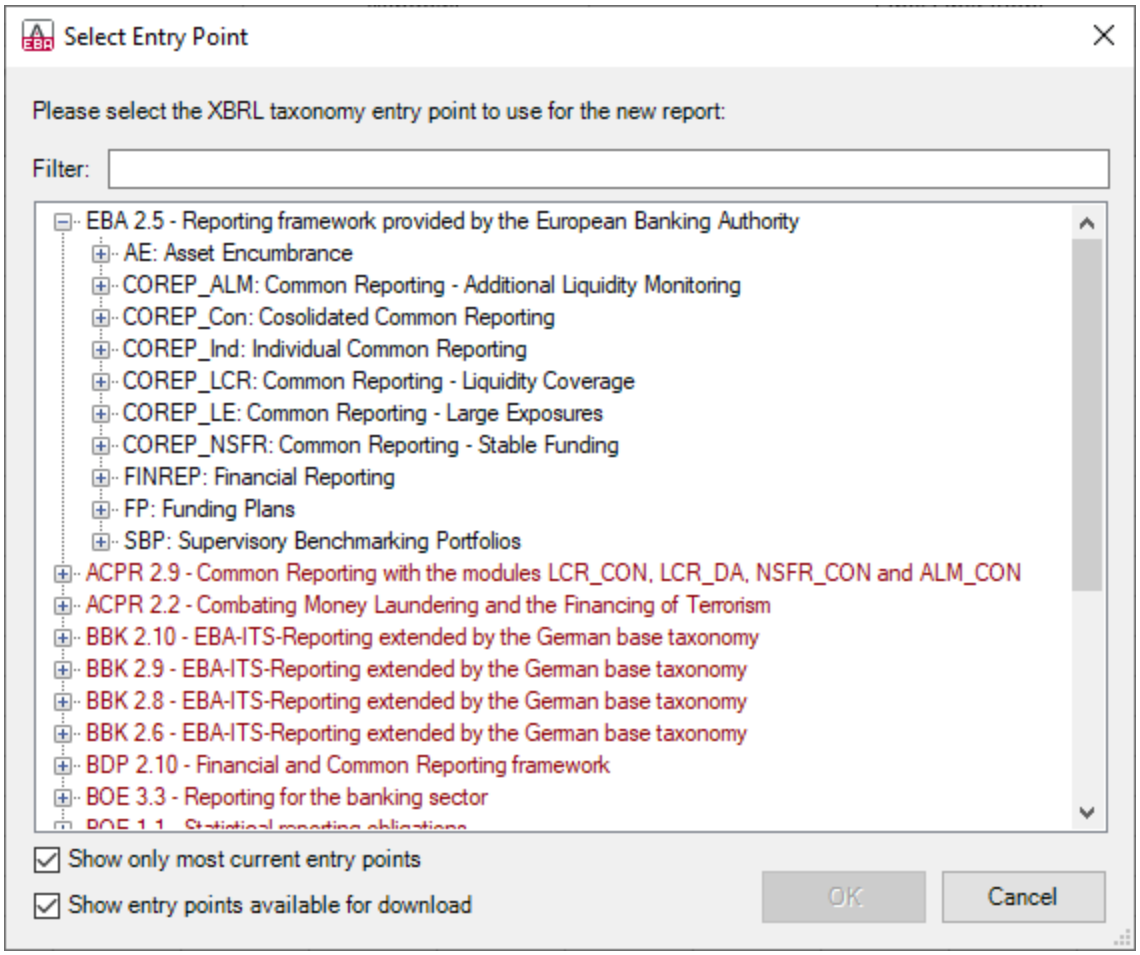
The instructions below show you how to prepare a new XBRL report based on the default EBA (European Banking Authority) taxonomy available in **Altova® European Banking Authority (EBA) XBRL add-in for Excel**. This XBRL taxonomy is installed by default on your computer when you install the add-in. Additional taxonomies can be installed separately, see Installing XBRL Taxonomies.

To create a new report:

1. In the Excel ribbon, click the **EBA** tab.
2. Click **Insert New Report**.



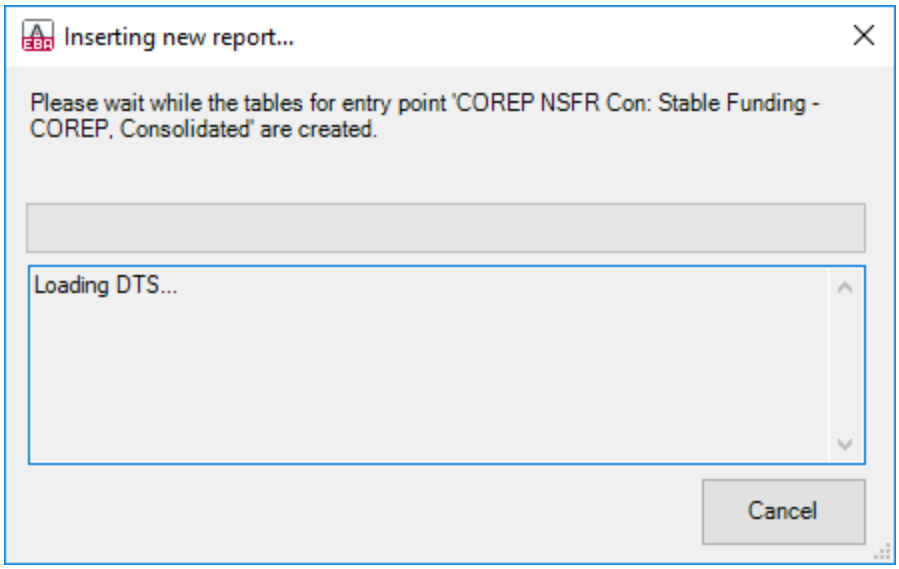
3. Select the taxonomy entry point corresponding to the report that you wish to create. Use the filter at the top of the dialog box to filter entry points by keywords such as "IFRS", "Funding", and so on. Note that, by default, only the most current entry points for the current version of the add-in are shown. To show all the XBRL taxonomies available for download, select the **Show entry points available for download** check box. To show all versions, clear the **Show only most current entry points** check box.



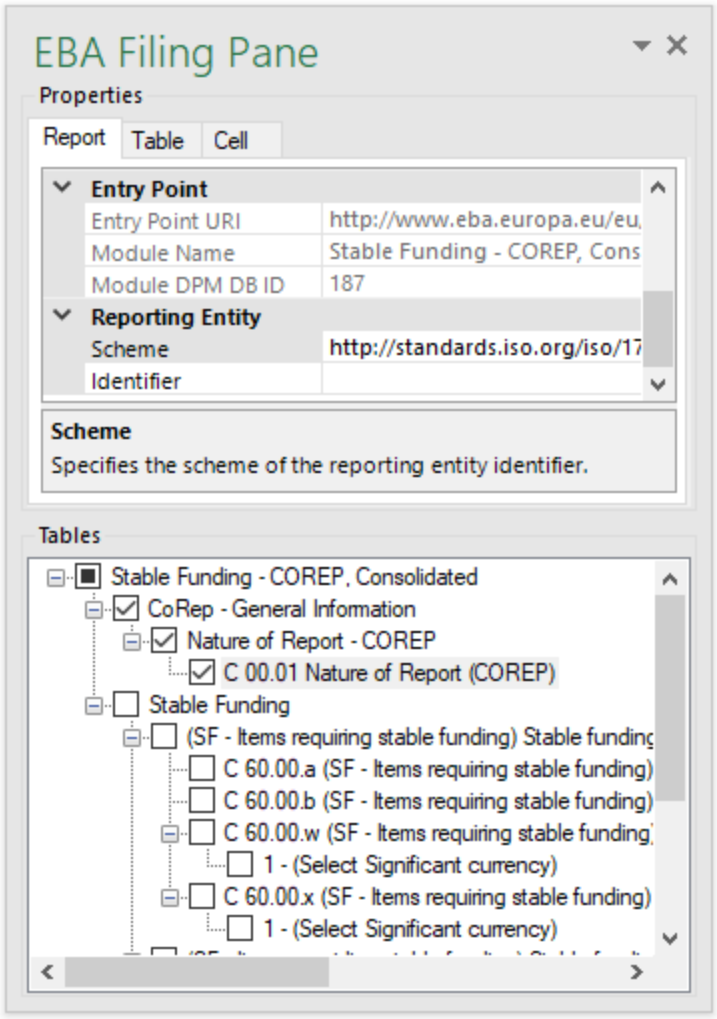
Entry points that are shown in brick red color are not installed. To install the respective XBRL taxonomies, select the entry point, and then click **OK**. This opens the XBRL Taxonomy Manager, where you can complete the installation, see Managing XBRL Taxonomies 37.

Because of memory requirements, some entry points cannot be loaded in the 32-bit version of the add-in, in which case they appear as grayed out on the dialog box above. To make loading of such entry points possible, use Excel 64-bit and install the 64-bit version of the add-in.

4. Be patient while the report tables are loaded into Excel. During this operation, a dialog box informs you of the progress, for example:



Once the report tables have finished loading, notice the **Tables** section in the **EBA Filing Pane**.



5. Select the check boxes next to tables that you want to include in the report. Notice that each included table appears on a new sheet in the Excel book.

You can now start entering data in tables, validate it, and export it to XBRL format. See the following topics for more information:

- Entering Data 17
- Controlling Accuracy of Cells 26
- Validating Data 29
- Exporting Data to XBRL 32

EBA Filing Pane

EBA Filing Pane is the area where you can include or exclude tables from the report, view information about each cell, and view or set various report properties. By default, this pane is visible; you can show or hide it by clicking **Toggle EBA Filing Pane** command in the ribbon. As illustrated above, the **EBA Filing Pane** consists of two main sections: **Properties** and **Tables**.

Properties

The properties displayed in the **EBA Filing Pane** directly affect the content of the XBRL instance file that will be created when you export the XBRL instance. To view what each property does, click it and observe the description displayed in the gray box under the grid. Properties that are grayed out are read-only; otherwise, you can edit a property by typing text or selecting a value as applicable.

The **Scheme** and **Identifier** properties under "Reporting Entity" are typically provided by the relevant competent authorities.

Also note that, even though some property values begin with "http" (for example, **Entry Point URI**, **Scheme**), they do not necessarily point to live web resources and thus should not be considered dead links. To resolve entry point URIs, the add-in uses a catalog mechanism that maps URIs to files on the local system. This is in large part due to the size of the taxonomies and the fact that they contain thousands of files. Accessing the taxonomy files over the Internet would result in extremely slow performance, even if their issuing organizations served them that way.

Properties are grouped into the following three tabs:

- **Report** - This tab displays properties applicable to the entire report (one report corresponds to one Excel workbook).
- **Table** - This tab displays only properties of the currently selected table. A table normally corresponds to a single Excel worksheet. Therefore, whenever you click inside a new Excel sheet, the properties are re-drawn to reflect the new worksheet.
- **Cell** - This tab displays only properties of the currently selected cell. Whenever you click a new cell, the cell properties are re-drawn accordingly.

You can set the accuracy-related properties at report, table, or cell level. For more information, see Controlling Accuracy of Cells 26.

Tables

To include a table in the report at any time, select its corresponding check box in the **EBA Filing Pane**. Each included table appears on a new sheet in the Excel book. To go to a specific sheet, either navigate to it using the standard Excel way, or click the corresponding table in the **EBA Filing Pane**. To remove a particular table from the report, clear the check box next to it. Tables that are not selected will not be included in the report.

Some tables support a Z-Axis (a third dimension). For information about adding a Z-Axis to a table, see Entering Data into Three-Dimensional Tables (Z-Axis) 22.

Each report table displayed in the **EBA Filing Pane** is XBRL-bound, meaning that data you enter directly in the table cells will be reflected in the XBRL instance file when the report is ready, see Exporting Data to XBRL 32. While the report data is work in progress, you can save the Excel workbook and reopen it at any time later, just like a standard Excel workbook.

Any sheets that contain tables are bound to the XBRL taxonomy, so they must not be deleted. It is also not recommended to rename such sheets. If necessary, you can add new sheets to the workbook; however, such sheets would not be bound to the XBRL taxonomy and consequently be ignored when you

generate the XBRL instance file.

2.1 Entering Data

You can populate a report with data either by entering data into cells manually or by pasting values. With some cells, you can select a value from a predefined list (such as countries or currencies). Also, in some report tables, you may need to add new rows or columns. The following is a list of tips and best practices for entering data.

Editable versus non-editable cells

As a general rule, gray cells must not be edited. Only cells that are included in the XBRL-bound area (delimited by the table boundaries) are to be edited. For guidance with respect to the purpose of the cell, and data expected to be entered, consult the cell information (properties) displayed in the **EBA Filing Pane**, in the **Cell** tab.

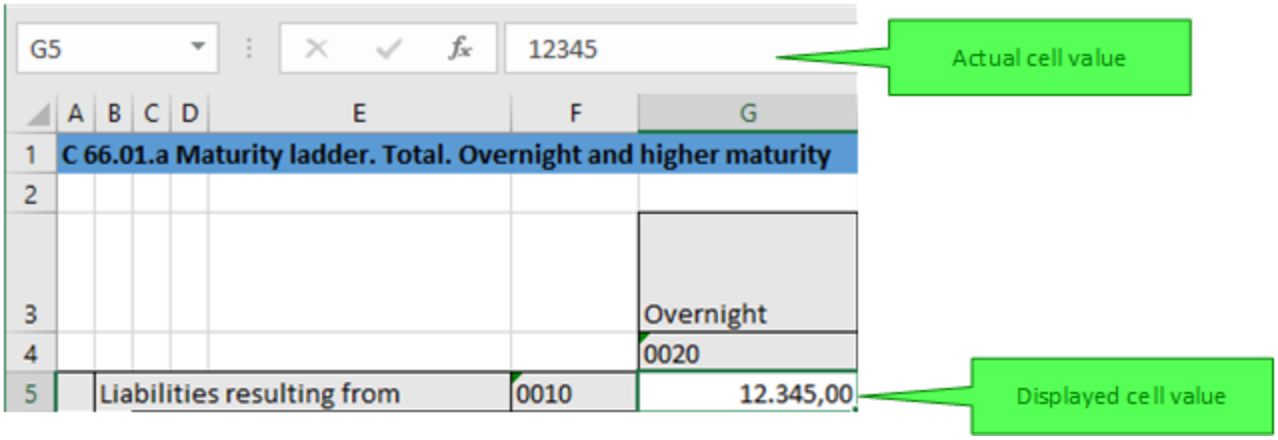
Pasting data

If you paste data from multiple columns, the number of pasted columns should correspond to the number of columns in the predefined sheet. If you accidentally paste a larger number of columns, or if you type text outside the default table, unwanted columns may appear outside the XBRL-bound area. To delete unwanted columns, right-click the cell and select **Delete > Table Columns**. To prevent Excel from adding new columns and rows automatically, go to **File > Options > Proofing > AutoCorrect Options > AutoFormat As You Type > Apply as you work**, and click to clear the **Include new rows and columns in table** check box.

When pasting data, it is recommended to keep only the values (and not the formatting). Namely, select the **Paste Values** option when pasting cells or rows.

Actual versus displayed cell value

While generating the XBRL instance file, the add-in ignores any cell formatting information and exports the *actual* value of the Excel cell. However, bear in mind that, as part of Excel functionality, the *actual* value may be different from the value displayed in the cell, because of the cell formatting information. You can view at any time the actual value (the one that will be written to the XBRL instance) in the formula bar of Excel. Consider this example:

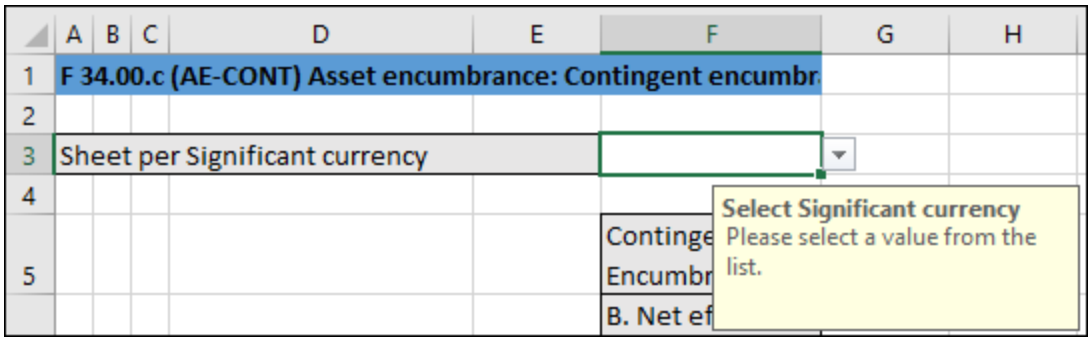


In the example above, the value that will be written to the XBRL instance is *12345*. Note that the number accuracy reported in the XBRL instance file also depends on the value you selected for the "Accuracy"

properties (see Controlling Accuracy of Cells 26).

