Documentation for

Document Generation for Mendix

appronto

MAKING COMPLEX SIMPLE

Table of Contents

1. Document Generation for Mendix

1.1 Introduction

Document Generation for Mendix (further called DocGen) is a Mendix module that generates output documents based on document templates. For example, if your company uses a specially designed template for all kinds of correspondence the DocGen module will allow you to use this template for your Mendix application document output, so that the output document from your Mendix application matches the corporate branding. The DocGen module is a powerful module for applications that need to generate documents beyond the limitations of the built-in Mendix output generator.

1.2 Prerequisites

Before importing the DocGen module, make sure that the following modules are part of your Mendix application:

- MxModelReflection
- ApprontoLicencer
- CommunityCommons

These modules can be imported from the Mendix App Store. The following documentation will provide information about importing modules in your Mendix application: Import modules from downloaded packages or Import modules from the App store.

note: to run the module in the cloud you will need a license. In development on your local machine the module will run without a license. For a license please contact info@appronto**.nl.*

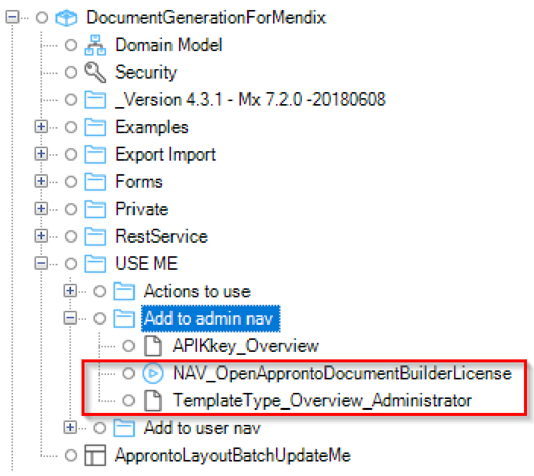
2. Installation

The module works with the following main concepts: the administrator configures **TemplateTypes**. These template types are technical configurations linking Word merge fields to actual data based on your domain model. The end user may create their own **Document Templates** which are Word based templates referring to a TemplateType.

2.1 Installing DocumentGenerationForMendix

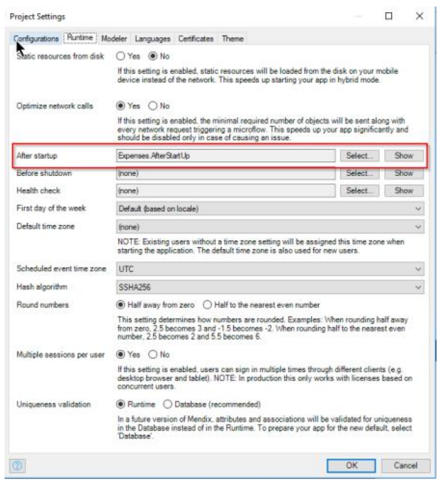
To install do the following:

- Download the DocumentGenerationForMendix module into your project from the app store (Import modules from the App store)
- Add the TemplateManager module role to your administrator user role
- Add the TemplateUser module role to your end user user roles which may use the templates
- Add the `NAV_OpenAppprontoDocumentBuilderLicense` and `TemplateType_Overview_Administrator` to your admin navigation (`USE ME / Add to admin nav`). *See figure 1*
- Add the `ASU_ActivateLicense` to your After Start up microflow (Project settings). *See figure 2 & 3*



DocumentGenerationForMendix
- Domain Model
- Security
- _Version 4.3.1 - Mx 7.2.0 -20180608
- Examples
- Export Import
- Forms
- Private
- RestService
- USE ME
  - Actions to use
  - Add to admin nav
    - APIKkey_Overview
    - NAV_OpenAppprontoDocumentBuilderLicense
    - TemplateType_Overview_Administrator
  - Add to user nav
- AppprontoLayoutBatchUpdateMe

Figure 1



Project Settings

Configurations | Runtime | Modeler | Languages | Certificates | Theme

Static resources from disk: Yes [✓ No]
If this setting is enabled, static resources will be loaded from the disk on your mobile device instead of the network. This speeds up starting your app in hybrid mode.

Optimize network calls: [✓ Yes] No
If this setting is enabled, the minimal required number of objects will be sent along with every network request triggering a microflow. This speeds up your app significantly and should be disabled only in case of causing an issue.

