Template Troubleshooting, Tips and Tricks

11/02/2026 9:26 am MST

General recommendations

Although most of the basic Excel features are supported in templates for Docboss documents, not every feature can be used. There are also some differences when building templates that would be output as PDF (usually various cover pages, section title pages, etc) and those to be used to produce an Excel workbook.

It is recommended to keep the template as simple as possible and sometimes it means that you would have to take some extra steps to make the template work as expected.

Also worth mentioning is that producing PDF output is limited to very basic set of fonts. Generally speaking, there are no limitations to the fonts that can be used in a template, but some exotic fonts or even modifications of common fonts would be replaced to the closest basic version during the generation process. It is recommended to use one or combination of these fonts: Arial, Calibri, Courier New, Georgia, Impact, Lucida Sans Unicode, Microsoft Sans Serif, Tahoma, Times New Roman, or Verdana. To explain the point above, font Arial MT in template will be replaced with Arial during conversion to PDF.

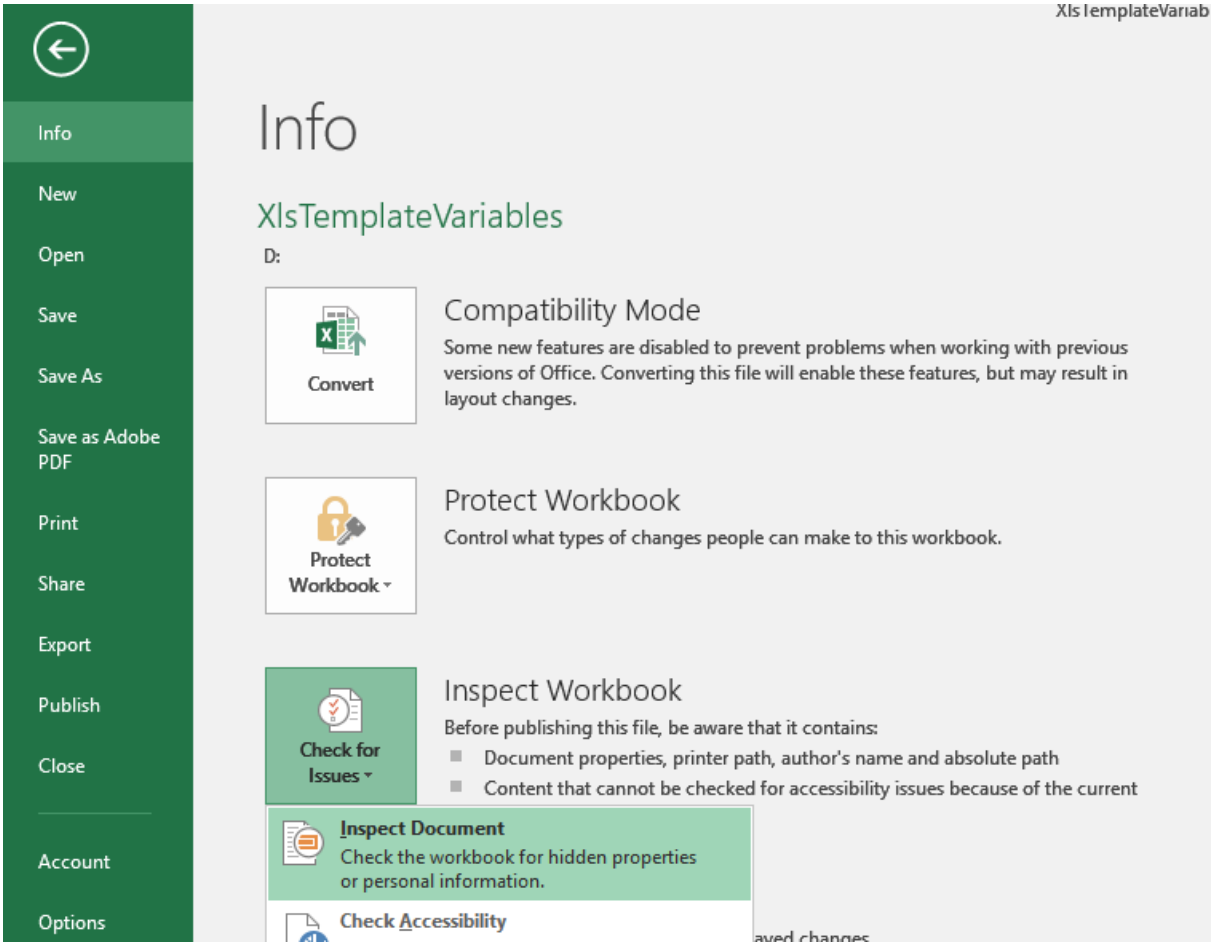
For PDF output templates only the first sheet of the Excel workbook will be used, so it is recommended to remove other sheets to avoid confusion. Excel output templates may have multiple sheets.