Enumeration values

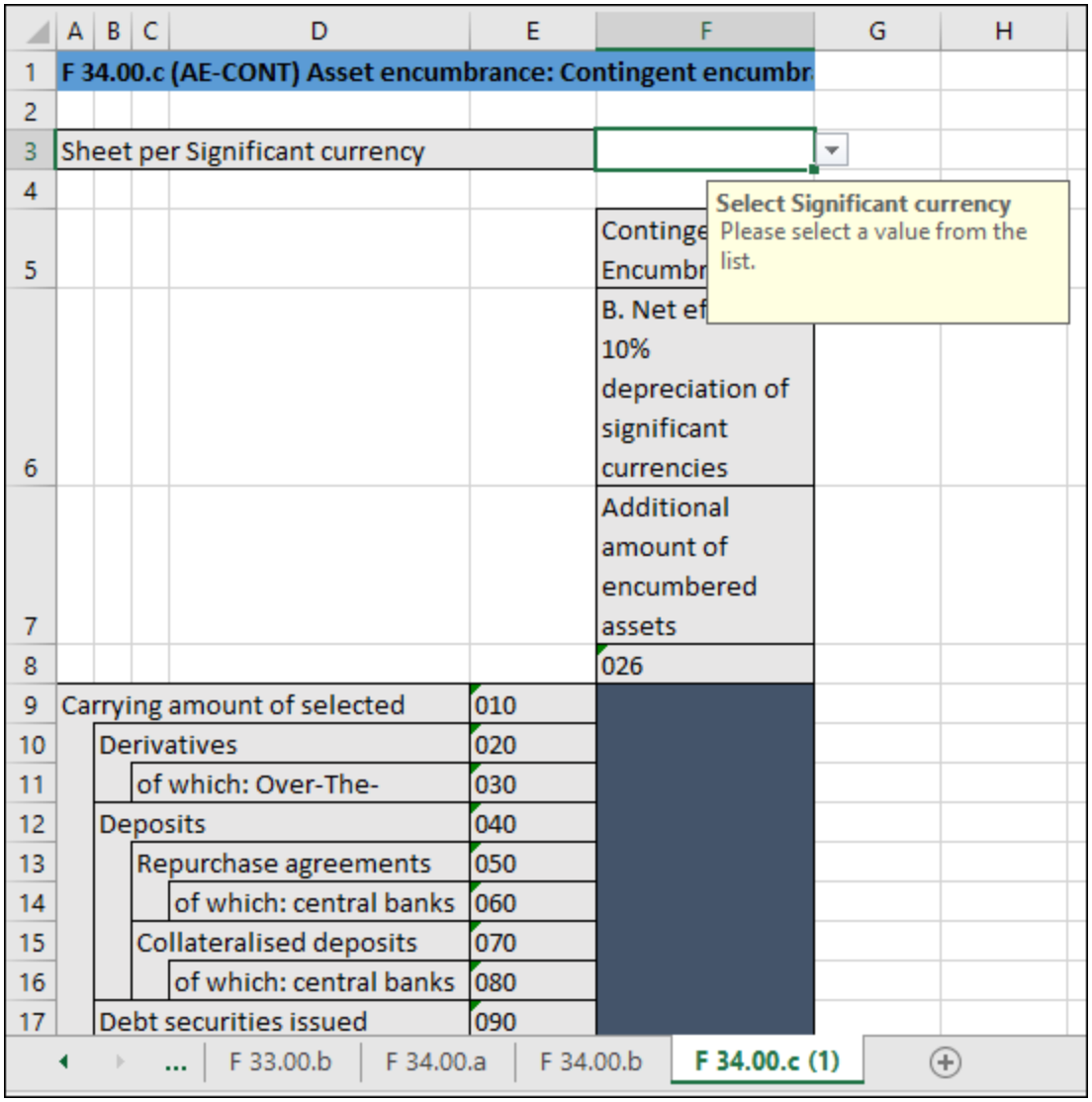
Some cells expect a fixed predetermined value (for example, cells that represent currencies or countries). In this case, the add-in displays a small tooltip when you click the cell. You can pick up the required value from the drop-down list:



To view the full list of all possible values, click the cell and observe the cell properties in the **Cell** tab of the **EBA Filing Pane**.

Conditional cells

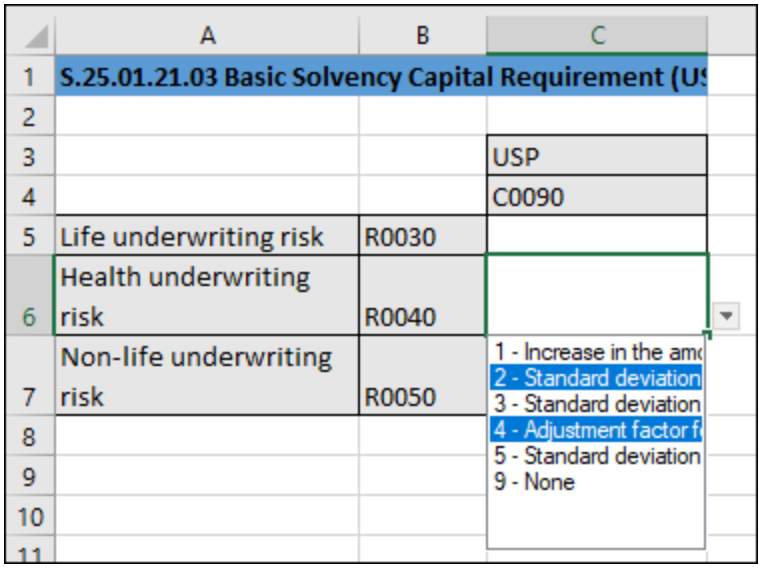
In some tables, you must first fill out a cell value in order to make other cells of the table editable. For example, in the table below, the cell F3 must be first be filled out before all other cells in the same column become editable.



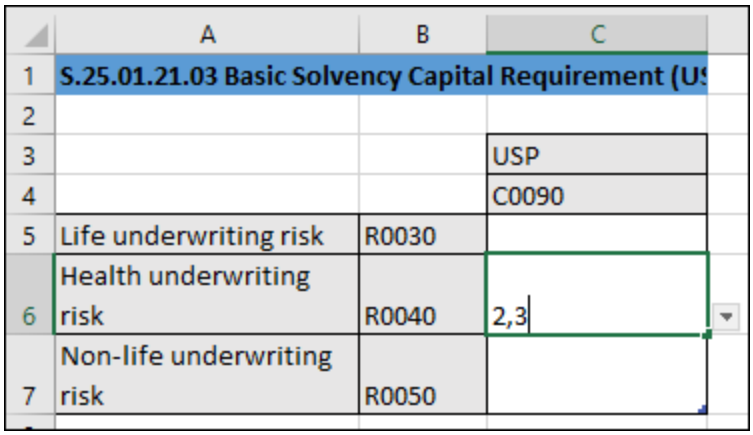
Cells with multiple values

Depending on the XBRL taxonomy, some reports might have facts that represent an arbitrary list of comma-separated multiple values. Consequently, in Excel, the corresponding cell also requires multiple values to be entered in the same cell.

To enter data for such cells, first expand the drop-down list, and then click all items that qualify.



Alternatively, you can type all the numeric values, separated by a comma, as shown below. Remember that you can view all possible values of a cell in the **Cell** tab of the **EBA FilingPane**.



After you exit a multi-valued cell, it is automatically re-drawn to display all selected values in a readable form (even though you may have entered only numbers).

Adding new rows

With some tables, you may need to create new rows. For example, this is the case for table "C 10.02" available through the entry point **EBA 2.6 COREP CON**. You can add new rows either using the standard Excel commands or shortcuts, or by clicking the **Add Row** button in the ribbon. For example, to add a new row to the table "C 10.02" of the entry point mentioned above, do one of the following:

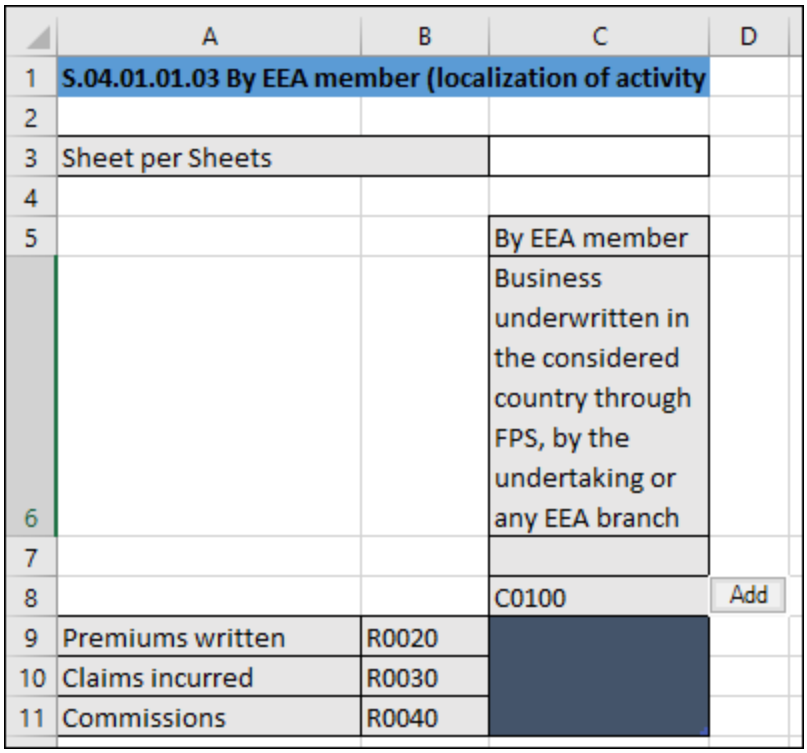
- In the Excel ribbon, click the **EBA** tab, and then click **Add Row | Insert Row Below**. Note that the commands to insert or delete rows are enabled only if the table supports adding new rows.
- Click the rightmost cell of the last row in the table, and press **Tab**.
- Right-click a cell in the empty row, and select **Insert | Table Row Below** from the context menu.

Note: Any newly added rows must be within the XBRL-bound area of the table, clearly delimited by black lines.

Adding new columns

Some tables may need extra columns to be added. In other words, they can grow horizontally. You can add new columns to such tables in one of the following ways:

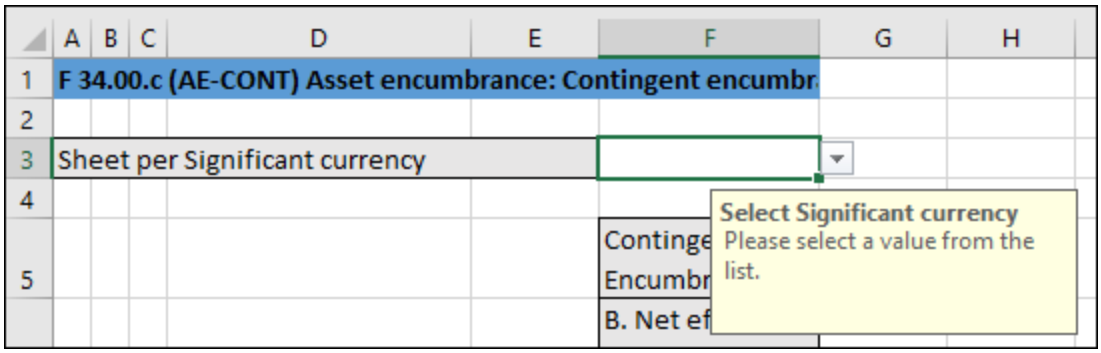
- In the Excel ribbon, click the **EBA** tab, and then click **Add Column**. Note that this command is enabled only if the table supports adding new columns according to the XBRL taxonomy.
- Click the **Add** button that appears next to the rightmost column of a table.



- Right-click a table cell, and then select **Insert | Table Columns to the Right** from the context menu.

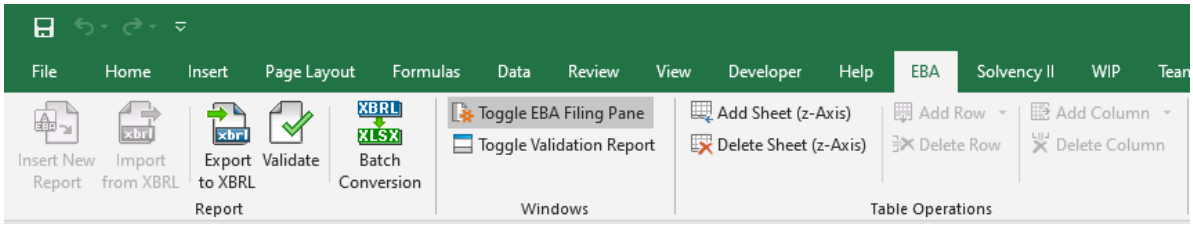
2.2 Entering Data into Three-Dimensional Tables (Z-Axis)

Most of the report tables have only two dimensions: the x-Axis (columns) and the y-Axis (rows). However, there are some tables where you may need to enter data into a third dimension (the z-Axis). An example of such a table is the "F 34.00.c (AE CON)" table available through the entry point **AE CON: Asset Encumbrance, Consolidated**. This table may need an additional sheet for each currency. As shown below, cell F3 is a drop-down list from where you can select a currency.

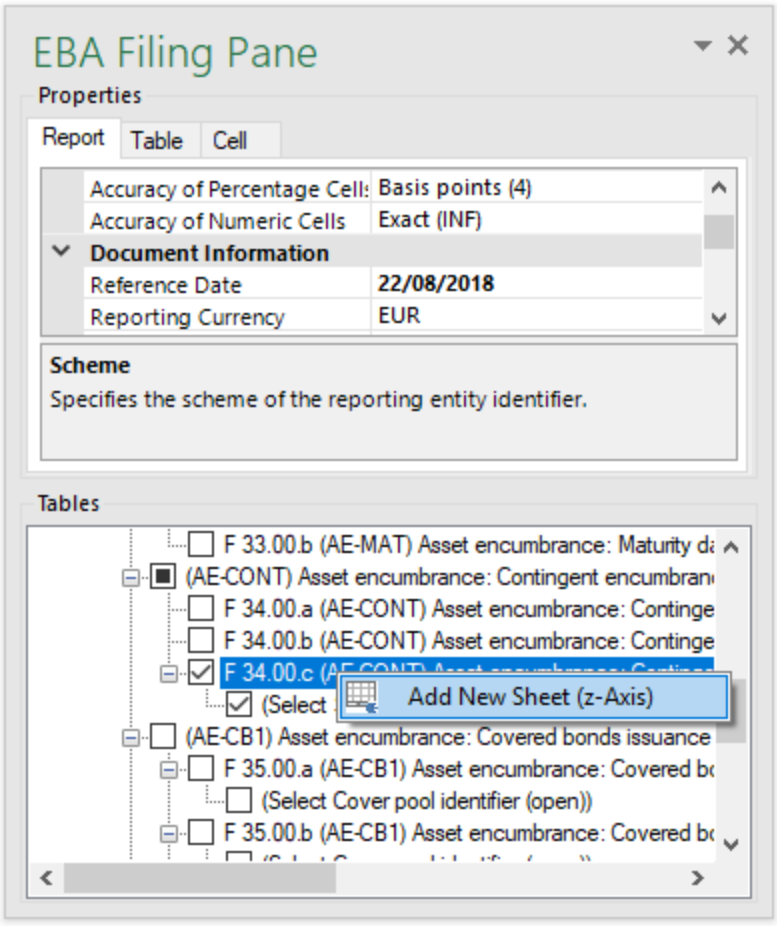


In cases such as the one above, you can add a new sheet along the z-Axis (third dimension) of the table, as follows:

1. In the Excel ribbon, click the **EBA** tab.
2. Click the **Add Sheet (z-Axis)** button. Note that the commands to insert or delete new z-Axis sheets are enabled only if the table supports adding a z-Axis.



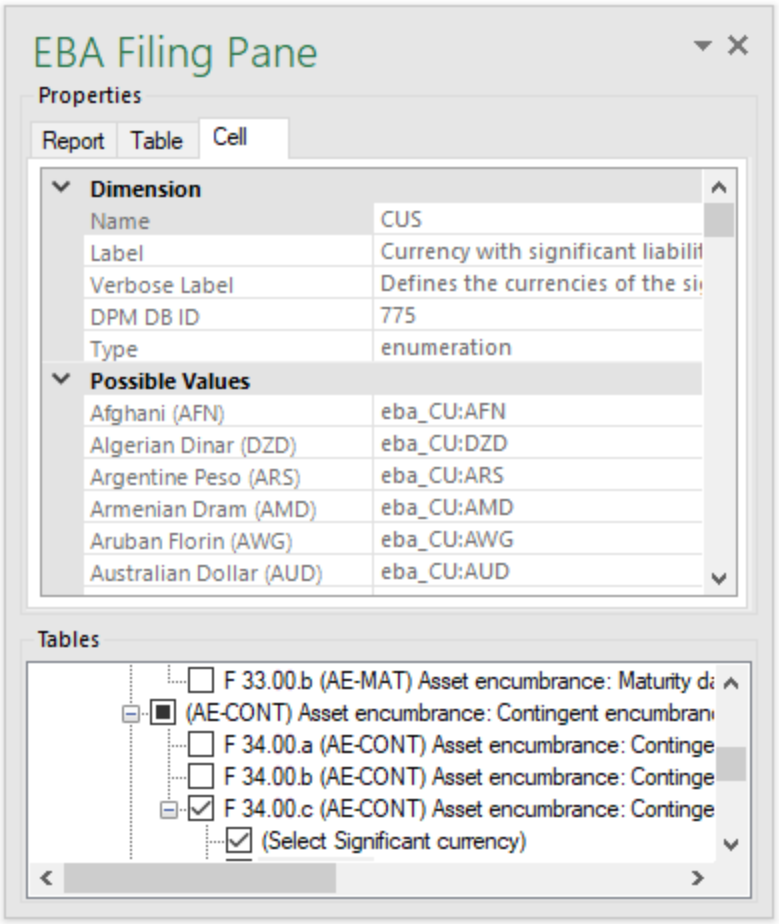
Alternatively, right-click the table in the **EBA Filing Pane** and select **Add New Sheet (z-Axis)** from the context menu, for example:



This creates a new sheet that displays the third dimension of the table (z-Axis). The sheet representing the z-Axis always has an indicative name that resembles the original table. In this example, if you select "Euro" as currency, the new sheet is called "F 34.00c (EUR).

Data from the third dimension (z-Axis) of a table are displayed as new sheets in Excel. Therefore, three-dimensional tables span across more than one sheet. This is an exception to the rule that one Excel sheet corresponds to one table in the XBRL report. In the XBRL instance, data that belongs to the z-Axis will be, however, correctly reported as part of the same table.

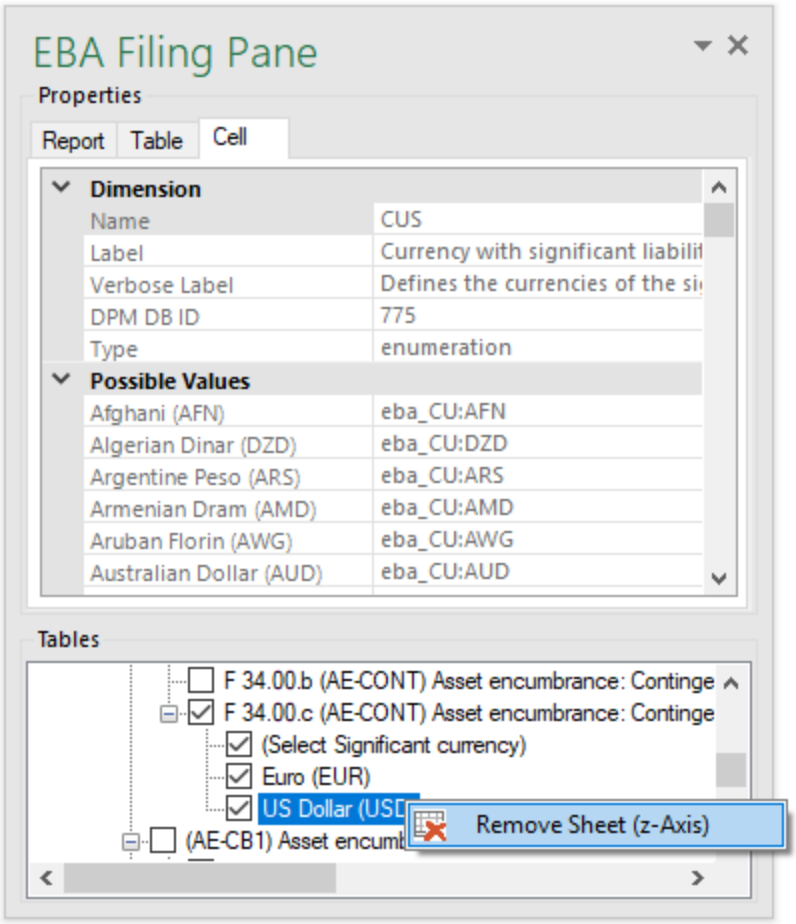
When you click a cell that represents z-Axis, all the possible values for the drop-down list are displayed in the **Cell** tab of the **EBA Filing Pane**, for example:



Deleting z-Axis sheets

You can delete sheets that contain data from the third dimension (z-Axis) in more than one way.