After startup: Expenses.AfterStartUp
Before shutdown: (none)
Health check: (none)
First day of the week: Default (based on locale)
Default time zone: (none)
NOTE: Existing users without a time zone setting will be assigned this time zone when starting the application. The default time zone is also used for new users.

Scheduled event time zone: UTC
Hash algorithm: SSHA256
Round numbers: [✓ Half away from zero] Half to the nearest even number
This setting determines how numbers are rounded. Examples: When rounding half away from zero, 2.5 becomes 3 and -1.5 becomes -2. When rounding half to the nearest even number, 2.5 becomes 2 and 5.5 becomes 6.

Multiple sessions per user: [✓ Yes] No
If this setting is enabled, users can sign in multiple times through different clients (e.g. desktop browser and tablet). NOTE: In production this only works with licenses based on concurrent users.

Uniqueness validation: [✓ Runtime] Database (recommended)
In a future version of Mendix, attributes and associations will be validated for uniqueness in the Database instead of in the Runtime. To prepare your app for the new default, select 'Database'.

Figure 2

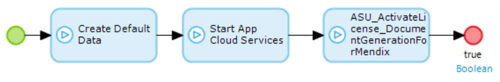


Figure 3

<u>It is highly recommended to avoid making changes in the imported modules (also a best practice in general). Not making changes will make sure that you can install future updates.</u> So, you probably do not want to use the `DocumentTemplate_Overview` (USE ME / Add to user nav), which is just there for demo purposes. The main reason for not using this, is that you want your own security applied to document templates and extend it with own functionalities. To create the optimal solution, follow these steps:

- Create a `CustomDocumentTemplate` object in one of your own modules and inherit from `DocumentGenerationForMendix.DocumentTemplate`. Configure the access rules how you want them.
- Create an overview and new_edit form for this new object.
- Add the DocumentGenerationForMendix / Forms / TemplateSnippet to this new_edit form. This snippet will allow your end users to upload a Word template and see which fields they may use in the template (see figure 4). *Note: do not make changes to TemplateSnippet but copy the content of TemplateSnippet to your created new_edit form if you want to make changes.*

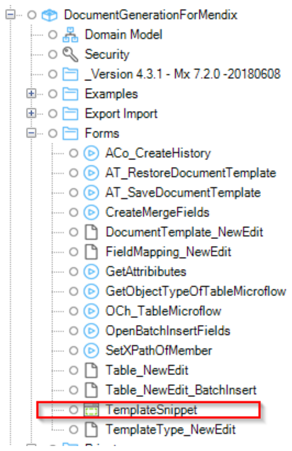

DocumentGenerationForMendix
Domain Model
Security
_Version 4.3.1 - Mx 7.2.0 -20180608
Examples
Export Import
Forms
ACo_CreateHistory
AT_RestoreDocumentTemplate
AT_SaveDocumentTemplate
CreateMergeFields
DocumentTemplate_NewEdit
FieldMapping_NewEdit
GetAttribibutes
GetObjectTypeOfTableMicroflow
OCh_TableMicroflow
OpenBatchInsertFields
SetXPathOfMember
Table_NewEdit
Table_NewEdit_BatchInsert
TemplateSnippet
TemplateType_NewEdit

Figure 4

2.2 Using the action SimpleMerge

The SimpleMerge uses the data that has been retrieved, but when there are multiple records in a list you will need to configure all the fieldnames individually. That is acceptable if you know that there are a few records in the list, but when you don't you probably will miss some data in your output document. To prevent omitting data on the output document, we recommend using the Java Action AdvancedMerge.

Setting up the Mendix application for generating documents with the SimpleMerge

To create a document with data from your Mendix application, first you need to create a new record in the DocumentTemplate object. *In the installation step you created a custom page for this object.*

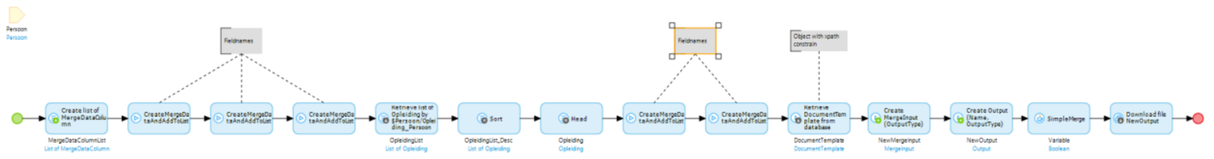
You need to fill in the following:

- Templatename: choose a name for the template
- Upload Word template: here you can upload the Word template