XLS vs XLSX

Although there are no known issues with old XLS workbook format, XLSX format is recommended to be used since it is easier for DocBoss to process.

3rd party templates

If you are trying to repurpose an already existing template rather than building it from scratch, it is recommended to sanitize it before uploading to Docboss. It is known that sometimes various metadata in the Excel templates, which are usually invisible to the user, can cause unpredictable errors in Docboss. There is a special tool in Excel called **Inspect Workbook** which allows you to remove such metadata.



After you hit **Inspect Document** option, a popup will show up with a list of various features which you can check and remove if present. None of it besides Header/Footer will be processed by Docboss, so it is recommended to remove the rest. Common case when you must sanitize file is when you get a **Template size is too big** error when trying to upload template.

Images, Excel Drawings and Shapes

It is recommended to preprocess static images before placing them in template. Setting size works fine (with drag and drop or via properties) but such feature as crop will not work properly when the template is output to PDF format. So it's best to crop images outside of the template in any graphical editor or even some viewers that allow you to crop with Ctrl+C\Ctrl+V. So if you don't see an image in the generated document but it was in the template, it's possibly because it is cropped.

Most of the time images in templates are logos, so if possible, it is recommended to use variables instead of static images. This method also makes it possible to use the same template for various vendors, sub-suppliers, etc.

As was mentioned before, more complex things are not supported. Among them are: shapes, drawings and image groups. Shapes and drawings are not very easy to deal with, you will have to convert them in graphical editor to png\jpeg\jpg\gif format and use those instead. One way would be to take a screenshot, paste it to graphical editor, cut off not needed parts and place result in place of shape/drawing. Dealing with image groups, you will need to UnGroup them and if there was only regular images (no shapes or drawings) then they would work as usual. Screenshot trick described above may be used in this case as well.

Footer \ header recommendations

Due to Excel's limitations for footers and headers it's often impossible to compose required footer\header with text capabilities. But there are two ways to work around this.

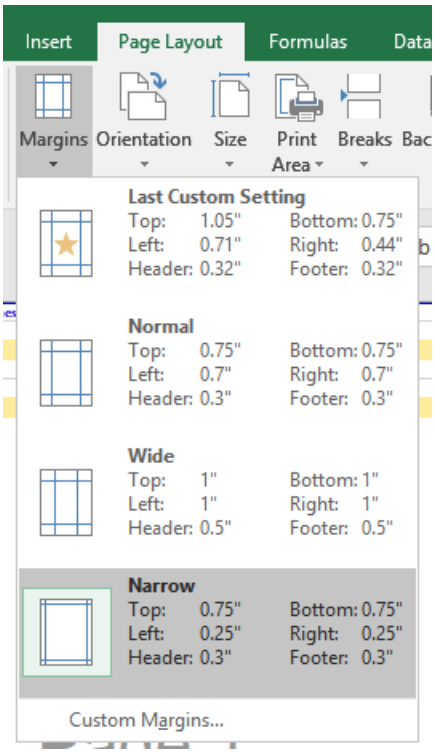
First option is to use image instead of text. Just compose content in graphical editor or even in Excel and use screenshot method described above in Images section and then add footer\header as single image as you would normally do.

Second option would be to not use Excel's footer\header at all, and use first and\or last row instead. Visually there would be no difference in final document after generation, but, compared to first option, this method imposes certain restrictions on the template. Content between first and last row must be static or at least it must have predictable height, otherwise "footer" may be printed on second page.

Depending on the situation you may be forced to use one method rather than the other.