1. Select the required sheet (or click the corresponding entry in the "Tables" section of **EBA Filing Pane**).
2. Do one of the following:
   - In the **EBA** tab, click **Remove Sheet (z-Axis)**, or
   - Right-click the table in the **EBA Filing Pane** and select **Remove Sheet (z-Axis)** from the context menu.

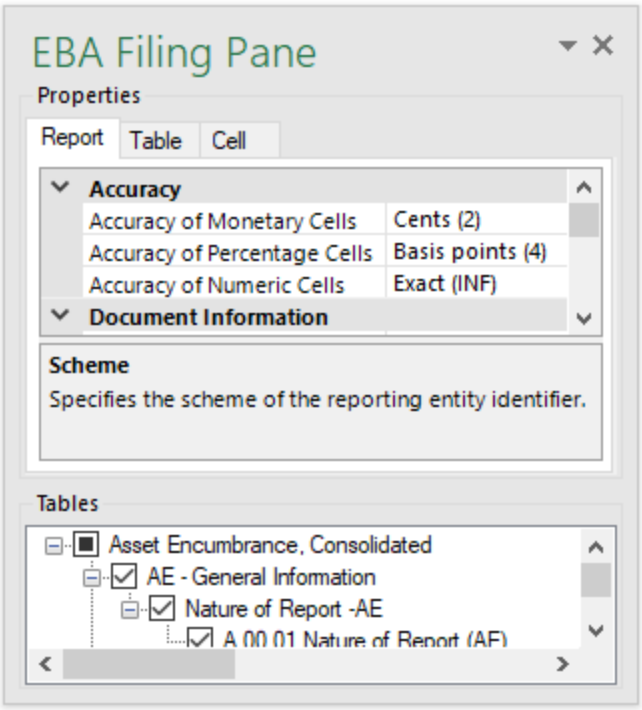

EBA Filing Pane
Properties
Report
Table
Cell
Dimension
Name
CUS
Label
Currency with significant liabili
Verbose Label
Defines the currencies of the si
DPM DB ID
775
Type
enumeration
Possible Values
Afghani (AFN)
eba_CU:AFN
Algerian Dinar (DZD)
eba_CU:DZD
Argentine Peso (ARS)
eba_CU:ARS
Armenian Dram (AMD)
eba_CU:AMD
Aruban Florin (AWG)
eba_CU:AWG
Australian Dollar (AUD)
eba_CU:AUD
Tables
F 34.00.b (AE-CONT) Asset encumbrance: Continge
F 34.00.c (AE-CONT) Asset encumbrance: Continge
(Select Significant currency)
Euro (EUR)
US Dollar (USD
Remove Sheet (z-Axis)
(AE-CB1) Asset encumb

2.3 Controlling Accuracy of Cells

The accuracy of monetary and other numeric values in the XBRL report can be controlled by setting the following report properties:

1. Accuracy of monetary cells
2. Accuracy of numeric cells
3. Accuracy of percentage cells

These properties are available in the **EBA Filing Pane**, in the **Properties** group.

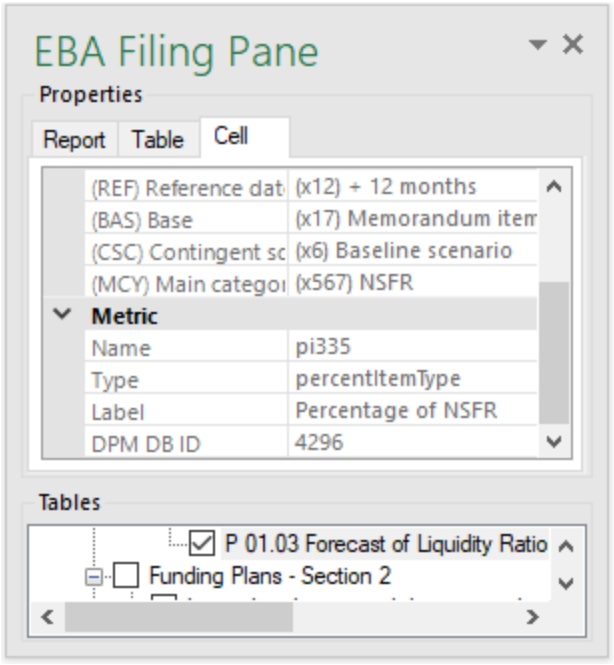


The property **Accuracy of Monetary Cells** applies to numeric cells in the report that represent a monetary value. For example, if you select entry point **FP IND: Funding Plans, Individual**, Table "P 01.03", this property affects all cells of type `monetaryItemType`.

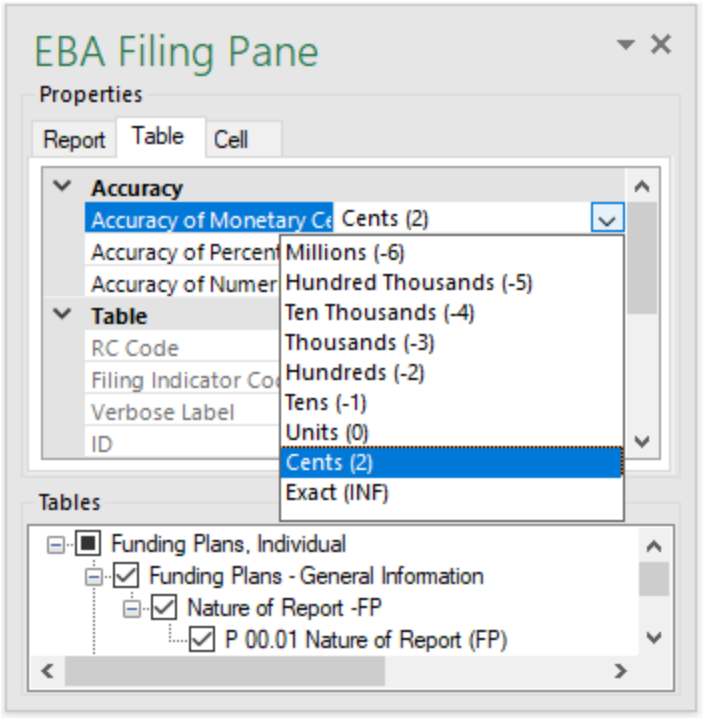
The property **Accuracy of Numeric Cells** applies to numeric values that have no unit (excluding percentage values, which have their own data type). For example, if you select entry point **SPB CON**, Table "C 101.00", this property affects the column "Maturity (160)".

The property **Accuracy of Percentage Cells** applies to values that represent a percentage. For example, if you select entry point **FP IND: Funding Plans, Individual**, Table "P 01.03", this property affects all cells of type `percentItemType`.

To view the type of a cell, first select a cell, and then click the **Cell** tab in the **EBA Filing Pane** and observe the **Type** property, for example:



The accuracy you select from the **EBA Filing Pane** controls the accuracy that will be written for this fact in the XBRL instance file. More specifically, the "Accuracy" properties are bound to the **decimals** attribute in the XBRL instance file. For example, in the image below, the accuracy value indicated in the brackets corresponds to the value of the **decimals** attribute in the XBRL instance file.



By default, accuracy is set as follows:

- The **Accuracy of Monetary Cells** is set to **Cents (2)**, which sets the value of the **decimals** attribute in the XBRL instance to "2"
- The **Accuracy of Percentage Cells** is set to **Basis Points (4)**, which sets the value of the **decimals** attribute in the XBRL instance to "4".
- The **Accuracy of Numeric Cells** is set to **Exact (INF)**, which sets the value of the **decimals** attribute in the XBRL instance to "INF".

You can set the accuracy-related properties at report, table, or cell level (see the corresponding tabs in the image above). If you set accuracy at multiple levels, keep in mind that the more specific property always overrides the more generic one. For example, the accuracy set at cell level takes priority over the one set at table level. Likewise, the accuracy set at table level takes priority over the one set at report level.

For monetary and numeric cells, the accuracy value can be either positive or negative.

A positive value *N* specifies the accuracy of up to *N* digits to the right of the decimal place. For example, the value 2 specifies the accuracy to be in cents, while the value 3 specifies the accuracy to be up to mills.

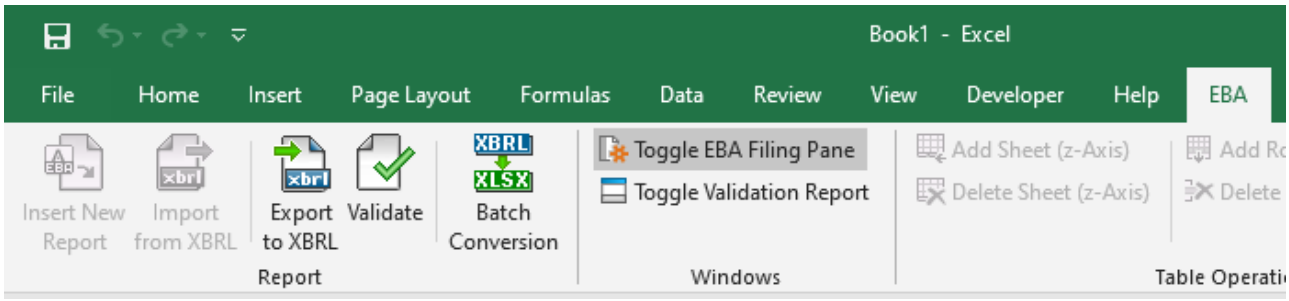
A negative value *N* specifies the accuracy of up to *N* digits to the left of the decimal place. For example, the value -3 specifies the accuracy to be up to thousands, while the value -6 specifies the accuracy to be up to millions.

3 Common Tasks

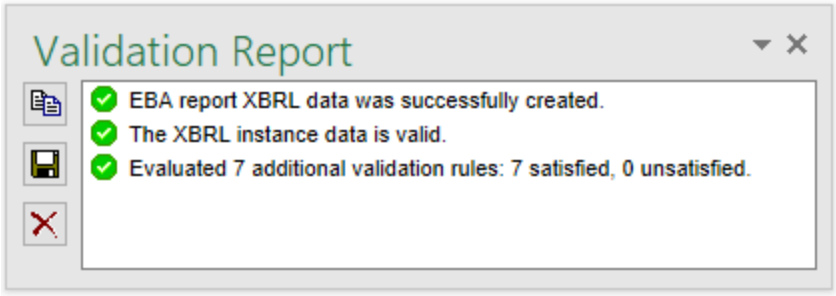
3.1 Validating Data

Validation ensures that the XBRL data you are filing conforms to the XBRL specification. The report data should be validated before you export it to XBRL. You might also want to validate data progressively, after each action that could potentially render it invalid (for example, after pasting new rows into the spreadsheet).

To validate data, click the **Validate** button in the **EBA** tab of the Excel ribbon.



Be patient while **Altova® European Banking Authority (EBA) XBRL add-in for Excel** performs the validation process. To validate XBRL data, the add-in creates a temporary in-memory XBRL instance. When validation of the in-memory instance completes, a validation report similar to the one below is displayed.



The validation result can be any of the following:

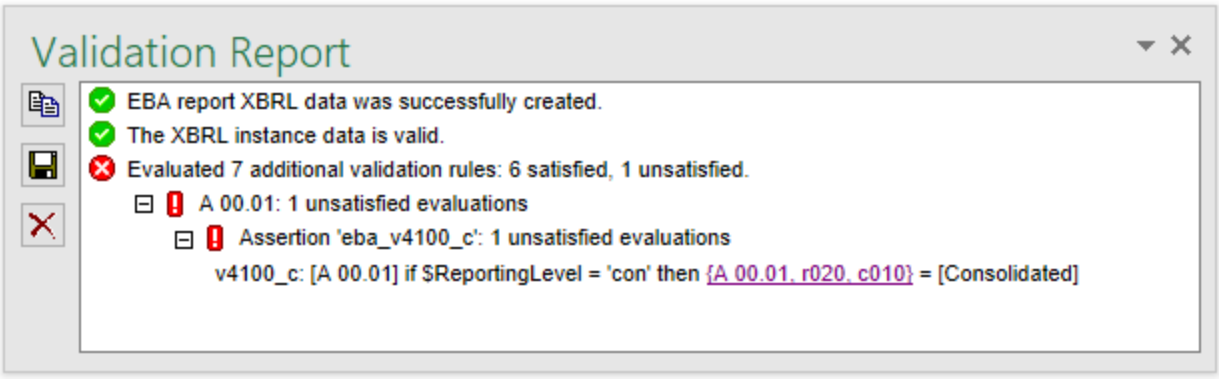
Message type	Meaning
✔	The instance data is valid.
⚠	The instance data is valid, but has inconsistencies or warnings.
✖	The instance data is not valid.

The Validation Report dialog box may additionally display any of the following message types: information messages, warnings, and errors.

Message type	Meaning
	Denotes an information message. Information messages do not make the XBRL instance invalid.
	Denotes a warning message, or an inconsistency. Warnings and inconsistencies do not make the XBRL instance invalid.
	Denotes an error. If there are validation errors, the XBRL instance is not valid, and you will need to edit the report data so as to resolve each error before proceeding with the export to XBRL. Note: During validation, the add-in checks XBRL formula assertions and reports them as errors. If you are using the Altova RaptorXML+XBRL Server for validation (https://www.altova.com/raptorxml), XBRL formula assertions may be optionally configured not to be reported as errors.

Note: By default, the add-in treats invalid cell values as errors. If necessary, you can configure the add-in to treat invalid cell values as warnings instead, see Settings 58.

When a report fails successful validation, the Validation Report window may display links to the cell where the error originates. To quickly find a cell where the error originates, click the underlined text and the cursor will be positioned automatically on the required cell. Note that there are cases where multiple cells are involved in a single validation check; in such cases, clicking on the error link will select just one of the affected cells.



To copy the contents of the validation report to clipboard:

- Click  **Copy**, and then paste into a target file (for example, an email). Alternatively, right-click inside the Validation Report window and select **Copy All Messages** from the context menu.

To save the validation report as text or HTML:

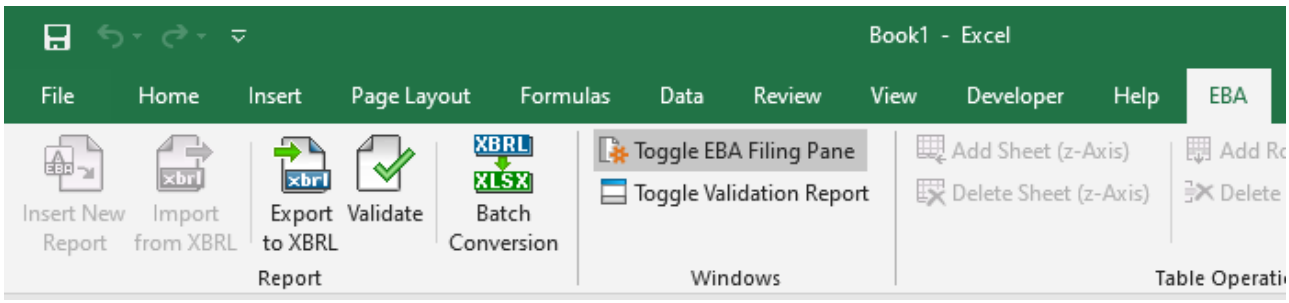
- Click  **Save**. Alternatively, right-click inside the Validation Report window and select **Save Validation Report** from the context menu.

To clear the validation report:

- Click  **Clear**. Alternatively, right-click inside the Validation Report window and select **Clear** from the context menu.

3.2 Exporting Data to XBRL

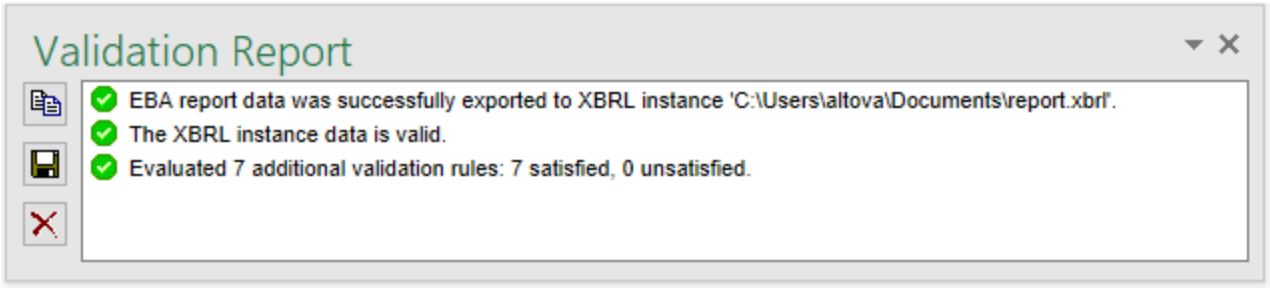
Once your report is ready and valid (see Validating Data 29), you can generate the XBRL instance file. To do this, click the **EBA** tab, and then click **Export to XBRL**.



By default, instance files are saved as files with .xbrl extension. If you need the exported file to have another extension (for example, .xml), type the file extension in the Export dialog box.

While the XBRL instance is being created, a dialog box which informs you about the progress may be displayed for a short time.

During the export operation, data is automatically validated. Any errors, inconsistencies and warnings are reported on the screen after the export finishes.



Note: Cell values that are not valid (that is, cells that don't conform to the data type of the underlying XBRL concept) prevent the report from being exported.

For tips on how to avoid data formatting errors, see Entering Data 17. Note, however, that not all XBRL validation errors might be related to incorrect formatting. Some errors might occur because entered data does not meet the XBRL validation rules applicable to the report you are filing.