After saving the DocumentTemplate, an action button needs to be created on the overview page of the object containing the data for the document. The microflow behind the button must contain the following actions:

- A list of the data (Mergedata)
- Retrieve of the DocumentTemplate
- Create MergeInput
- Create Output
- SimpleMerge java action
- Optional: download file

Example:


Persoon
Persoon
Fieldnames
Fieldnames
Object with xpath constrain
Create list of MergeDataColumn
CreateMergeDataAndAddToList
CreateMergeDataAndAddToList
CreateMergeDataAndAddToList
Retrieve list of Opleiding by $Persoon/Opleiding_Persoon
Sort
Head
CreateMergeDataAndAddToList
CreateMergeDataAndAddToList
Retrieve DocumentTemplate from database
Create MergeInput (OutputType)
Create Output (Name, OutputType)
SimpleMerge
Download file NewOutput
MergeDataColumnList
List of MergeDataColumn
OpleidingList
List of Opleiding
OpleidingList_Desc
List of Opleiding
Opleiding
Opleiding
DocumentTemplate
DocumentTemplate
NewMergeInput
MergeInput
NewOutput
Output
Variable
Boolean

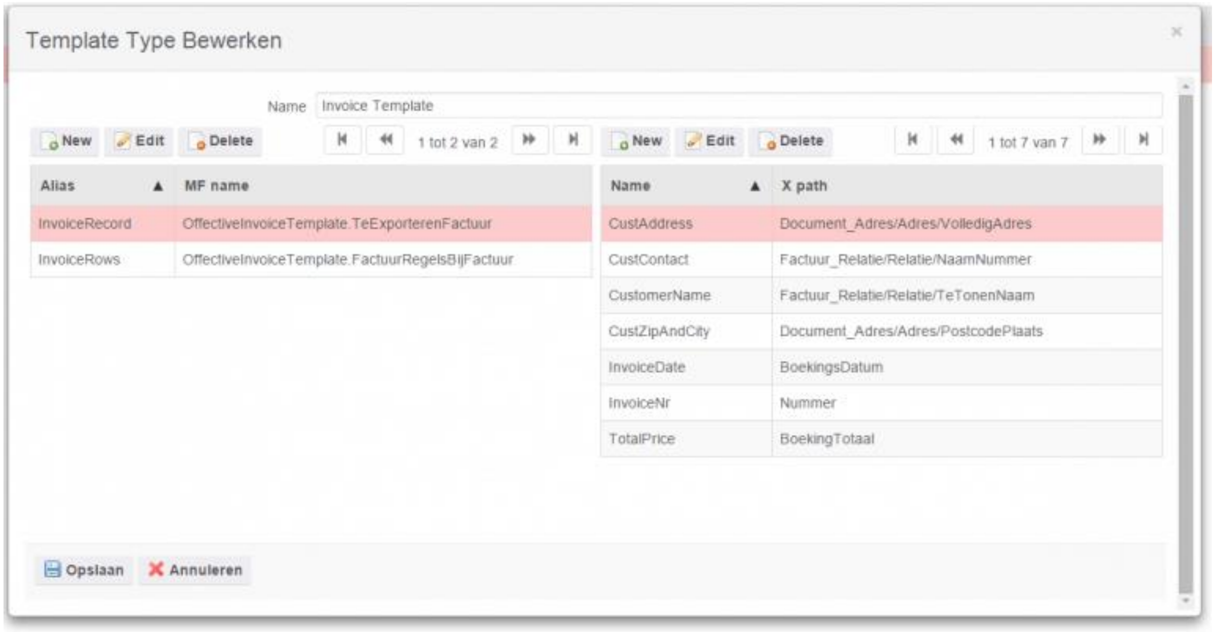
As you can notice we create a list of the MergeDataColumn. In this list we add all the data we need for the document output. We do this with the microflow CreateMergeDataAndAddToList, which can be found at DocumentGenerationforMendix -> USE ME. Of course, you can also create your custom logic and custom object to do this.

2.3 Using the Java action AdvancedMerge

The AdvancedMerge allows the use of records in a list, without naming the fields for each record in the list individually.