Tutorial - Building Cover Page template from scratch

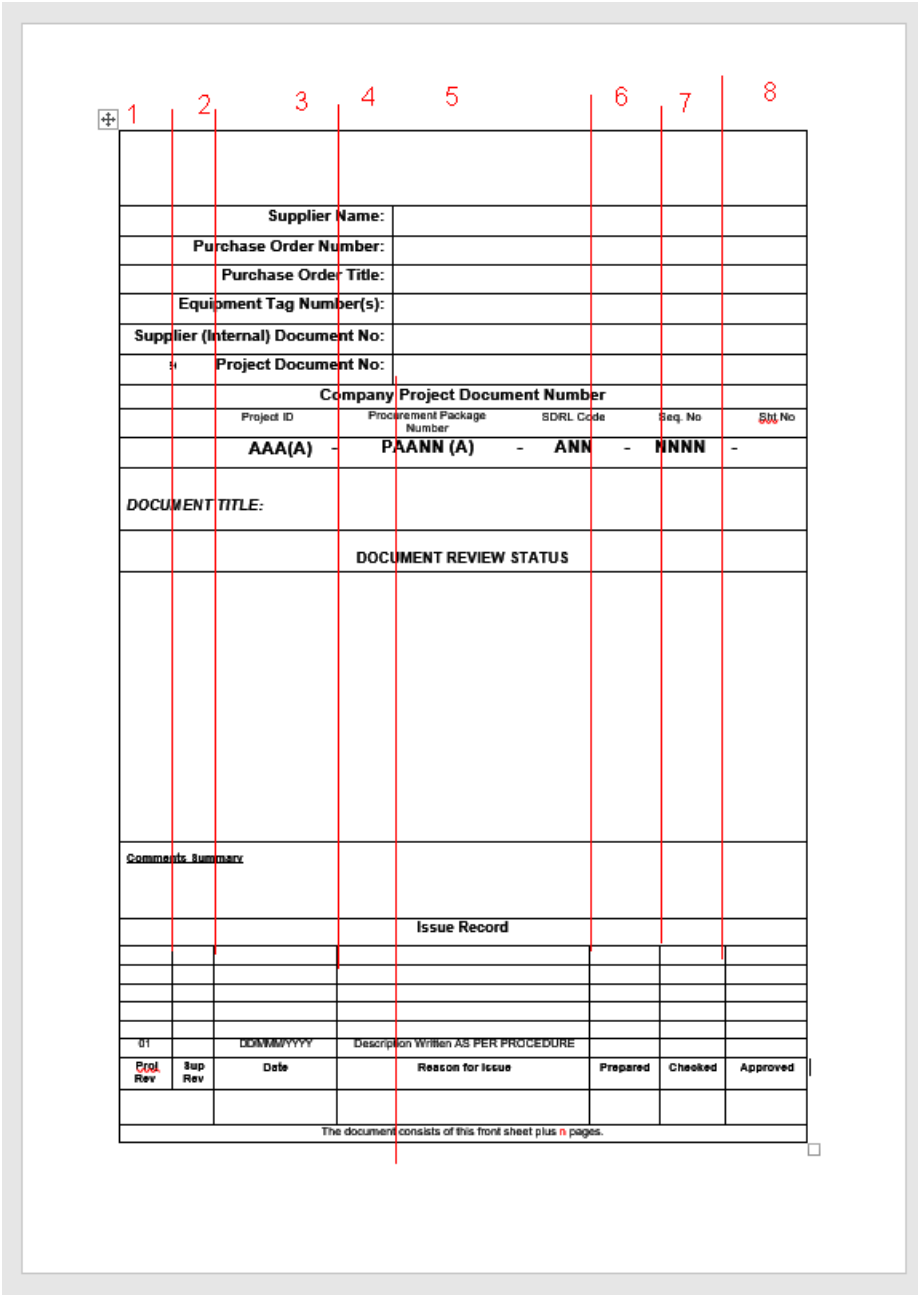
First you would want to set View mode to Page Break Preview or Page Layout mode. To be able to fit more into page and to ensure that whole page will fit during PDF generation it is recommended to set Narrow margins



Be sure to set Page size to match page size set in DocBoss for your Instance.

Next step is to calculate how many rows and columns you will need to fit in page. Do not worry about width and height just yet. Open your sample Cover page in fullscreen mode and moving from left to right, count number of vertical lines you see. Make note that if you want to use different styles for text, each style must be in separate cell, be sure to provision for that when counting columns.