3.3 Importing Data from XBRL

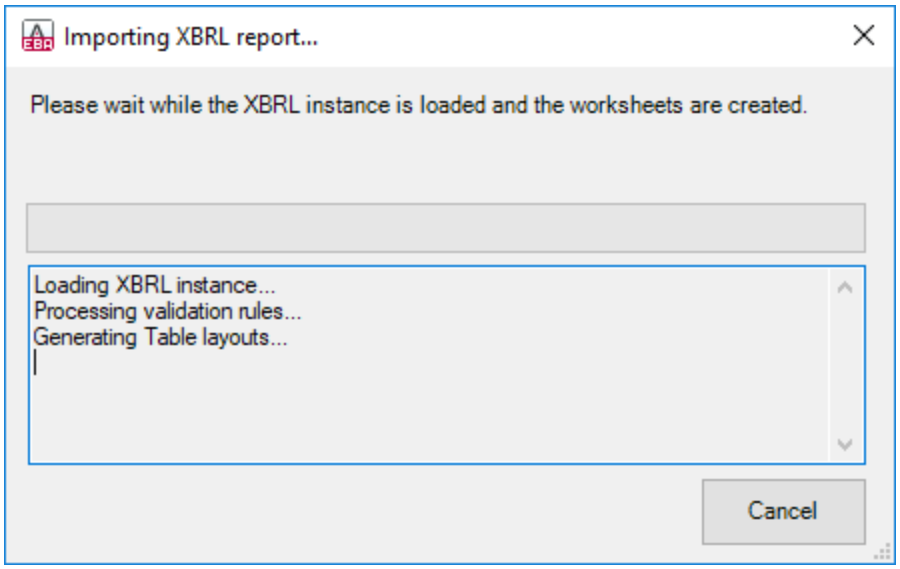
You can import data from existing instances of XBRL reports into Excel (typically, files with .xbrl extension). For the import to be successful, the imported instances must be valid XBRL reports. They may be either reports you have previously generated using the **Altova® European Banking Authority (EBA) XBRL add-in for Excel**, or reports that you received from other parties.

To import an existing XBRL instance file into Excel:

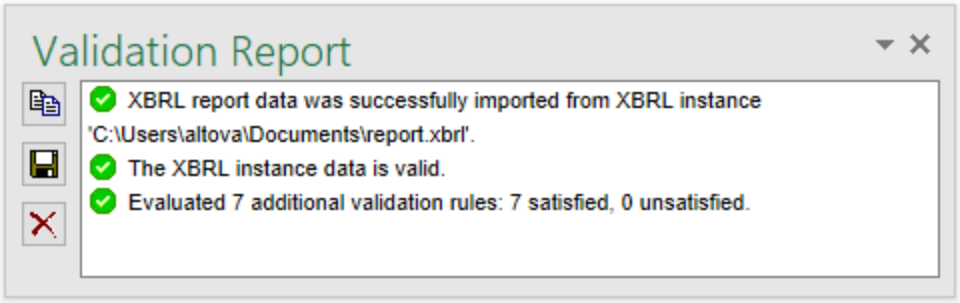
1. In the Excel ribbon, click the **EBA** tab.
2. Click **Import from XBRL**, and browse for the XBRL instance file.

Note: If a report is already open in Excel, the **Import from XBRL** button is disabled. To enable the command, save and close the current report (workbook) and create a new workbook.

During the import operation, a dialog box informs you about the progress:



While the report data is loaded into Excel, it is automatically validated. A dialog box notifies you about potential warnings, inconsistencies, or errors (see also Validating Data 29).



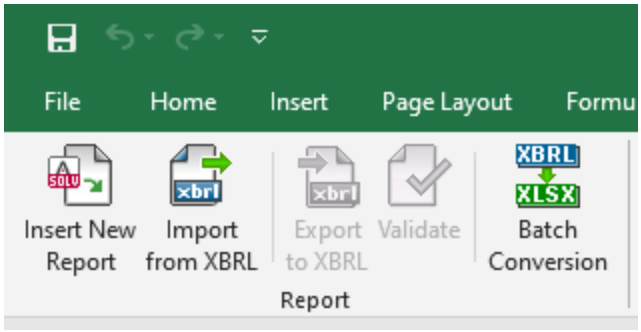
Note: During import, the add-in validates XBRL formula assertions. A report will be imported even if it contains unsatisfied assertions.

3.4 Batch Conversion from XBRL to Excel

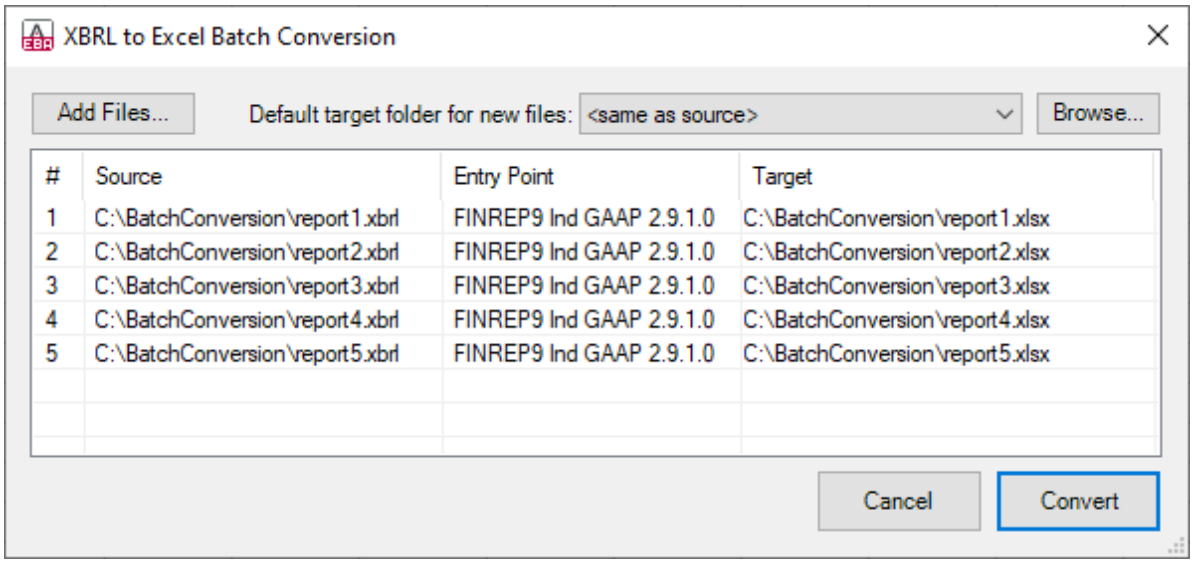
The **Batch Conversion** command in the Excel ribbon enables you to convert multiple XBRL instance files to Excel format. The result is the same as if you imported [33] multiple XBRL instance files and then saved them to Excel format—while having the advantage that conversion takes place in batch.

In order to perform a batch conversion, you must first add all the required files to a batch, as follows:

1. In the Excel ribbon, click the **EBA** tab.



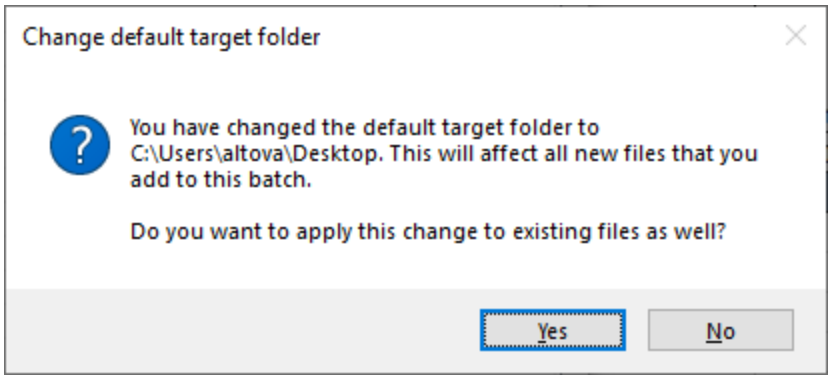
2. Click the **Batch Conversion** button.
3. Click **Add Files**. (Alternatively, right-click the grid, and select **Add Files** from the context menu.)



From the conversion dialog box, you can perform the following additional tasks:

1. To add files from additional source locations to the same batch, click **Add Files**.
2. Whenever you add new files to the batch, their default target folder is the same as the source folder. If you want to assign a specific target folder to all new files by default, select it from the **Default target folder for new files** list. To add new entries to the list, click **Browse** and choose a folder.

Note: By default, the option **Default target folder for new files** affects new files that you add to the batch. However, if you change this option and files already exist in the batch (on the grid), a dialog box like the one below appears. Click **Yes** if the target folder of existing files should be changed as well.



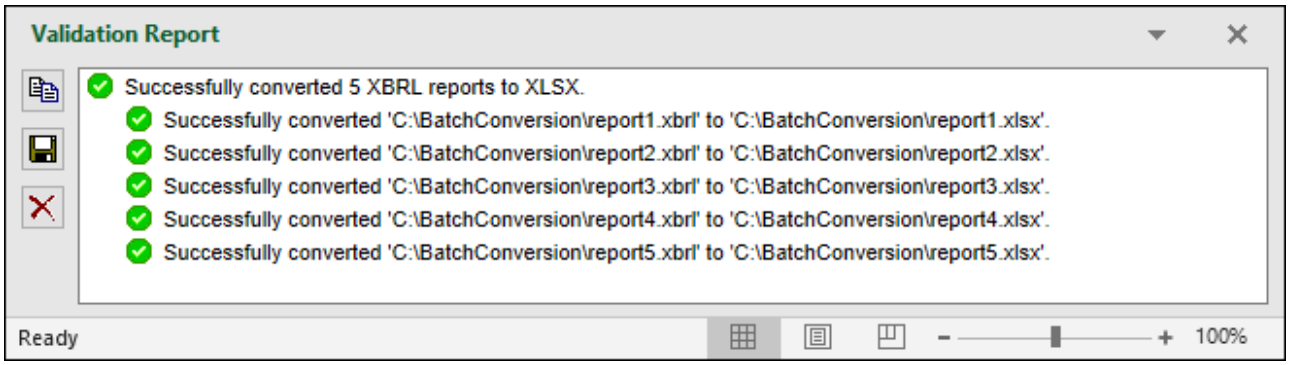
3. You can choose to save all converted files to the same target folder, or set a different target folder for each file. To change the target folder of specific files, first select the files on the grid, and then right-click the grid and select **Set target folder** from the context menu.
4. You can rename the target files. First select the files on the grid, and then right-click the grid and select **Rename** from the context menu (or press **F2**). You can change both the file name and the file path. However, if you change the path, make sure that it exists.
5. To remove files from the batch, first select them, and then right-click the grid and select **Remove** from the context menu (or press the **Del** key).

Tip: To select multiple files from the grid, the standard Windows key combinations apply, for example:

- While holding the **Ctrl** key pressed, click to select the files of interest.
- Click an empty area in the dialog box and then drag the cursor over the files to be selected (rectangular selection)
- Press **Ctrl+A** to select all files in the grid.

Once the batch is ready, click **Convert** to process all files in it. If any files with the same name already exist at the target folder, a message box appears asking your confirmation to overwrite them.

While a batch conversion is running, a dialog box appears that informs you about the progress. The outcome of the conversion operation is reported in the Validation Report window, for example:



When performing batch conversion, the same validation takes place as when importing a single XBRL instance file. If validation is successful, a message similar to "Successfully converted file.xlsx" is reported in the Validation Report for each file. If there are validation errors or warnings, they are reported in the Validation Report as well.

4 Managing XBRL Taxonomies

XBRL Taxonomy Manager is a tool that provides a centralized way to install and manage XBRL taxonomies for use across all Altova XBRL-enabled applications, including Altova® European Banking Authority (EBA) XBRL add-in for Excel. On Windows, XBRL Taxonomy Manager has a graphical user interface and is also available at the command line. On Linux and Mac*, the tool is available at the command line only.

* The Linux and macOS operating systems are applicable only if you are running XBRL Taxonomy Manager on those operating systems in conjunction with Altova cross-platform server applications such as MapForce Server, StyleVision Server, or RaptorXML+XBRL Server.

XBRL Taxonomy Manager provides the following features:

- View XBRL taxonomies installed on your computer, and check whether new versions are available for download.
- Download newer versions of XBRL taxonomies independently of the Altova product release cycle. All taxonomies are maintained by Altova on an online-based storage accessible to XBRL Taxonomy Manager, and you can download them as soon as they become available.
- Install or uninstall any of the multiple versions of a given taxonomy (or all versions if necessary).
- A single XBRL taxonomy represents a "package" but it may have dependencies on other taxonomies. Whenever you choose to install or uninstall a particular taxonomy, any dependent taxonomies are detected and also installed or removed automatically. The graphical user interface (or the command line if applicable) informs you when dependencies are being added or removed.
- XBRL taxonomies maintained through XBRL Taxonomy Manager benefit from the XML catalog mechanism that enables URI references in instance or schema documents to be resolved from local files, as opposed to being retrieved from the Internet. This is extremely important in the case of big XBRL taxonomies where schema resolution from remote URIs is not practical or even recommended, mainly for performance reasons.

XBRL Taxonomy Manager provides a way to administer any of the XBRL taxonomies required for use in any one of the Altova XBRL-enabled applications 37. These include the European Banking Authority Reporting Framework taxonomies, US-GAAP Financial Reporting taxonomies, and various other country- or domain-specific XBRL taxonomies. To view the full list, either run XBRL Taxonomy Manager or run the `list` command at the command line.

Altova XBRL-enabled applications

The following Altova applications are XBRL-enabled and thus benefit from the features provided by XBRL Taxonomy Manager:

- Altova XBRL Add-ins for Excel (EBA, Solvency II)
- MapForce Enterprise Edition
- MapForce Server
- MapForce Server Advanced Edition
- RaptorXML+XBRL Server
- StyleVision Server
- StyleVision Enterprise Edition
- XMLSpy Enterprise Edition

Changes in XBRL taxonomies using XBRL Taxonomy Manager take effect for all the applications listed above if they are installed on the same computer.

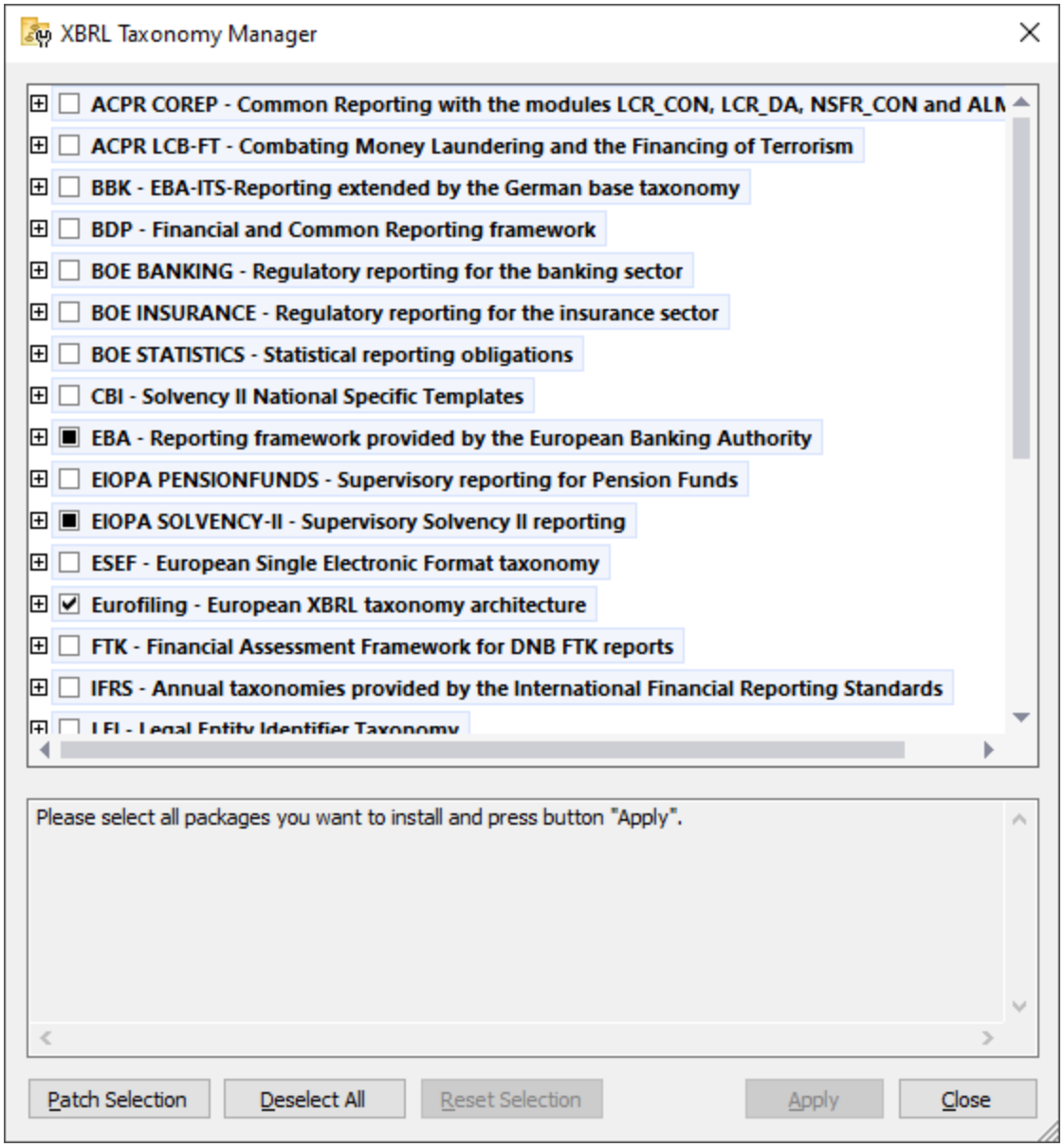
Installation

XBRL Taxonomy Manager is installed automatically whenever you install any Altova XBRL-enabled application or the **Altova Mission Kit Enterprise Edition**. Likewise, it is removed automatically when you uninstall the last Altova XBRL-enabled application from your computer or the **Altova Mission Kit Enterprise Edition**.

How it works

Altova maintains an online taxonomy storage where all XBRL taxonomies used in Altova products are stored. This taxonomy storage is updated on a periodical basis, for example, shortly after new versions of relevant taxonomies are released by their issuing organizations.

Whenever you run XBRL Taxonomy Manager at the graphical user interface, information about the latest available taxonomies is displayed in a dialog box, where you can view, install, upgrade or uninstall them. You can also perform the same actions at the command line interface.



You may also install taxonomies by running .altova_taxonomies files downloaded from the Altova website (https://www.altova.com/taxonomy-manager).