Setting up the Mendix application for generating documents with the AdvancedMerge

As an administrator you have to configure the TemplateType(s). After login go to the TemplateType_Overview which you added to the navigation. The example template is shown below:



Name: Invoice Template

Alias	MF name
InvoiceRecord	OffectiveInvoiceTemplate.TeExporterenFactuur
InvoiceRows	OffectiveInvoiceTemplate.FactuurRegelsBijFactuur

Name	X path
CustAddress	Document_Adres/Adres/VolledigAdres
CustContact	Factuur_Relatie/Relatie/NaamNummer
CustomerName	Factuur_Relatie/Relatie/TeTonenNaam
CustZipAndCity	Document_Adres/Adres/PostcodePlaats
InvoiceDate	BoekingsDatum
InvoiceNr	Nummer
TotalPrice	BoekingTotaal

Figure 5

Within a template two things are configured:

- Table
- Merge Field

Table

A table contains all data that has been retrieved by the microflow for that table. The table is shown on the left side of figure 5.

- Alias: this is the alias end users have to use in the Word template indicating a TableStart of TableEnd
- MF Name: This is a reference to a microflow that should return a **List** of records for the table (maybe just one record, but it should be in a list). The microflows are called by the Merging module. So, you don’t call them yourself, just refer to them from the template based on the TableStart:alias construction.

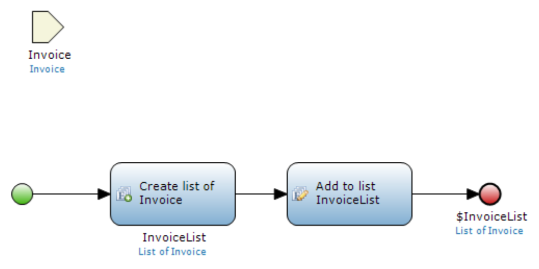
Merge Field

The data within a table can be configured as Merge Fields (FieldMappings). After adding a table just click the table and add Merge Field mappings in the list grid. A field mapping has the following properties:

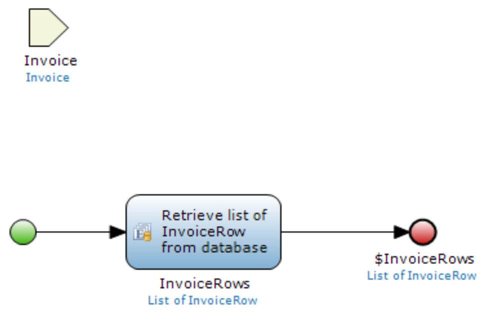
- Name: this is the alias end users must use in the Word template indicating a Merge Field.
- XPath: N-deep path to the domain attribute relative to the Table (use the modeler (data grids) to help you creating this paths)
- Render as HTML: if the attribute contains HTML and you want to render it as HTML click the check box

- Formatting: custom formatting for dates and float/currencies based on Java SimpleDateFormat and DecimalFormat. Examples are: 'dd-MM-yyyy' for datetimes or '#,##0.00' for currencies with thousand separator.

Below the microflows we used for this example:



Left is the microflow for the Invoice table. As you can see it does not retrieve anything from database (you could do that as well of course) but it just passes its input parameter. This is because you have the Invoice Object available within your Export microflow and you want to pass it instead of retrieving it again (which can be hard if there is no context).



This microflow is for the second table and retrieves the InvoiceRow list from database related to the input Invoice.

For a better understanding of why there is a difference between the microflow, a screenshot of an example domain model is added below. *Please note that the attributes differ from figure 5.

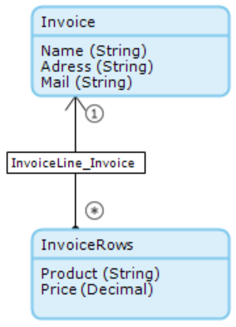
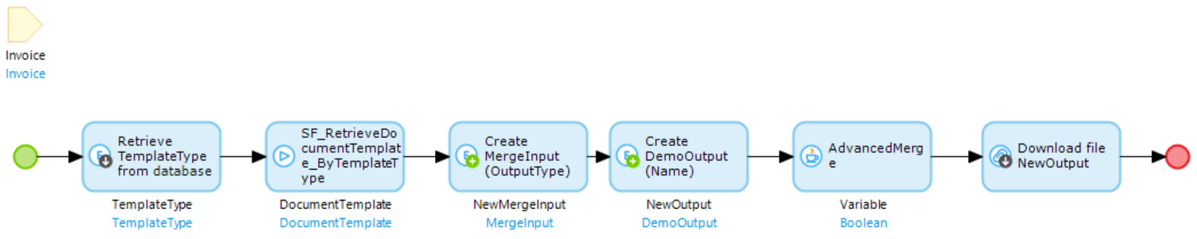


Figure 8

To generate a document with the data from the selected record, a microflow action needs to be created. It will need at least the following actions:

- Retrieve TemplateType
- Retrieve DocumentTemplate
- Create MergeInput
- Create Output
- AdvancedMerge java action

Example:



2.4 Using the Java action AdvancedMergeXML

The AdvancedMergeXML action generates documents based on data from an XML file.