Common Cover page layout is similar to image below: some header, title area in the middle, history table, footer.



In example above, you would need 8 columns in Excel sheet. Stretch them out as needed to fit the page. Same approach must be taken for horizontal lines. Row below REVIEW STATUS may be a single row stretched by height or multiple standard rows, depending on your needs.

After that merge cells where needed, for example columns 1-4 must be merged for Supplier name, and 5-8 for its value. When all cells are merged, add borders and fill static text and variables.

When all of that is done be sure to set Print Area for all cells within borders!

Print Area is used by Docboss as an indicator as to where actual data is located. Without it, the template may not function properly!

Frequently used Cover page elements

Revision table / History table

Almost every cover page includes a revision table, which is most commonly displayed in a **vertical** format.

It contains three rows.

First, <Start_Document_History> - defines configuration of table (sorting, minimal number of rows, etc). In example above there are 6 rows designated for revisions but when you count number of rows you must count them as 3. To account that there has to be at least 6 rows, the minRows parameter must be used <Start_Document_History|minRows=6>. If three blank rows are left as is, they would always be added on top of revision table and wouldn't be populated with data, even when there is more the 6 rows in history records.

By default, history table does not include current revision value. To change that **IncludeCurrent** parameter must be used <Start_Document_History|minRows=6|IncludeCurrent>

Second row of history table contains actual variables that would be replaced with values from document history during generation.

Third row signifies that revision table has ended - <End_Document_History>

During generation, rows with Start_ and End_ variables are removed. Second row is "duplicated" number of times with data or blank rows to fulfill data or minRows parameter as required.

VERTICAL REV HISTORY LAYOUT

<Start_Document_History|StartFrom=Bottom|SortOrder=DESC|minrows=7|IncludeCurrent>			
<Document_History.PrimaryDocRev>	<Document_History.Date>	<Document_History.StageName>	<Document_History.Role1.User_Initials>
<End_Document_History>			
Rev.	**Date**	**Status**	**Submitted By**

If the revision table should be displayed in a **horizontal** format, follow the same three row structure but instead using <CPrint_Document_History> to start the array, and <CEndPrint_Document_History> to end the array. Inside the array, use the Document_History variables as usual.

Sorting the table in ascending order (SortOrder=ASC) will result in the revisions appearing left to right.

If using the horizontal format, the exact number of rows and columns must be present, empty, and within the print area (minRows parameter cannot be used, so the final table structure (rows/columns) must be predefined in the template). Cells in the table **can** be merged across columns, but **cannot** be merged across rows.

HORIZONTAL REV HISTORY LAYOUT

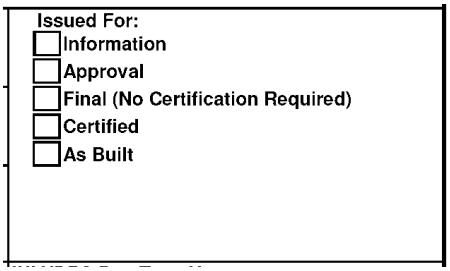
<CPrint_Document_History|IncludeCurrent|SortOrder=ASC>

Rev.	<Document_History.PrimaryDocRev>						
Date	<Document_History.Date>						
Status	<Document_History.StageName>						
Submitted By	<Document_History.Role1.User_Initials>						

<CEndPrint_Document_History>

Lists with checkbox boxes

It is often required to have layout in template as shown below



Although it's tempting to use Excel's shape for those squares, unfortunately it will not work because Docboss cannot work with shapes. It is recommended to form them out of cells and use a borders for those cells. Second option is to use image with square for that purpose.

Merged cells with dynamic size

Often it is hard to predict the amount of data that would be placed in generated document in place of template variable. Normally when a variable is placed in a single cell, you can use Excel's Wrap Text feature which will automatically resize cell by height but often variables are placed in merged cells and unfortunately Wrap Text will not work there. We provisioned *AutoHeight* parameter that can be used with any variable in merged cell like this <Document_Title|AutoHeight> to provide such option. Note! It works correctly **only** with fonts mentioned in General recommendations section. Additional information can be found on Parameters sheet in Variables help file.