Regardless of the manner in which taxonomies were installed, all information about installed taxonomies is tracked in a centralized location on your computer, also known as the local cache directory. The local cache directory contains information about Altova packages (except for the actual taxonomy files, which are installed on demand). The local cache directory is at the following path:

Linux	`/var/opt/Altova/pkgs`
macOS	`/var/Altova/pkgs`

Windows	`C:\ProgramData\Altova\pkgs`

The local cache directory gets updated automatically from time to time, so as to propagate the latest state of the online storage to the local computer. More specifically, the cache is updated as follows:

- When you run [41] the XBRL Taxonomy Manager.
- When you run Altova® European Banking Authority (EBA) XBRL add-in for Excel for the first time in the same calendar day.
- If Altova® European Banking Authority (EBA) XBRL add-in for Excel is already running, the cache directory gets updated every 24 hours.
- You can also update the local cache from the online storage on demand, by running the update command at the command line interface.

As you install or uninstall taxonomies, the local cache directory gets automatically updated with information about the available and installed taxonomies, as well as the taxonomy files themselves.

The local cache directory is maintained automatically based on the taxonomies you install or uninstall; it should not be altered or deleted manually. If you ever need to reset XBRL Taxonomy Manager to the original "pristine" state, run the `reset` command of the command line interface, and then run the `initialize` command. (Alternatively, run the `reset` command with the `-i` option.)

4.1 Run Taxonomy Manager

To run the XBRL Taxonomy Manager, do one of the following:

- In the **EBA** ribbon, click **Manage Taxonomies**.
- Run (double-click) a file with .altova_taxonomies extension downloaded from the Altova website. To open the XBRL Taxonomy Download Center in your browser, select the **EBA** ribbon, and then click **Altova on the Web > Taxonomy Download**.
- From the Windows Control Panel, right-click the **Altova Taxonomy Manager** entry and select **Change** or **Uninstall** from the context menu.

In addition, the check box **Invoke Altova Taxonomy Manager** is available on the last page of the installation wizard, after you complete the installation of Altova® European Banking Authority (EBA) XBRL add-in for Excel.

Command line interface

To run XBRL Taxonomy Manager from a command line interface:

1. Open a command prompt window and change directory to **C:\ProgramData\Altova\SharedBetweenVersions**.
2. To display help at the command line, run:

```
TaxonomyManager.exe --help
```

4.2 Install a Taxonomy

To install a taxonomy:

1. Run [41] XBRL Taxonomy Manager.
2. Select the check box next to the taxonomies or taxonomy versions you want to install, and click **Apply**.

Alternatively, if you have downloaded a file with **.altova_taxonomies** extension from the Altova website (https://www.altova.com/taxonomy-manager), double-click the **.altova_taxonomies** file to run it. XBRL Taxonomy Manager opens when you run the **.altova_taxonomies** file.

Note the following:

- Installing or uninstalling a taxonomy from XBRL Taxonomy Manager takes effect for all users accounts on the same computer.
- Installing or uninstalling a taxonomy from XBRL Taxonomy Manager takes effect in all Altova XBRL-enabled applications installed on the same computer.
- If the current taxonomy has dependencies on other taxonomies, the dependent taxonomies are also installed (or uninstalled, as applicable).

Command line interface

To install a taxonomy, run:

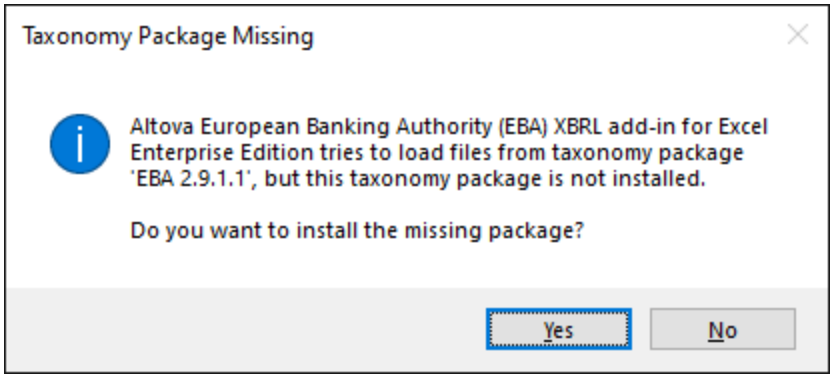
```
TaxonomyManager.exe install FILTER...
```

Where `FILTER` means one of the following:

1. A taxonomy identifier in the format `<name>-<version>`, for example: **eba-2.10**, **us-gaap-2020.0**. To view all the available taxonomy identifiers and versions, run the `list` command.
2. An **.altova_taxonomies** file downloaded from the Altova website.

Installing taxonomies on demand

Whenever XBRL taxonomies required by the add-in are missing from your computer, you may be prompted to install taxonomies on demand. For example, a dialog box such as the one below may appear when you run an action that requires loading XBRL taxonomies:



When you click **Yes**, the missing taxonomies will be installed and the local cache directory will be updated to keep track of this information. You can always view all of the previously installed taxonomies by clicking the **Manage Taxonomies** button in the **EBA** ribbon.

If you select **No**, the missing taxonomies will not be installed, but you will not be able to use the add-in in this case.

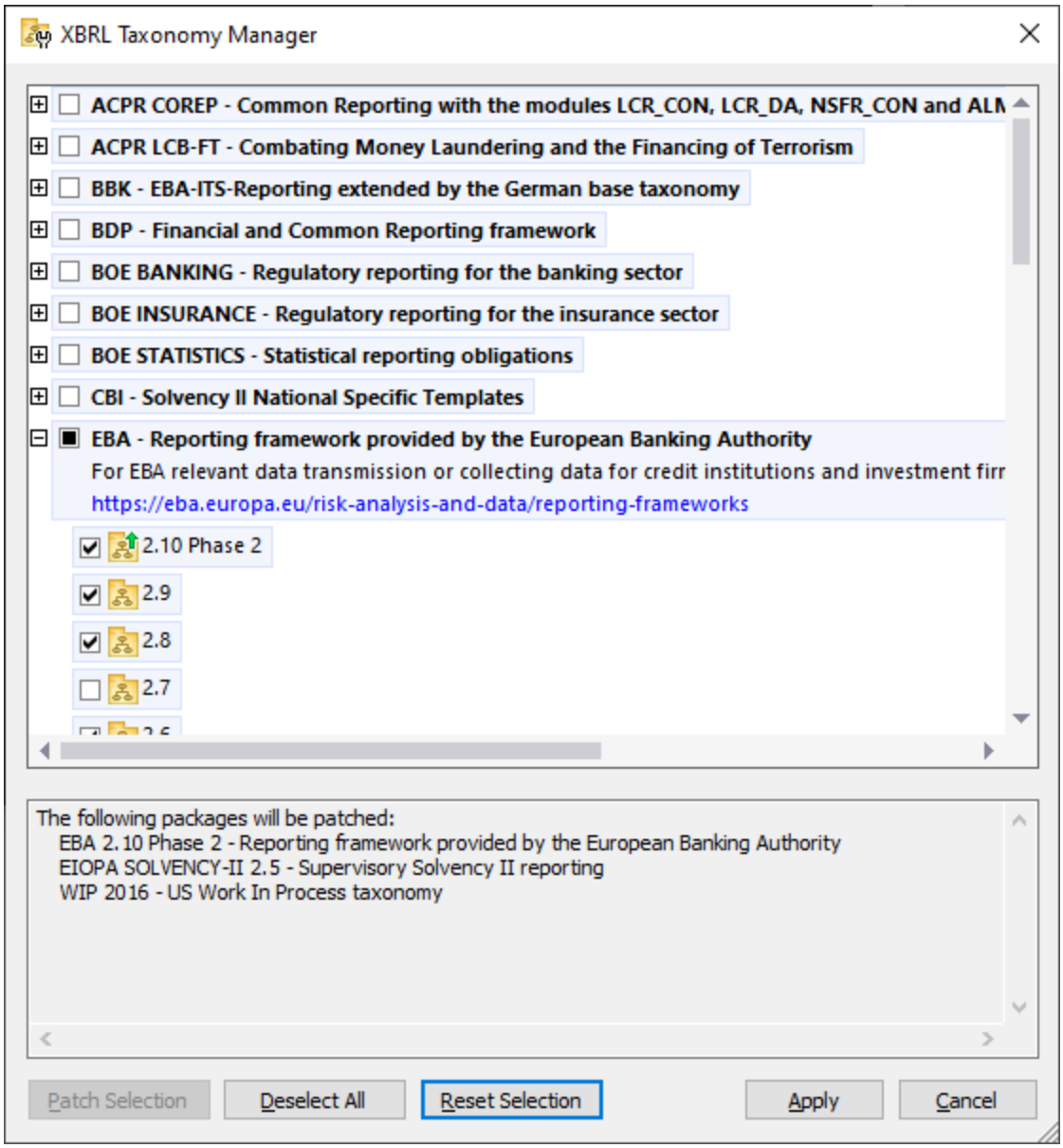
4.3 Apply Patches

Occasionally, XBRL taxonomies may receive patches from their issuers. When the XBRL Taxonomy Manager detects that patches are available, the following happens:

- If you use XBRL Taxonomy Manager through the Windows graphical user interface, the respective XBRL taxonomies are shown with the icon.
- If you use the command line or a Linux/macOS system, any XBRL taxonomies that have patches are listed when you run the executable with the `list -u` command.

To apply a patch on Windows:

1. Click the **Patch Selection** button. The icon of each XBRL taxonomy that qualifies changes from to , and the dialog box informs you about the patches that are to be applied, for example:



Note: The **Patch Selection** button is enabled only when there are patches available for any of the currently installed XBRL taxonomies.

2. Click **Apply**.

To apply a patch at the command line interface:

1. Run the `list -u` command. This lists any taxonomies where patch upgrades are available.
2. Run the `upgrade` command to install the patches.

For more information, see the reference to the Command Line Interface[48].

4.4 View Installed Taxonomies

To view all installed taxonomies from a graphical user interface, run [41] XBRL Taxonomy Manager. A selected check box next to a taxonomy (or a taxonomy version) indicates that that taxonomy is installed.

Command line interface

To view all available taxonomies from a command line interface, run:

```
TaxonomyManager.exe list
```

To view only installed taxonomies, run:

```
TaxonomyManager.exe list -i
```

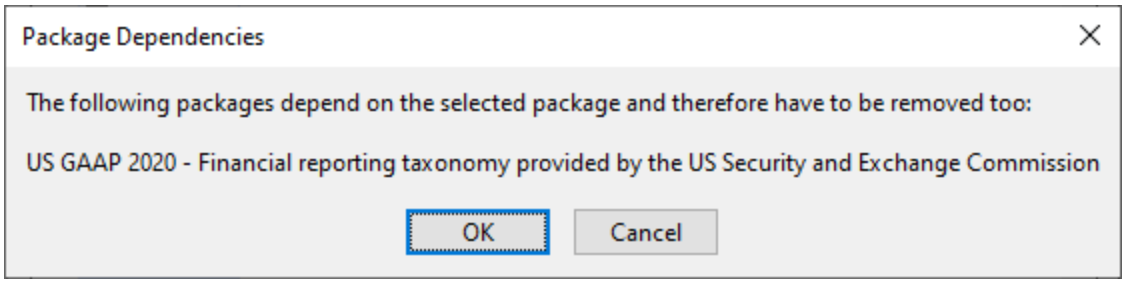
To view only taxonomies where a newer version is available, run:

```
TaxonomyManager.exe list -u
```

4.5 Uninstall a Taxonomy

To uninstall a taxonomy:

1. Run [41] XBRL Taxonomy Manager.
2. Clear the check box next to the taxonomies or taxonomy versions you want to uninstall. If the selected taxonomy is dependent on other taxonomy packages, a dialog box opens, informing you that the dependencies will be removed as well, for example:



3. Click **Apply** to remove the taxonomy and its dependencies.

Note the following:

- Installing or uninstalling a taxonomy from XBRL Taxonomy Manager takes effect for all users accounts on the same computer.
- Installing or uninstalling a taxonomy from XBRL Taxonomy Manager takes effect in all Altova XBRL-enabled applications installed on the same computer.
- If the current taxonomy has dependencies on other taxonomies, the dependent taxonomies are also installed (or uninstalled, as applicable).

Command line interface

To uninstall a taxonomy, run:

```
TaxonomyManager.exe uninstall FILTER...
```

Where `FILTER` means one of the following:

1. A taxonomy identifier in the format `<name>-<version>`, for example: **eba-2.10**, **us-gaap-2020.0**. To view all the available taxonomy identifiers and versions, run the `list` command.
2. An **.altova_taxonomies** file downloaded from the Altova website.

4.6 Command Line Interface

To call XBRL Taxonomy Manager at the command line, you need to know the path of the executable. By default, the XBRL Taxonomy Manager executable is installed at the following path:

*Linux**	`/opt/Altova/<%APPNAME-UL%>2021/bin/taxonomymanager`
*macOS**	`/usr/local/Altova/<%APPNAME-UL%>2021/bin/taxonomymanager`
Windows	`C:\ProgramData\Altova\SharedBetweenVersions\TaxonomyManager.exe`

* The Linux and macOS paths are applicable only if you are running XBRL Taxonomy Manager on those operating systems in conjunction with Altova cross-platform server applications such as MapForce Server, StyleVision Server, or RaptorXML+XBRL Server.

By convention, this documentation omits the full path of the executable when describing a given command, and uses **<exec>** instead of the executable name, for example:

```
<exec> help
```

Where **<exec>** is the path or name of the executable.

Calling XBRL Taxonomy Manager from the installation directory

To call the executable without having to type the full path, change the current directory to the one below:

Linux	`cd /opt/Altova/<%APPNAME-UL%>2021/bin`
macOS	`cd /usr/local/Altova/<%APPNAME-UL%>2021/bin`
Windows	`cd "C:\ProgramData\Altova\SharedBetweenVersions"`

You can now run a command by calling the executable with a relative path, for example:

Linux	`sudo ./taxonomymanager help`
macOS	`sudo ./taxonomymanager help`
Windows	`TaxonomyManager.exe help`

Note: On Linux and macOS systems, the prefix `./` indicates that the executable is in the current directory. The prefix **sudo** indicates that the command must be run with root privileges.

Calling XBRL Taxonomy Manager from any directory

To call the executable from any directory, refer to it using the absolute path. Alternatively, if you want to call the program by typing just the executable name, you can edit the PATH environment variable of your operating

system so that it includes the full path to the executable's directory. For ways to change the PATH environment variable, refer to the documentation of your operating system.

Notes:

- After changing the PATH environment variable, you may need to close the terminal window and open a new one, in order for the changes to take effect.
- On Linux and macOS, using **sudo** does not take into account the user's PATH.

Command line syntax

The general syntax for using the command line is as follows:

```
<exec> -h | --help | --version | <command> [options] [arguments]
```

In the listing above, the vertical bar **|** separates a set of mutually exclusive items. The square brackets **[]** indicate optional items. Essentially, you can type the executable path followed by either `--h`, `--help`, or `--version` options, or by a command. Each command may have options and arguments. The list of commands is described in the following sections.

4.6.1 help

This command provides contextual help about commands pertaining to XBRL Taxonomy Manager executable.

Syntax

```
<exec> help [command]
```

Where [command] is an optional argument which specifies any valid command name.

Remarks

You can also invoke help by typing a command followed by `-h` or `--help`, for example:

```
<exec> list -h
```

You can also invoke general help by typing `-h` or `--help` directly after the executable, for example:

```
<exec> --help
```

Example

The following command displays help about the `list` command:

Linux, macOS	`./taxonomymanager help list`
Windows	`TaxonomyManager.exe help list`

4.6.2 initialize

This command initializes XBRL Taxonomy Manager environment. It creates a cache directory where information about all taxonomies is stored. Initialization is performed automatically the first time when you install an Altova application that includes supports for XBRL Taxonomy Manager, so you don't need to run this command under normal circumstances. You typically need to run this command after executing the `reset` command.

Syntax

```
<exec> initialize [options]
```

The alias of this command is `init`.

Options

`--help, --h`	Display help about this command at the command line.
`--silent, --s`	Display only error messages. The default value is **false**.
`--verbose, --v`	Display more information during execution. The default value is **false**.

4.6.3 install

This command installs one or more taxonomies. Note the following:

- Installing or uninstalling a taxonomy from XBRL Taxonomy Manager takes effect for all users accounts on the same computer.
- Installing or uninstalling a taxonomy from XBRL Taxonomy Manager takes effect in all Altova XBRL-enabled applications installed on the same computer.
- If the current taxonomy has dependencies on other taxonomies, the dependent taxonomies are also installed (or uninstalled, as applicable).

Syntax

```
<exec> install [options] FILTER...
```

To specify multiple taxonomies to install, repeat `FILTER` as many times as necessary.

Arguments

`FILTER`	Where `FILTER` means one of the following: 1. A taxonomy identifier in the format `<name>-<version>`, for example: **eba-2.10**, **us-gaap-2020.0**. To view all the available taxonomy identifiers and versions, run the `list` command. 2. An **.altova_taxonomies** file downloaded from the Altova website.

	You can also use abbreviated identifiers if they are unique, for example, **eba**. If you use an abbreviated identifier, this will install the latest available version of that taxonomy.

Options

`--help, --h`	Display help about this command at the command line.
`--silent, --s`	Display only error messages. The default value is **false**.
`--verbose, --v`	Display more information during execution. The default value is **false**.

Example

The following command installs the latest **eba** (European Banking Authority) and **us-gaap** (US Generally Accepted Accounting Principles) taxonomies:

Linux, macOS	`./taxonomymanager install eba us-gaap`
Windows	`TaxonomyManager.exe install eba us-gaap`

4.6.4 list

Use this command to list taxonomies at the command line, in one of the following ways:

- list all available taxonomies
- list specific taxonomies
- list only installed taxonomies
- list only taxonomies that require upgrade.

Syntax

```
<exec> list [options] [FILTER...]
```

This command can be abbreviated with `ls`.

Arguments

`FILTER`	List only taxonomies that contain this string in their name. You can specify this argument multiple times.

Options

`--help, --h`	Display help about this command at the command line.
`--installed, --i`	List only installed taxonomies. The default value is **false**.

`--upgradeable, --u`	List only taxonomies where patch upgrades are available. The default value is **false**.

Examples

To list all available taxonomies, run:

Linux, macOS	`./taxonomymanager list`
Windows	`TaxonomyManager.exe list`

To list only installed taxonomies, run:

Linux, macOS	`./taxonomymanager list -i`
Windows	`TaxonomyManager.exe list -i`

To list all taxonomies that contain either "eba" or "us-gaap" in their name, run:

Linux, macOS	`./taxonomymanager list eba us-gaap`
Windows	`TaxonomyManager.exe list eba us-gaap`

4.6.5 reset

This command removes all installed taxonomies and the cache directory.