Setting up the Mendix application for generating documents with AdvancedMergeXML

When using AdvancedMerge there is no need to configure a TemplateType because there are no microflows needed to collect the data. In this case the XML file contains the data for the output document. Like the AdvancedMerge a DocumentTemplate needs to be created. Just like with SimpleMerge only the ‘Templatename’ and ‘Upload Word template’ parameters need to be filled in. The structure of the Word Template resembles the one of AdvancedMerge: it must contain a TableStart:Container and TableEnd:Container.

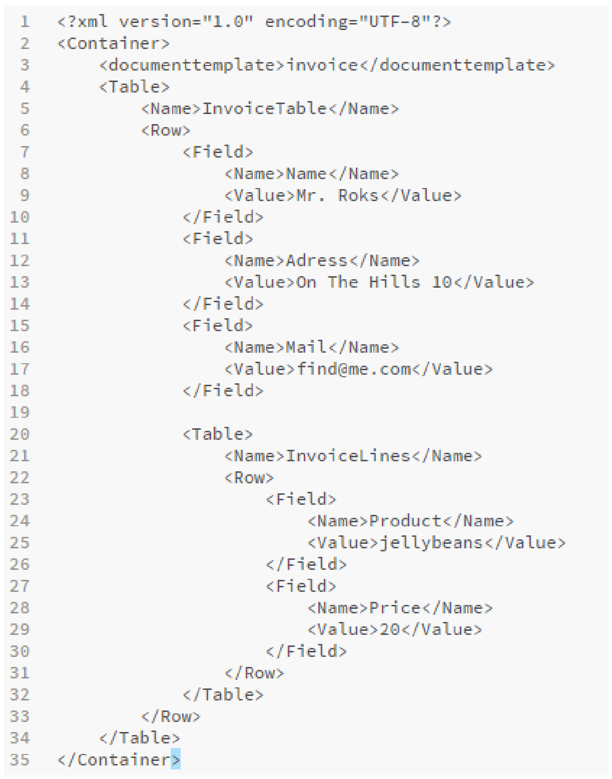
In the XML example below we have used the same domain model structure as for AdvancedMerge. *There is also another example under DocumentGenerationForMendix->Examples*

Notice that the XML also contains <Container> and </Container>.

The XML structure must resemble as if you retrieve data through a microflow as for the AdvancedMerge.

At the beginning of the XML file, you need to specify the DocumentTemplate name in an XML tag.

You see in the example that the tables are identified by a name. It is the same name as we use in the Word template. The fields are also identified with names followed by the values that need to be displayed. The XML structure has an abstract resemblance with the AdvancedMerge way.



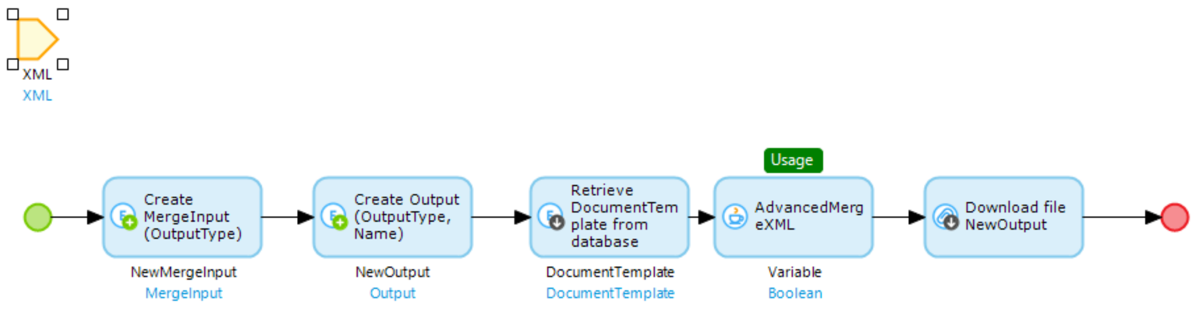
```xml
<?xml version="1.0" encoding="UTF-8"?>
<Container>
    <documenttemplate>invoice</documenttemplate>
    <Table>
        <Name>InvoiceTable</Name>
        <Row>
            <Field>
                <Name>Name</Name>
                <Value>Mr. Roks</Value>
            </Field>
            <Field>
                <Name>Adress</Name>
                <Value>On The Hills 10</Value>
            </Field>
            <Field>
                <Name>Mail</Name>
                <Value>find@me.com</Value>
            </Field>

            <Table>
                <Name>InvoiceLines</Name>
                <Row>
                    <Field>
                        <Name>Product</Name>
                        <Value>jellybeans</Value>
                    </Field>
                    <Field>
                        <Name>Price</Name>
                        <Value>20</Value>
                    </Field>
                </Row>
            </Table>
        </Row>
    </Table>
</Container>
```