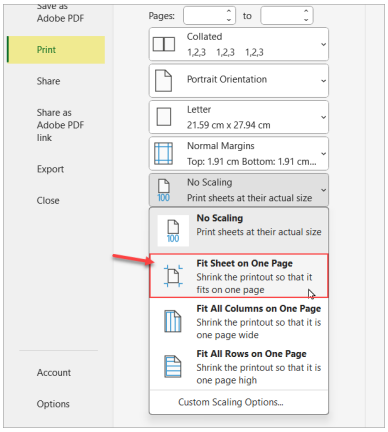
Multi-page templates

Sometimes there is a need to create cover pages that consists of more than one page. This can be achieved by setting up multiple sheets in the template file. DocBoss will treat each sheet as a separate page (e.g., 2 sheets = 2 output pages). Importantly, make sure to define the print area on **each** sheet that should be included in the output (see note).

> **Note:** The sheet that has the cursor on it when the file is saved becomes the "active" sheet. If the file includes a sheet that does not need to be included in the output (e.g., an "instruction" sheet) and the file is saved with this sheet open, you will receive an error when uploading to DocBoss (indicating that the print area has not been identified). In these cases, re-open the file, move your cursor to the correct sheet, save, and re-upload the file again.

Whether or not multiple sheets are used, if generated content cannot fit onto the page, the content will naturally moved to second page. To take control over this process you can use page break functionality present in Excel (Page Layout > Page Setup > Breaks) or use special variable <PageBreak> in template. Difference between them is that variable is applied only during PDF generation process and row with <PageBreak> variable is removed from final document (so everything in the same row is removed as well).

As an alternative, choosing the "Fit Sheet on One Page" scaling option will ensure that all content (included in that sheet) is generated on one page. Keep in mind that the output may shrink as a result.



Tips and Tricks

Dealing with geometrically complex cover pages

If configuration of a template is geometrically complex it might be helpful to set column width to a low value. As a result you will have a very flexible grid of small cells, that can be shaped as required a bit easier. Downside is that you will have to merge those cells a lot, so this method is not recommended for simple templates.

Method to convert Word sample cover page to template

Although this method is not recommended, you can convert Word documents to Excel following these steps. Open the sample file in Word, save it as a Single File Web Page (*.mht, *.mhtml) format. Excel knows how to work with *.mht files so you can open it in Excel now. Open it and save it as *.xlsx. With simple templates it would produce visually similar layout in Excel. This might save some time on building layout, but all other steps (setting variables, setting print area, etc.) must be done manually.

Note! If template was converted using this method, it must be sanitized as described in the **3rd party templates** section of this document.

Dealing with a big tag list

Printing the full list of tags is often required. Excel allows 32767 characters in a single cell but depending on the template layout, font type and character size, the tag list may not fit to A4 or Letter page format. If a cell with a big list of data can not fit to a page it will be cropped. To avoid that it is recommended to use AutoHeight in conjunction with ValuesPerCell parameter (e.g. <Documents.TagList|AutoHeight|ValuesPerCell=1>. In this case each tag value will be printed in a new row.

Note however that the max height of an excel row is 409pts (excel restriction). The combination of column width and size of tag list could result in hidden values if excel cannot make the cell large enough display all tags. In this case, you shoudl icnrease the WIDTH of the column, to allow more tages to display on each line, OR use the COUNT and OFFSET parameters and put the tags in two separate fields. This will be an issue whether you use docboss or not. The data will be available (so if sending an excel file, no trouble, but printing to PDF will cause the tags to seem cut off).

Other Excel Limitations:

Max row height: 409pts
Col width: 255
Characters in header/footer: 255
Characters in cell: 32,767

Correcting hyphenated values splitting between two lines

See this article for steps to change this.

Extending a Tag List to a second page (cover sheets)

There are situations where a coversheet requires a list of tags but if the tag list is too large, it can make a very unseemly output. Some parameters like all_values=ALL will output "All" if all tags are in the list.

In other scenarios, you can limit the number of tags you show on the cover page using the COUNT parameter. Then, create a second worksheet in your cover page. You only want this to appear if there are more than X tags, and you only want to start from the Xth tag. So - the combination of parameters Offset=X|HideSheetIfEmpty will allow you to execute this requirement.