Warning: This command deletes all installed taxonomies and their information.

After running this command, make sure to run the `initialize` command, in order to recreate the cache directory. Alternatively, run the `reset` command with the `-i` option.

Note that `reset -i` restores the original installation of the product, so it's recommended to run the `update` command as well, after performing a reset. Alternatively, run the reset command with the `-i` and `-u` options.

Syntax

```
<exec> reset [options]
```

Options

`--help, --h`	Display help about this command at the command line.
`--init, --i`	Initialize the XBRL Taxonomy Manager environment after reset. Valid values are

	true and **false**. The default value is **false**.
`--silent, --s`	Display only error messages. The default value is **false**.
`--update, --u`	Initialize and update the XBRL Taxonomy Manager environment after reset. Valid values are **true** and **false**. The default value is **false**.
`--verbose, --v`	Display additional information during execution. The default value is **false**.

Examples

To reset the XBRL Taxonomy Manager, run:

Linux, macOS	`./taxonomymanager reset`
Windows	`TaxonomyManager.exe reset`

4.6.6 setdeflang

This command sets the language of XBRL Taxonomy Manager.

Syntax

```
<exec> setdeflang language
```

Where `language` is a mandatory argument supplying the language code. The alias of this command is `sdl`.

Arguments

`language`	The language to be set. Valid values are English (`en`), French (`fr`), German (`de`), Japanese (`ja`), and Spanish (`es`).

Options

`--help, --h`	Display help about this command at the command line.

Examples

To set the language to Spanish, run:

Linux, macOS	`./taxonomymanager setdeflang es`
Windows	`TaxonomyManager.exe setdeflang es`

4.6.7 uninstall

This command uninstalls one or more taxonomies. By default, any taxonomies referenced by the current one are uninstalled as well. To uninstall just the current taxonomy and keep the referenced taxonomies, set the option `--k`.

Syntax

```
<exec> uninstall FILTER...
```

To specify multiple taxonomies, repeat `FILTER` as many times as necessary.

Arguments

`FILTER`	Where `FILTER` means one of the following: 1. A taxonomy identifier in the format `<name>-<version>`, for example: **eba-2.10**, **us-gaap-2020.0**. To view all the available taxonomy identifiers and versions, run the `list` command. 2. An **.altova_taxonomies** file downloaded from the Altova website.

Options

`--help, --h`	Display help about this command at the command line.
`--keep-references, --k`	If this option is set, then referenced taxonomies are not uninstalled. The default value is **false**.
`--silent, --s`	Display only error messages. The default value is **false**.
`--verbose, --v`	Display additional information during execution. The default value is **false**.

Example

The following command uninstalls the **eba-2.10** and **us-gaap-2020.0** taxonomies and their dependencies:

Linux, macOS	`./taxonomymanager uninstall eba-2.10 us-gaap-2020.0`
Windows	`TaxonomyManager.exe uninstall eba-2.10 us-gaap-2020.0`

4.6.8 update

This command queries the list of taxonomies available from the online storage and updates the local cache directory. The update of this information takes place implicitly and you shouldn't need to run this command unless you have performed a `reset` and `initialize`.

Syntax

```
<exec> update [options]
```

Options

`--help, --h`	Display help about this command at the command line.
`--silent, --s`	Display only error messages. The default value is **false**.
`--verbose, --v`	Display additional information during execution. The default value is **false**.

Example

The following command updates the taxonomies information explicitly.

Linux, macOS	`./taxonomymanager update`
Windows	`TaxonomyManager.exe update`

4.6.9 upgrade

This command upgrades all eligible taxonomies to the latest available *patch* version. In other words, it performs only upgrades at patch level of a specific release. Running this command is meaningful only if there are upgradeable taxonomies available. You can identify upgradeable taxonomies by running the `list -u` command.

Syntax

```
<exec> upgrade [options]
```

Options

`--help, --h`	Display help about this command at the command line.
`--silent, --s`	Display only error messages. The default value is **false**.
`--verbose, --v`	Display additional information during execution. The default value is **false**.

5 Command Reference

The add-in commands available in the **EBA** tab of the Excel ribbon are listed below.

Insert New Report

Creates a new EBA filing report, see Creating a New Report 11. This command is disabled if the report sheet has already been inserted into the workbook.

Import from XBRL

Imports an XBRL instance file into the current Excel spreadsheet (see Importing Data from XBRL 33). This command is disabled if a report has already been inserted into the workbook. To enable the command, save and close the current report (workbook), and create a new workbook.

Export to XBRL

Exports data from all currently active sheets to an XBRL instance file (see Exporting Data to XBRL 32).

Validate

Performs a validation of report data against the underlying XBRL taxonomy and displays the validation results in a dialog box (see Validating Data 29).

Batch Conversion

Converts multiple XBRL instance files to Excel, see Batch Conversion fom XBRL to Excel 35.

Toggle EBA Filing Pane

Toggles the **EBA Filing Pane** on or off. By default, this pane is visible.

Toggle Validation Report

Shows or hides the "Validation Report" window, see Validating Data 29.

Add Sheet (z-Axis)

Adds a new sheet which provides the ability to enter data in a third dimension. This button is enabled only if the table supports a third dimension according to XBRL taxonomy. For more information, see Entering Data into Three-Dimensional Tables (Z-Axis) 22.

Remove Sheet (z-Axis)

Removes a previously added Z-axis sheet.

Add Row

Adds a new row to the currently selected table. This button is enabled only if the table supports growing vertically.

Delete Row

Deletes an existing row. This button is enabled only if the table supports growing vertically.

Add Column

Adds a new column to the currently selected table. This button is enabled only if the table supports growing horizontally.

Delete Column

Removes an existing column. This button is enabled only if the table supports growing horizontally.

Manage Taxonomies

This command opens the XBRL Taxonomy Manager tool, from where you can view, install, and uninstall XBRL taxonomies. See Managing XBRL Taxonomies 37.

Settings

Displays a dialog box where you can view or change the add-in settings 58.

Help

Opens this help file, in CHM (Microsoft Compiled HTML Help) format.

Add-In Activation

Displays the activation status of the add-in, or provides options to enter or purchase a license key code.

About EBA Add-In

Displays version information about the add-in.

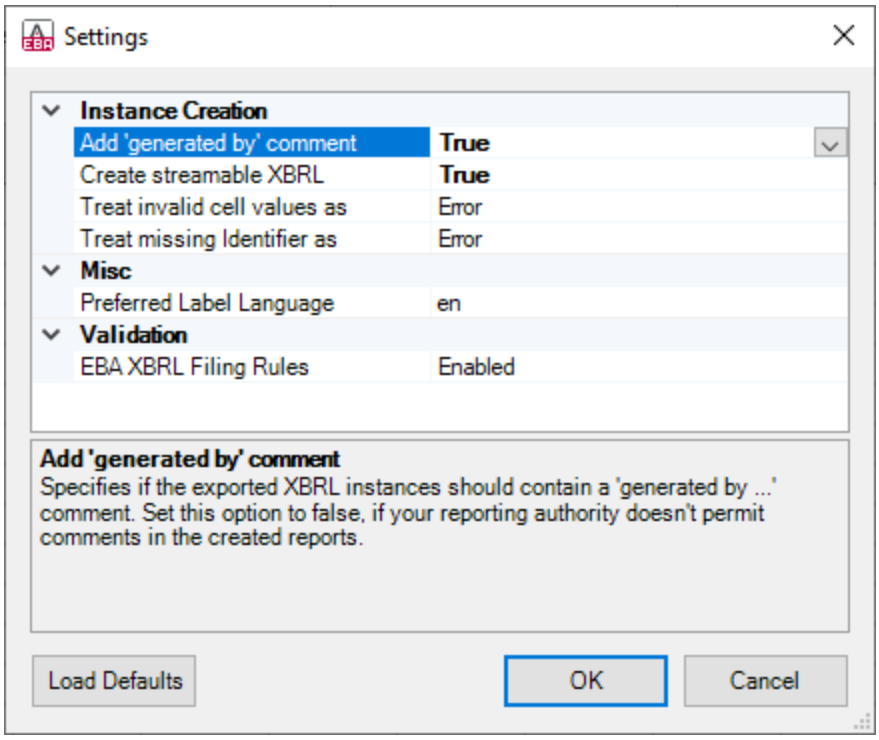
Altova on the Web

Provides links to the Altova website (including Online Support Center, XBRL Taxonomy Download Center, training and tutorials).

6 Settings

To view or change the add-in settings:

1. In the Excel ribbon, click the **EBA** tab.
2. Click **Settings**.



The settings you can configure are listed below.

Instance Creation

Add 'generated by' comment	Specifies if the exported XBRL instances should contain a "Generated by..." comment. Set this option to **false** if your reporting authority doesn't permit comments in the created reports. The default value is **true**.
Create streamable XBRL	Specifies if the exported XBRL instances should be streamable and contain the **xbrl-streamable-instance** processing instruction. Set this option to **false** if your reporting authority doesn't permit processing instructions in the created reports. The default value is **true**.
Treat invalid cell values as	Specifies whether invalid cell values should be treated as validation errors or warnings. The default value is "Error".
Treat missing Identifier as	After you create a new report, the **EBA Filing Pane** contains a property called **Identifier** which is by default empty. This option specifies whether an empty **Identifier**

	property should trigger a validation error or a warning. The default value is "Error".

Misc

Preferred Label Language	Specifies the preferred language to be used in the headers of created worksheets. Note that the respective label resources must be defined in the taxonomy for this setting to take effect. The default value is "en".

Validation

EBA XBRL Filing Rules	Specifies if the additional EBA XBRL filing rules (as specified by the "EBA XBRL Filing Rules" document) should be checked. The default value is "Enabled".

7 COM API

The add-in provides a COM API that can be used from programming languages that support interacting with Excel and accessing COM objects programmatically, such as VBA or .NET languages. Specifically, the API provides the means to create, import and export XBRL reports, and also to read and write form data.

Platform requirements:

- .NET Framework 4 or later
- Visual Studio Tools for Office runtime, version 4.0 or later

If you intend to distribute the API to other clients, note the following:

1. Excel and **Altova® European Banking Authority (EBA) XBRL add-in for Excel** must be installed on each client machine.
2. Each API client that consumes your custom code or application must hold a valid **Altova® European Banking Authority (EBA) XBRL add-in for Excel** license.

7.1 Accessing the API

You can access programmatically the add-in's COM API in one of the following ways:

- from within Excel, by using Visual Basic for Applications (see also https://docs.microsoft.com/en-us/office/vba/library-reference/concepts/getting-started-with-vba-in-office)
- from your custom program, by using the Office Interop API from any .NET language.

The main interface is the `IAutomationAPI` interface. The following code listing illustrates how to create a new instance of the automation object in VBA.

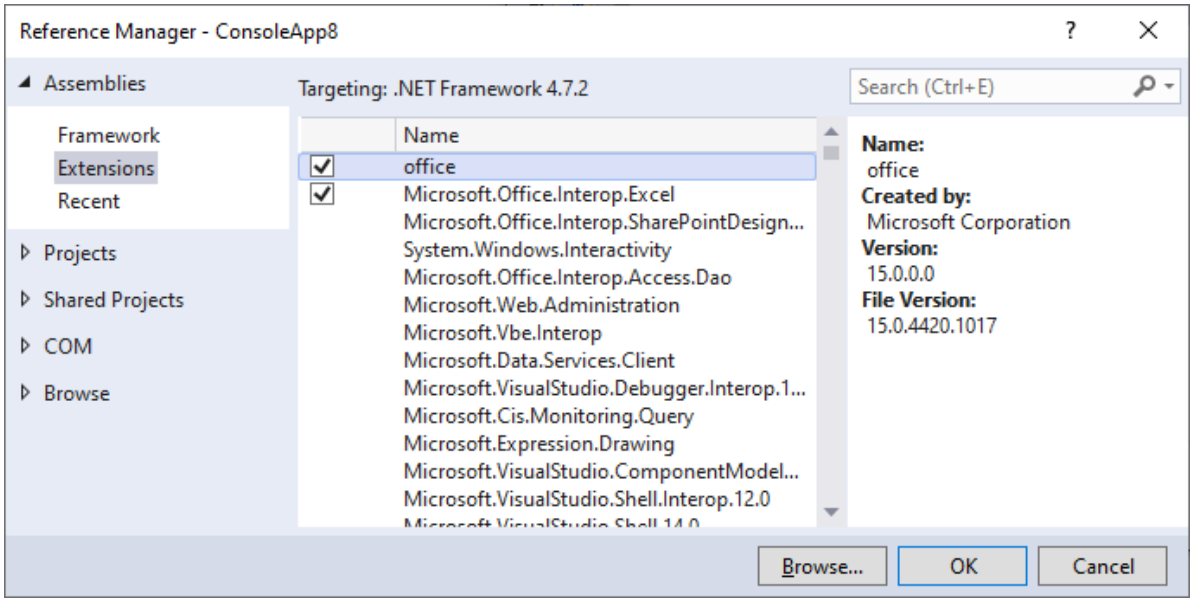
VBA

```
Dim automationObject As Object
Set automationObject = Application.COMAddIns.Item("Altova.EBAAddIn").Object
```

Accessing the COM API from a .NET project

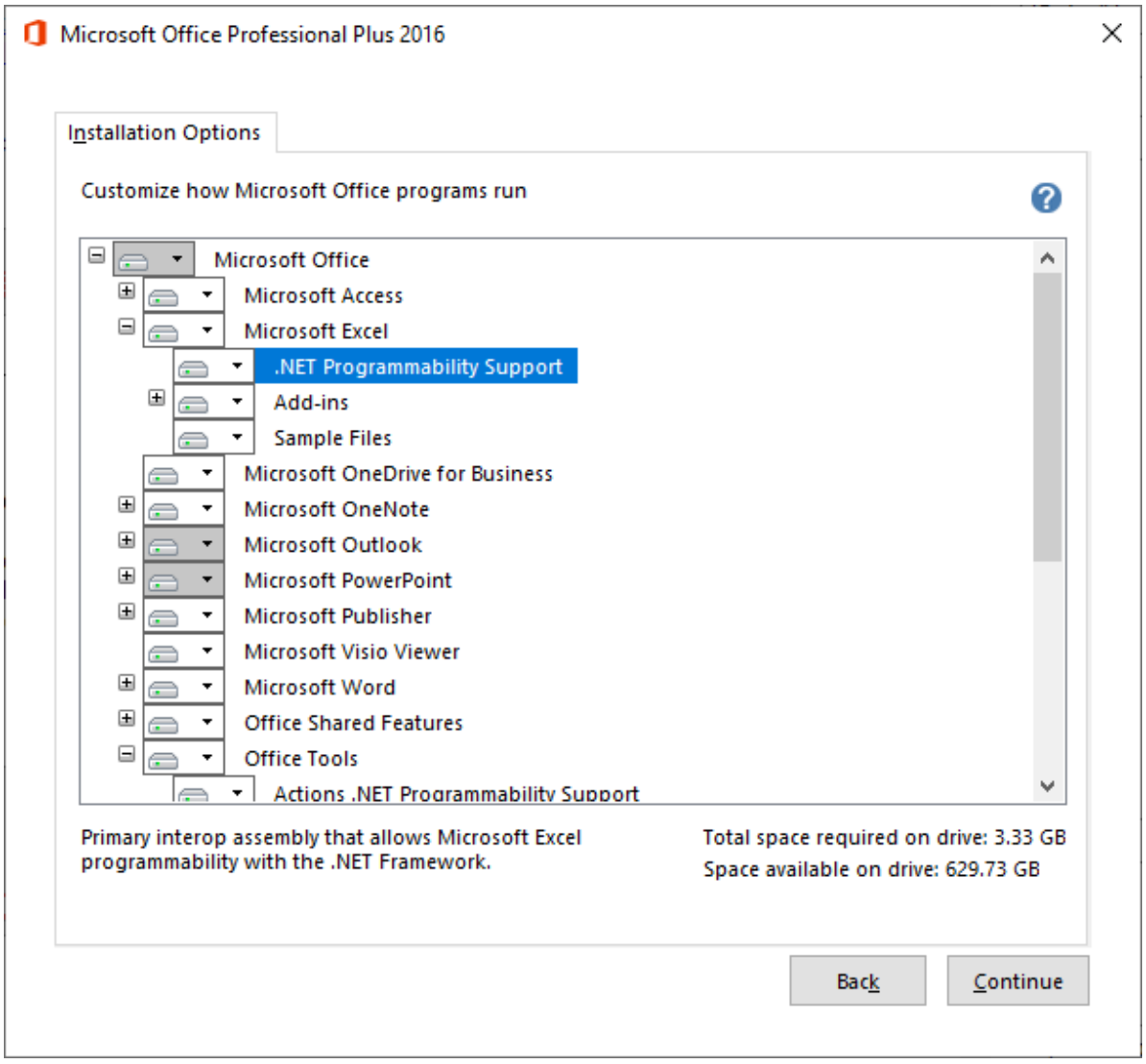
To access the COM API from a Visual Studio .NET project, add a reference to the **Microsoft Office Object Library (office.dll)** and **Microsoft.Office.Interop.Excel** assemblies:

1. In **Solution Explorer**, right-click your project's name and then click **Add Reference**. The **Add Reference** dialog box appears.
2. On the **Assemblies** page, select **office** and **Microsoft.Office.Interop.Excel** from the component list, and click **OK**.



If you do not see the assemblies above:

1. Make sure that you have installed Microsoft Office and that you have selected the **.NET Programmability Support** feature for Excel, for example:



2. Run the Visual Studio setup and make sure that you choose **Office/SharePoint development** workload (or the **Microsoft Office Developer Tools**, if applicable).

For more information about accessing Office interop assemblies from .NET projects, see https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/interop/how-to-access-office-onterop-objects.

After adding the assembly references, you can create a new add-in instance as shown below.

C#

```
// Make sure that your project references the following two assemblies:
//  * Microsoft.Office.InterOp.Excel
//  * Microsoft Office Object Library (office.dll)
var app = new Microsoft.Office.Interop.Excel.Application();
dynamic automationObject = app.COMAddIns.Item("Altova.EBAAddIn").Object;
```

7.2 C# Example

Each of the C# code listings illustrated below represents the **Program.cs** file in a standard .NET Framework console application.

> Before attempting to run the program code, make sure that you have added the required assembly references to the Visual Studio project, as described in Accessing the API 61.

Export XBRL from Excel

The following code listing illustrates how to create a new Excel workbook using a specific XBRL entry point, populate a few properties and data cells, and then save data to an XBRL file on the disk. The code will attempt to save both the Excel workbook and the XBRL file to the **C:\XBRL_Examples** directory.

```
using System;
using Excel = Microsoft.Office.Interop.Excel;

namespace EbaAddinClient
{
    class Program
    {
        static void Main(string[] args)
        {
            var app = new Excel.Application();

            try
            {
                // Suppress Excel alerts and create a new workbook
                Console.WriteLine("Creating a new workbook...");
                app.DisplayAlerts = false;
                var wb = (Excel._Workbook)(app.Workbooks.Add());

                // Get the Automation API object
                Console.WriteLine("Getting the COM automation object...");
                dynamic addIn = app.COMAddIns.Item("Altova.EBAAddIn");
                dynamic automationObject = addIn.Object;

                // Create a new report using taxonomy entry point
                Console.WriteLine("Creating the new report...");
                automationObject.InsertNewReport("http://www.eba.europa.eu/eu/fr/xbrl/crr
/fws/fp/gl-2019-05/2020-02-29/mod/fp_con.xsd");

                // Set the report properties
                Console.WriteLine("Setting the report properties...");
                var rp = automationObject.GetReportProperties(wb);
                rp.ReportingEntityScheme = "http://standards.iso.org/iso/17442";
                rp.ReportingEntityIdentifier = "123456";

                // Find table by its code and ensure it is included in this report
                var tab = automationObject.GetTableTree(wb);
                var tableNode = tab.FindTableByRCCode("P 00.01");
```

```
                tableNode.IncludeInFiling = true;

                // Populate cells
                Console.WriteLine("Populating cells...");
                tableNode.Forms.Item(0).DataRange.Item(1).Value = "National GAAP";
                tableNode.Forms.Item(0).DataRange.Item(2).Value = "Consolidated";

                // Export data to XBRL
                Console.WriteLine("Exporting the XBRL instance...");
                automationObject.ExportXBRL(wb, @"C:\XBRL_Examples\Example.xbrl");

                // Save and close the .xlsx workbook
                Console.WriteLine("Saving the .xlsx file...");
                wb.SaveAs(@"C:\XBRL_Examples\Example.xlsx");
                wb.Close();

                Console.WriteLine("Task completed.");
            }
            catch (Exception e)
            {
                Console.WriteLine(e.Message);
            }
            finally
            {
                app.DisplayAlerts = true;
                app.Quit();
            }
        }
    }
}
```

Import XBRL to Excel

The following code listing illustrates how to convert an XBRL file to an Excel file. To run this example successfully, an XBRL instance file must exist at **C:\XBRL_Examples\Example.xbrl**; otherwise, change the path accordingly. You can create an XBRL file either by running the previous code listing, or manually from Excel, by using the **Export** command, see Exporting Data to XBRL 32.

```
using System;
using Excel = Microsoft.Office.Interop.Excel;

namespace ConsoleApp1
{
    class Program
    {
        static void Main(string[] args)
        {
            var app = new Excel.Application();
            try
            {
                // Suppress Excel alerts and create a new workbook
                Console.WriteLine("Creating a new workbook...");
                app.DisplayAlerts = false;
```

```
                var wb = (Excel._Workbook)(app.Workbooks.Add());

                // Get the Automation API object
                Console.WriteLine("Getting the COM automation object...");
                dynamic addIn = app.COMAddIns.Item("Altova.EBAAddIn");
                dynamic automationObject = addIn.Object;

                // Import EBA report eba_example.xbrl
                Console.WriteLine("Importing XBRL...");
                automationObject.ImportXBRL(@"C:\XBRL_Examples\Example.xbrl");

                // Save as xlsx
                Console.WriteLine("Saving the .xlsx file...");
                wb.SaveAs(@"C:\XBRL_Examples\Example.xlsx");
                wb.Close();

                Console.WriteLine("Task complete.");
            }
            catch (Exception e)
            {
                Console.WriteLine(e.Message);
            }
            finally
            {
                app.DisplayAlerts = true;
                app.Quit();
            }
        }
    }
}
```

7.3 VBA Example

The following code listing illustrates how to insert an XBRL report into an Excel file and populate the first cell, using VBA.

```
' VBA Example 1:
' Creates a new EBA COREP ALM report with form 'C 68.00.a' and sets the value of the
first cell
Sub Example1()
    Dim addIn As COMAddIn
    Dim automationObject As Object
    Dim Workbook As Object
    Dim tableTree As Object
    Dim tableNode As Object

    ' Get the Automation API object
    Set addIn = Application.COMAddIns.Item("Altova.EBAAddIn")
    Set automationObject = addIn.Object

    ' Insert a new EBA 2.9.1 COREP ALM report
    Set Workbook =
automationObject.InsertNewReport("http://www.eba.europa.eu/eu/fr/xbrl/crr/fws/corep/cir-
680-2014/2019-11-15/mod/corep_alm_con.xsd")
    Set tableTree = automationObject.GetTableTree(Workbook)

    ' Find table tree node for form 'C 68.00.a'
    Set tableNode = tableTree.FindTableByRCCode("C 68.00.a")

    ' Include this table in the filing (this will also create the respective Excel
worksheet)
    tableNode.IncludeInFiling = True

    ' Get the Data range of this form and set the value of the first cell to 42
    tableNode.Forms.Item(0).DataRange.Item(1).Value = "42"
End Sub
```

7.4 API Reference

This section provides reference to the objects of the **Altova® European Banking Authority (EBA) XBRL add-in for Excel** COM API. The objects are described in a generic manner, since the API may be used with virtually any language that supports calling a COM object.

7.4.1 Interfaces

7.4.1.1 IAutomationAPI

The `IAutomationAPI` interface is the main automation interface of the **Altova® European Banking Authority (EBA) XBRL add-in for Excel**. This interface is the starting point to do any further operations with the add-in or to retrieve or create other related automation objects. It allows you to create, import and export reports, and also to read and write form data. For information about creating an instance of this interface, see Accessing the API 61.

Methods

Name	Description
`InsertNewReport` 68	Use this method to insert a new report of the respective taxonomy entry point.
`ImportXBRL` 68	Imports an XBRL report. Returns the Excel workbook that contains the imported XBRL report.
`ExportXBRL` 68	Exports the report from the respective Excel workbook to XBRL, and also validates it. To get the validation results, call `GetValidationReport` after calling this method.
`Validate` 69	Validates the current report. To get the validation results, call `GetValidationReport` after calling this method.
`GetValidationReport` 69	Returns an `IValidationReport` object representing the validation report currently shown in the validation report pane.
`GetEntryPointTree` 70	Returns an `IEntryPointTree` object representing a tree of the available taxonomy entry points.
`GetTableTree` 70	Returns an `ITableTree` object representing the tree of the available tables in the report opened in the specified Excel workbook.
`GetReportProperties` 70	Returns an `IReportProperties` object providing the properties of the XBRL report.
`GetFormProperties` 71	Returns an `IFormProperties` object providing the properties of the XBRL form in the specified Excel worksheet.

Name	Description
GetCellProperties 71	Returns an ICellProperties object providing the properties of the fact in the specified Excel range.

7.4.1.1.1 Methods

7.4.1.1.1.1 InsertNewReport

Use this method to insert a new report of the respective taxonomy entry point.

Signature

```
InsertNewReport(in entryPointUrl:String) -> Microsoft.Office.Interop.Excel.Workbook
```

Parameters

Name	Type	Description
entryPointUrl	String	The URI of the taxonomy a report should be created for. Use the GetEntryPointTree method to get the available entry points.

7.4.1.1.1.2 ImportXBRL

Imports an XBRL report. Returns the Excel workbook that contains the imported XBRL report.

Signature

```
ImportXBRL(in inputFile:String) -> Microsoft.Office.Interop.Excel.Workbook
```

Parameters

Name	Type	Description
inputFile	String	The path to the XBRL report which should be imported.

7.4.1.1.1.3 ExportXBRL

Exports the report from the respective Excel workbook to XBRL, and also validates it. To get the validation results, call GetValidationReport after calling this method.

Signature

```
ExportXBRL(in workbook:Microsoft.Office.Interop.Excel.Workbook, in outputFile:String) ->
Void
```

Parameters

Name	Type	Description
workbook	`Microsoft.Office.Interop.Excel.Workbook`	The Excel workbook to be exported.
outputFile	`String`	The path to the output XBRL file.

7.4.1.1.1.4 Validate

Validates the current report. To get the validation results, call `GetValidationReport` after calling this method.

Signature

```
Validate(in workbook:Microsoft.Office.Interop.Excel.Workbook) -> Void
```

Parameters

Name	Type	Description
workbook	`Microsoft.Office.Interop.Excel.Workbook`	The Excel workbook to be validated.

7.4.1.1.1.5 GetValidationReport

Returns an `IValidationReport` object representing the validation report currently shown in the validation report pane.

Signature

```
GetValidationReport(in workbook:Microsoft.Office.Interop.Excel.Workbook) ->
IValidationReport
```

Parameters

Name	Type	Description
workbook	`Microsoft.Office.Interop.Exc`	The Excel workbook from which to

Name	Type	Description
	el.Workbook	get the validation report.

7.4.1.1.1.6 GetEntryPointTree

Returns an `IEntryPointTree` object representing a tree of the available taxonomy entry points.

Signature

```
GetEntryPointTree() -> IEntryPointTree
```

7.4.1.1.1.7 GetTableTree

Returns an `ITableTree` object representing the tree of the available tables in the report opened in the specified Excel workbook.

Signature

```
GetTableTree(in workbook:Microsoft.Office.Interop.Excel.Workbook) -> ITableTree
```

Parameters

Name	Type	Description
workbook	Microsoft.Office.Interop.Excel.Workbook	The Excel workbook containing the XBRL report.

7.4.1.1.1.8 GetReportProperties

Returns an `IReportProperties` object providing the properties of the XBRL report.

Signature

```
GetReportProperties(in workbook:Microsoft.Office.Interop.Excel.Workbook) ->
IReportProperties
```

Parameters

Name	Type	Description
workbook	Microsoft.Office.Interop.Excel.Workbook	The workbook containing the XBRL report.

7.4.1.1.1.9 GetFormProperties

Returns an `IFormProperties` object providing the properties of the XBRL form in the specified Excel worksheet.

Signature

```
GetFormProperties(in worksheet:Microsoft.Office.Interop.Excel.Worksheet) ->
IFormProperties
```

Parameters

Name	Type	Description
worksheet	`Microsoft.Office.Interop.Excel.Worksheet`	An Excel worksheet containing an XBRL form.

7.4.1.1.1.10 GetCellProperties

Returns an `ICellProperties` object providing the properties of the fact in the specified Excel range.

Signature

```
GetCellProperties(in cell:Microsoft.Office.Interop.Excel.Range) -> ICellProperties
```

Parameters

Name	Type	Description
cell	`Microsoft.Office.Interop.Excel.Range`	An Excel cell containing an XBRL fact (must be within the data range of a form).

7.4.1.2 IEntryPointTree

The `IEntryPointTree` interface provides information about the available taxonomy entry points in a structured way.

Properties

Name	Description
`Groups` 72	Read-only.

Name	Description
	Returns a collection of `IEntryPointGroup` representing the available taxonomy groups, for example, "2.9" or "Bank of England Banking Taxonomy 3.1.1".

7.4.1.2.1 Properties

7.4.1.2.1.1 Groups

Returns a collection of `IEntryPointGroup` representing the available taxonomy groups, for example, "2.9" or "Bank of England Banking Taxonomy 3.1.1".

Signature

```
Groups : Collection
```

7.4.1.3 IEntryPointGroup

The `IEntryPointGroup` interface provides information about a group of taxonomy entry points.

Properties

Name	Description
`Name` 73	Read-only. The Name of the group. For example, "EBA 2.9".
`Country` 73	Read-only. The country code for which this entry point group is relevant. This is set only for country-specific taxonomies such as "Bank of England Banking Taxonomy".
`Version` 73	Read-only. The version of the taxonomy. May be empty for sub-groups.
`IsCurrentVersion` 73	Read-only. **True** if this group contains the current version of the taxonomy, **false** for older versions.
`Groups` 73	Read-only. A collection of `IEntryPointGroup` representing the sub-groups of this entry point group.
`EntryPoints` 74	Read-only. A collection of `IEntryPoint` representing the specific taxonomy entry points of this group.

7.4.1.3.1 Properties

7.4.1.3.1.1 Name

The Name of the group. For example, "EBA 2.9".

Signature

```
Name : String
```

7.4.1.3.1.2 Country

The country code for which this entry point group is relevant. This is set only for country-specific taxonomies such as "Bank of England Banking Taxonomy".

Signature

```
Country : String
```

7.4.1.3.1.3 Version

The version of the taxonomy. May be empty for sub-groups.

Signature

```
Version : String
```

7.4.1.3.1.4 IsCurrentVersion

True if this group contains the current version of the taxonomy, **false** for older versions.

Signature

```
IsCurrentVersion : Boolean
```

7.4.1.3.1.5 Groups

A collection of `IEntryPointGroup` representing the sub-groups of this entry point group.

Signature

```
Groups : Collection
```

7.4.1.3.1.6 EntryPoints

A collection of `IEntryPoint` representing the specific taxonomy entry points of this group.

Signature

```
EntryPoints : Collection
```

7.4.1.4 IEntryPoint

The `IEntryPoint` interface provides information about a specific taxonomy entry point.

Properties

Name	Description
Name 75	Read-only. The name of the entry point. For example, "Additional Liquidity Monitoring - COREP, Consolidated"
ShortName 75	Read-only. The abbreviated form of the entry points name. For example, "COREP ALM Con".
Version 75	Read-only. The version of the entry point. This may be empty (use the parents group version in this case).
URI 75	Read-only. The URI of the taxonomy entry point, for example, "http://www.eba.europa.eu/eu/fr/xbrl/crr/fws/corep/cir-680-2014/2019-04-30/mod/corep_alm_con.xsd". Pass this to the `IAutomationAPI.InsertNewReport` method.
Needs64Bit 75	Read-only. True if the entry point requires the 64-bit version of Excel, false otherwise.

7.4.1.4.1 Properties

7.4.1.4.1.1 Name

The name of the entry point. For example, "Additional Liquidity Monitoring - COREP, Consolidated"

Signature

```
Name : String
```

7.4.1.4.1.2 ShortName

The abbreviated form of the entry points name. For example, "COREP ALM Con".

Signature

```
ShortName : String
```

7.4.1.4.1.3 Version

The version of the entry point. This may be empty (use the parents group version in this case).

Signature

```
Version : String
```

7.4.1.4.1.4 URI

The URI of the taxonomy entry point, for example, "http://www.eba.europa.eu/eu/fr/xbrl/crr/fws/corep/cir-680-2014/2019-04-30/mod/corep_alm_con.xsd". Pass this to the `IAutomationAPI.InsertNewReport` method.

Signature

```
URI : String
```

7.4.1.4.1.5 Needs64Bit

True if the entry point requires the 64-bit version of Excel, false otherwise.

Signature

```
Needs64Bit : Boolean
```

7.4.1.5 ITableTree

The `ITableTree` interface provides structured information about the available tables and forms in an XBRL report.

Properties

Name	Description
Nodes 76	Read-only. Returns a collection of `IGroupNode` and `ITableNode` objects, which represent groups of tables and tables, respectively.

Methods

Name	Description
FindTableByRCCode 77	Returns the table node with the specified RC Code.

7.4.1.5.1 Properties

7.4.1.5.1.1 Nodes

Returns a collection of `IGroupNode` and `ITableNode` objects, which represent groups of tables and tables, respectively.

Signature

```
Nodes : Collection
```

7.4.1.5.2 Methods

7.4.1.5.2.1 FindTableByRCCode

Returns the table node with the specified RC Code.

Signature

```
FindTableByRCCode(in rcCode:string) -> ITableNode
```

Parameters

Name	Type	Description
rcCode	`string`	The RC Code of the desired table node.

7.4.1.6 IGroupNode

The `IGroupNode` interface provides information about a group of tables.

Properties

Name	Description
`Text` 78	Read-only. The name of the group of tables as displayed in the EBA Filing Pane.
`IsGroup` 78	Read-only. This is always **true** for group nodes. Use this to distinguish between `IGroupNode` and `ITableNode` members of the `Nodes` collection.
`IsTable` 78	Read-only. This is always **false** for group nodes. Use this to distinguish between `IGroupNode` and `ITableNode` members of the `Nodes` collection.
`Nodes` 78	Read-only. Returns a collection of `IGroupNode` and `ITableNode` objects, which represent groups of tables and tables, respectively.

7.4.1.6.1 Properties

7.4.1.6.1.1 Text

The name of the group of tables as displayed in the EBA Filing Pane.

Signature

```
Text : String
```

7.4.1.6.1.2 IsGroup

This is always **true** for group nodes. Use this to distinguish between `IGroupNode` and `ITableNode` members of the `Nodes` collection.

Signature

```
IsGroup : Boolean
```

7.4.1.6.1.3 IsTable

This is always **false** for group nodes. Use this to distinguish between `IGroupNode` and `ITableNode` members of the `Nodes` collection.

Signature

```
IsTable : Boolean
```

7.4.1.6.1.4 Nodes

Returns a collection of `IGroupNode` and `ITableNode` objects, which represent groups of tables and tables, respectively.

Signature

```
Nodes : Collection
```

7.4.1.7 ITableNode

The `ITableNode` interface provides information about a table.

Properties

Name	Description
`Text` 80	Read-only. The name of the table as displayed in the EBA Filing Pane.
`IsGroup` 80	Read-only. This is always **false** for table nodes. Use this to distinguish between `IGroupNode` and `ITableNode` members of the `Nodes` collection.
`IsTable` 80	Read-only. This is always **true** for table nodes. Use this to distinguish between `IGroupNode` and `ITableNode` members of the `Nodes` collection.
`Forms` 80	Read-only. Returns a collection of `IForm` objects, which represent concrete forms that can be displayed as Excel worksheet.
`CanAddSubForm` 80	Read-only. This is **true** if additional sub forms can be added to the table, for example, if the table has open aspects on the z-Axis. In most cases, this is a form for each country or currency.
`FilingIndicator` 81	Read-only. The filing indicator code of the table.
`RCCode` 81	Read-only. The RC code of the table.
`IncludeInFiling` 81	**True** if the table should be part of the report, **false** otherwise. If you set this property to **true** for the first time, a new worksheet for this table will be created.

Methods

Name	Description
`AddSubForm` 81	Creates a new sub-form of this table and returns the respective `IForm` object. This method returns null if no sub-form can be added.

7.4.1.7.1 Properties

7.4.1.7.1.1 Text

The name of the table as displayed in the EBA Filing Pane.

Signature

```
Text : String
```

7.4.1.7.1.2 IsGroup

This is always **false** for table nodes. Use this to distinguish between `IGroupNode` and `ITableNode` members of the `Nodes` collection.

Signature

```
IsGroup : Boolean
```

7.4.1.7.1.3 IsTable

This is always **true** for table nodes. Use this to distinguish between `IGroupNode` and `ITableNode` members of the `Nodes` collection.

Signature

```
IsTable : Boolean
```

7.4.1.7.1.4 Forms

Returns a collection of `IForm` objects, which represent concrete forms that can be displayed as Excel worksheet.

Signature

```
Forms : Collection
```

7.4.1.7.1.5 CanAddSubForm

This is **true** if additional sub forms can be added to the table, for example, if the table has open aspects on the z-Axis. In most cases, this is a form for each country or currency.

Signature

```
CanAddSubForm : Boolean
```

7.4.1.7.1.6 FilingIndicator

The filing indicator code of the table.

Signature

```
FilingIndicator : String
```

7.4.1.7.1.7 RCCode

The RC code of the table.

Signature

```
RCCode : String
```

7.4.1.7.1.8 IncludeInFiling

True if the table should be part of the report, **false** otherwise. If you set this property to **true** for the first time, a new worksheet for this table will be created.

Signature

```
IncludeInFiling : Boolean
```

7.4.1.7.2 Methods

7.4.1.7.2.1 AddSubForm

Creates a new sub-form of this table and returns the respective `IForm` object. This method returns null if no sub-form can be added.

Signature

```
AddSubForm() -> IForm
```

7.4.1.8 IForm

The `IForm` interface provides information about a form.

Properties

Name	Description
`Text` 82	Read-only. The name of the form as displayed in the EBA Filing Pane.
`DataRange` 83	Read-only. The Excel range containing the data of this form.
`FormSelectorRange` 83	Read-only. The Excel range containing the form selector of this form. Namely, the cells that contain the data that distinguishes this form from the other forms of the same table. This returns null if the table may not consist of multiple forms.
`Worksheet` 83	Read-only. The Excel worksheet containing this form. This may be null if `IncludeInFiling` is false.
`IncludeInFiling` 83	**True** if this form should be part of the report, **false** otherwise. This shows/hides the respective Excel worksheet.

Methods

Name	Description
`Remove` 84	Removes this form and deletes the respective Excel worksheet.

7.4.1.8.1 Properties

7.4.1.8.1.1 Text

The name of the form as displayed in the EBA Filing Pane.

Signature

```
Text : String
```

7.4.1.8.1.2 DataRange

The Excel range containing the data of this form.

Signature

```
DataRange : Microsoft.Office.Interop.Excel.Range
```

7.4.1.8.1.3 FormSelectorRange

The Excel range containing the form selector of this form. Namely, the cells that contain the data that distinguishes this form from the other forms of the same table. This returns null if the table may not consist of multiple forms.

Signature

```
FormSelectorRange : Microsoft.Office.Interop.Excel.Range
```

7.4.1.8.1.4 Worksheet

The Excel worksheet containing this form. This may be null if `IncludeInFiling` is false.

Signature

```
Worksheet : Microsoft.Office.Interop.Excel.Worksheet
```

7.4.1.8.1.5 IncludeInFiling

True if this form should be part of the report, **false** otherwise. This shows/hides the respective Excel worksheet.

Signature

```
IncludeInFiling : Boolean
```

7.4.1.8.2 Methods

7.4.1.8.2.1 Remove

Removes this form and deletes the respective Excel worksheet.

Signature

```
Remove() -> Void
```

7.4.1.9 IReportProperties

The `IReportProperties` interface provides properties of the whole XBRL report.

Properties

Name	Description
EntryPointURI 85	Read-only. The URI of the taxonomy entry point. For example, "Additional Liquidity Monitoring - COREP, Consolidated".
EntryPointModuleName 85	Read-only. The module name of the taxonomy entry point. For example, "http://www.eba.europa.eu/eu/fr/xbrl/crr/fws/corep/cir-680-2014/2019-04-30/mod/corep_alm_con.xsd".
EntryPointModuleDPMID 85	Read-only. The Data Point Model Database ID of the module. For example, "218".
ReportingEntityScheme 85	The scheme of the reporting entity. For example, "http://standards.iso.org/iso/17442".
ReportingEntityIdentifier 86	The identifier of the reporting entity.
ReferenceDate 86	The reference date of the report.
MonetaryCellsAccuracy 86	The accuracy of monetary facts in this report. Applies to each monetary fact for which no separate accuracy was specified (at cell or table level).
PercentageCellsAccuracy 86	The accuracy of percentage facts in this report. Applies to each percentage fact for which no separate accuracy was specified (at cell or table level).
PureCellsAccuracy 87	The accuracy of pure facts in this report. Applies to each pure fact for which no separate accuracy was specified (at cell or

Name	Description
	table level).
ReportingCurrency 87	The reporting currency used in the XBRL report as an ISO 4217 currency code.
ReportingLanguage 87	The language of footnotes in the XBRL report as BCP-47 language tag. For example, "en-US".

7.4.1.9.1 Properties

7.4.1.9.1.1 EntryPointURI

The URI of the taxonomy entry point. For example, "Additional Liquidity Monitoring - COREP, Consolidated".

Signature

```
EntryPointURI : String
```

7.4.1.9.1.2 EntryPointModuleName

The module name of the taxonomy entry point. For example, "http://www.eba.europa.eu/eu/fr/xbrl/crr/fws/corep/cir-680-2014/2019-04-30/mod/corep_alm_con.xsd".

Signature

```
EntryPointModuleName : String
```

7.4.1.9.1.3 EntryPointModuleDPMID

The Data Point Model Database ID of the module. For example, "218".

Signature

```
EntryPointModuleDPMID : String
```

7.4.1.9.1.4 ReportingEntityScheme

The scheme of the reporting entity. For example, "http://standards.iso.org/iso/17442".

Signature

```
ReportingEntityScheme : String
```

7.4.1.9.1.5 ReportingEntityIdentifier

The identifier of the reporting entity.

Signature

```
ReportingEntityIdentifier : String
```

7.4.1.9.1.6 ReferenceDate

The reference date of the report.

Signature

```
ReferenceDate : DateTime
```

7.4.1.9.1.7 MonetaryCellsAccuracy

The accuracy of monetary facts in this report. Applies to each monetary fact for which no separate accuracy was specified (at cell or table level).

Signature

```
MonetaryCellsAccuracy : String
```

7.4.1.9.1.8 PercentageCellsAccuracy

The accuracy of percentage facts in this report. Applies to each percentage fact for which no separate accuracy was specified (at cell or table level).

Signature

```
PercentageCellsAccuracy : String
```

7.4.1.9.1.9 PureCellsAccuracy

The accuracy of pure facts in this report. Applies to each pure fact for which no separate accuracy was specified (at cell or table level).

Signature

```
PureCellsAccuracy : String
```

7.4.1.9.1.10 ReportingCurrency

The reporting currency used in the XBRL report as an ISO 4217 currency code.

Signature

```
ReportingCurrency : String
```

7.4.1.9.1.11 ReportingLanguage

The language of footnotes in the XBRL report as BCP-47 language tag. For example, "en-US".

Signature

```
ReportingLanguage : String
```

7.4.1.10 IFormProperties

The `IFormProperties` interface provides properties of one form of the report.

Properties

Name	Description
`MonetaryCellsAccuracy` 88	The accuracy of monetary facts in this form. Applies to each monetary fact for which no separate accuracy was specified.
`PercentageCellsAccuracy` 88	The accuracy of percentage facts in this form. Applies to each percentage fact for which no separate accuracy was specified.
`PureCellsAccuracy` 88	The accuracy of pure facts in this form. Applies to each pure fact for which no separate accuracy was specified.
`TableRCCode` 89	Read-only. The RC Code of the table. For example, "C 66.01.a".

Name	Description
FilingIndicatorCode 89	Read-only. The filing indicator code of the table. For example, "C_66.01".
Label 89	Read-only. The label of the table. For example, "C 66.01.a".
VerboseLabel 89	Read-only. The verbose label of the table. For example, "C 66.01.a Maturity ladder. Total. Overnight and higher maturity".
TableID 90	Read-only. The id of the table resource. For example, "eba_tC_66.01.a".
TableDPMID 90	Read-only. The Data Point Model Database ID of the table. For example, "429.1431".
ValidationRules 90	Read-only. The collection of validation rules that apply to this table.

7.4.1.10.1 Properties

7.4.1.10.1.1 MonetaryCellsAccuracy

The accuracy of monetary facts in this form. Applies to each monetary fact for which no separate accuracy was specified.

Signature

```
MonetaryCellsAccuracy : String
```

7.4.1.10.1.2 PercentageCellsAccuracy

The accuracy of percentage facts in this form. Applies to each percentage fact for which no separate accuracy was specified.

Signature

```
PercentageCellsAccuracy : String
```

7.4.1.10.1.3 PureCellsAccuracy

The accuracy of pure facts in this form. Applies to each pure fact for which no separate accuracy was specified.

Signature

```
PureCellsAccuracy : String
```

7.4.1.10.1.4 TableRCCode

The RC Code of the table. For example, "C 66.01.a".

Signature

```
TableRCCode : String
```

7.4.1.10.1.5 FilingIndicatorCode

The filing indicator code of the table. For example, "C_66.01".

Signature

```
FilingIndicatorCode : String
```

7.4.1.10.1.6 Label

The label of the table. For example, "C 66.01.a".

Signature

```
Label : String
```

7.4.1.10.1.7 VerboseLabel

The verbose label of the table. For example, "C 66.01.a Maturity ladder. Total. Overnight and higher maturity".

Signature

```
VerboseLabel : String
```

7.4.1.10.1.8 TableID

The id of the table resource. For example, "eba_tC_66.01.a".

Signature

```
TableID : String
```

7.4.1.10.1.9 TableDPMID

The Data Point Model Database ID of the table. For example, "429.1431".

Signature

```
TableDPMID : String
```

7.4.1.10.1.10 ValidationRules

The collection of validation rules that apply to this table.

Signature

```
ValidationRules : Collection
```

7.4.1.11 ICellProperties

The `ICellProperties` interface provides properties of one fact of the report.

Properties

Name	Description
`Accuracy` [91]	The accuracy of the numeric item as string. This may be "INF" or a number representing the number of decimal places to which this fact is accurate. This is null for non-numeric items.
`Footnote` [91]	The footnote of the fact.
`Name` [91]	Read-only. The concept name of the fact. For example, "eba_met:mi256".

Name	Description
Type 92	Read-only. The type of the fact. For example, "xbrli:monetaryItemType".
Label 92	Read-only. The label of the fact. For example, "Cash value".
DPMID 92	Read-only. The Data Point Model Database ID of this fact. For example, "6062".
Dimensions 92	Read-only. The collection of `IDimension` objects representing the dimensions for which this fact is reported.

7.4.1.11.1 Properties

7.4.1.11.1.1 Accuracy

The accuracy of the numeric item as string. This may be "INF" or a number representing the number of decimal places to which this fact is accurate.

This is null for non-numeric items.

Signature

```
Accuracy : String
```

7.4.1.11.1.2 Footnote

The footnote of the fact.

Signature

```
Footnote : String
```

7.4.1.11.1.3 Name

The concept name of the fact. For example, "eba_met:mi256".

Signature

```
Name : String
```

7.4.1.11.1.4 Type

The type of the fact. For example, "xbrli:monetaryItemType".

Signature

```
Type : String
```

7.4.1.11.1.5 Label

The label of the fact. For example, "Cash value".

Signature

```
Label : String
```

7.4.1.11.1.6 DPMID

The Data Point Model Database ID of this fact. For example, "6062".

Signature

```
DPMID : String
```

7.4.1.11.1.7 Dimensions

The collection of `IDimension` objects representing the dimensions for which this fact is reported.

Signature

```
Dimensions : Collection
```

7.4.1.12 IDimension

The `IDimension` interface provides basic information of a `Dimension` and its value.

Properties

Name	Description
`Name` 93	Read-only. The name of the dimension.
`Value` 93	Read-only. The value of the dimension as `String`.

7.4.1.12.1 Properties

7.4.1.12.1.1 Name

The name of the dimension.

Signature

```
Name : String
```

7.4.1.12.1.2 Value

The value of the dimension as `String`.

Signature

```
Value : String
```

7.4.1.13 IValidationReport

The `IValidationReport` interface provides access to the validation report.

Properties

Name	Description
`Messages` 94	Read-only. Returns a collection of `IValidationReportMessage` representing the main lines of the validation report.

Methods

Name	Description
`CreateTextReport` 94	Returns a textual representation of the validation report.
`CreateHTMLReport` 95	Returns an HTML representation of the validation report.

7.4.1.13.1 Properties

7.4.1.13.1.1 Messages

Returns a collection of `IValidationReportMessage` representing the main lines of the validation report.

Signature

```
Messages : Collection
```

7.4.1.13.2 Methods

7.4.1.13.2.1 CreateTextReport

Returns a textual representation of the validation report.

Signature

```
CreateTextReport() -> String
```

7.4.1.13.2.2 CreateHTMLReport

Returns an HTML representation of the validation report.

Signature

```
CreateHTMLReport() -> String
```

7.4.1.14 IValidationReportMessage

The `IValidationReportMessage` interface represents a main line in the validation report.

Properties

Name	Description
Severity 95	Read-only. Returns the severity of this report message as `String`. Possible values are: "success", "info", "warning" and "error".
Text 95	Read-only. Returns the value of this report message.
Details 96	Read-only. Returns the details of this report message (if any) as `String`.

7.4.1.14.1 Properties

7.4.1.14.1.1 Severity

Returns the severity of this report message as `String`. Possible values are: "success", "info", "warning" and "error".

Signature

```
Severity : String
```

7.4.1.14.1.2 Text

Returns the value of this report message.

Signature

```
Text : String
```

7.4.1.14.1.3 Details

Returns the details of this report message (if any) as `String`.

Signature

```
Details : String
```

8 License Information

This section contains information about:

- the distribution of this software product
- software activation and license metering
- the license agreement governing the use of this product

Please read this information carefully. It is binding upon you since you agreed to these terms when you installed this software product.

To view the terms of any Altova license, go to the Altova Legal Information page at the Altova website.

8.1 Electronic Software Distribution

This product is available through electronic software distribution, a distribution method that provides the following unique benefits:

- You can evaluate the software free-of-charge for 30 days before making a purchasing decision. *(Note: Altova MobileTogether Designer is licensed free of charge.)*
- Once you decide to buy the software, you can place your order online at the Altova website and get a fully licensed product within minutes.
- When you place an online order, you always get the latest version of our software.
- The product package includes an onscreen help system that can be accessed from within the application interface. The latest version of the user manual is available at www.altova.com in (i) HTML format for online browsing, and (ii) PDF format for download (and to print if you prefer to have the documentation on paper).

30-day evaluation period

After downloading this product, you can evaluate it for a period of up to 30 days free of charge. About 20 days into the evaluation period, the software will start to remind you that it has not yet been licensed. The reminder message will be displayed once each time you start the application. If you would like to continue using the program after the 30-day evaluation period, you must purchase a product license, which is delivered in the form of a license file containing a key code. Unlock the product by uploading the license file in the Software Activation dialog of your product.

You can purchase product licenses at https://shop.altova.com/.

Helping Others within Your Organization to Evaluate the Software

If you wish to distribute the evaluation version within your company network, or if you plan to use it on a PC that is not connected to the Internet, you may distribute only the installer file, provided that this file is not modified in any way. Any person who accesses the software installer that you have provided must request their own 30-day evaluation license key code and after expiration of their evaluation period, must also purchase a license in order to be able to continue using the product.

8.2 Software Activation and License Metering

As part of Altova's Software Activation, the software may use your internal network and Internet connection for the purpose of transmitting license-related data at the time of installation, registration, use, or update to an Altova-operated license server and validating the authenticity of the license-related data in order to protect Altova against unlicensed or illegal use of the software and to improve customer service. Activation is based on the exchange of license related data such as operating system, IP address, date/time, software version, and computer name, along with other information between your computer and an Altova license server.

Your Altova product has a built-in license metering module that further helps you avoid any unintentional violation of the End User License Agreement. Your product is licensed either as a single-user or multi-user installation, and the license-metering module makes sure that no more than the licensed number of users use the application concurrently.

This license-metering technology uses your local area network (LAN) to communicate between instances of the application running on different computers.

Single license

When the application starts up, as part of the license metering process, the software sends a short broadcast datagram to find any other instance of the product running on another computer in the same network segment. If it doesn't get any response, it will open a port for listening to other instances of the application.

Multi-user license

If more than one instance of the application is used within the same LAN, these instances will briefly communicate with each other on startup. These instances exchange key-codes in order to help you to better determine that the number of concurrent licenses purchased is not accidentally violated. This is the same kind of license metering technology that is common in the Unix world and with a number of database development tools. It allows Altova customers to purchase reasonably-priced concurrent-use multi-user licenses.

We have also designed the applications so that they send few and small network packets so as to not put a burden on your network. The TCP/IP ports (2799) used by your Altova product are officially registered with the IANA (*see the IANA Service Name Registry for details*) and our license-metering module is tested and proven technology.

If you are using a firewall, you may notice communications on port 2799 between the computers that are running Altova products. You are, of course, free to block such traffic between different groups in your organization, as long as you can ensure by other means, that your license agreement is not violated.

Note about certificates

Your Altova application contacts the Altova licensing server (`link.altova.com`) via HTTPS. For this communication, Altova uses a registered SSL certificate. If this certificate is replaced (for example, by your IT department or an external agency), then your Altova application will warn you about the connection being insecure. You could use the replacement certificate to start your Altova application, but you would be doing this at your own risk. If you see a *Non-secure connection* warning message, check the origin of the certificate and consult your IT team (who would be able to decide whether the interception and replacement of the Altova certificate should continue or not).

If your organization needs to use its own certificate (for example, to monitor communication to and from client machines), then we recommend that you install Altova's free license management software, Altova LicenseServer, on your network. Under this setup, client machines can continue to use your organization's certificates, while Altova LicenseServer can be allowed to use the Altova certificate for communication with Altova.

8.3 Altova XBRL Add-in Software License Agreement

- The Altova XBRL Add-in Software License Agreement is available here: https://www.altova.com/legal/xbrl-add-in-eula
- Altova's Privacy Policy is available here: https://www.altova.com/privacy

Index

6

A

B

C

D

E

I

L

R

S

V

X