You can choose to make an XML entity in your domain model and generalize it with System.FileDocument or create a custom entity to work with. In this example we created an XML entity with the generalization of System.FileDocument. To upload an XML file, a page to upload the XML will be needed, see Import XML Document.

To generate a document with the data within the XML file, a microflow action needs to be created. It will need at least the following actions:

- Create MergeInput
- Create Output
- Retrieve DocumentTemplate
- AdvancedMergeXML java action

Example:



2.5 Setting up the MS Word template inserting the merge field manually

Before you can generate a document with the DocGen module, a document template is necessary. You'll find an example right here: invoice template. As you will notice in the example, the document contains 'TableStart' and 'TableEnd', and are only necessary in the template when the javaAction: AdvancedMerge(XML) is used in the Mendix application. The AdvancedMerge is commonly used when multiple row tables are needed. For example, you need additional data that needs to be retrieved over a one-to-many association. This will result in a list, therefore an extra table with rows is needed to display the data.

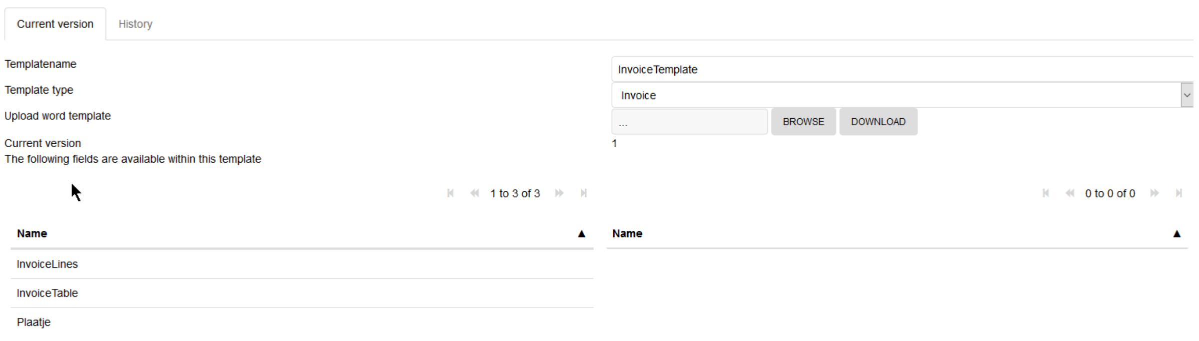
These templates all have the same things in common:

- They contain the concept TableStart:Container. This is always required for technical reasons. Every table is closed with TableEnd:... In this example template we have two tables for retrieving data: Invoice table and InvoiceRows table.
- They contain merge fields. To create new merge fields you can copy existing merge fields. Be sure to edit them in the right way (right click: Edit Field, then edit the field name). Typing directly will just edit the alias and not the technical name, unless you have the fields displayed with their technical names. *OR* follow these steps: insert FieldCode and choose in the Field names column: MergeField. In the Field properties under Field name type in the name of the MergeField.

After the TemplateType and Word template have been set up, a DocumentTemplate record needs to be created. You need to do this on the page DocumentTemplateOverview. In the overview you will see the following:

- Templatename: choose a name for the template
- Template type: here you can choose the template type you configured on the TemplateOverview page. When chosen you will see below the tables related to the template
- Upload word template: choose the Word template for this Document template.

Below an example:


Current version | History

Templatename	InvoiceTemplate
Template type	Invoice
Upload word template	... BROWSE DOWNLOAD
Current version	1

The following fields are available within this template

1 to 3 of 3

Name ▲
InvoiceLines
InvoiceTable
Plaatje

0 to 0 of 0

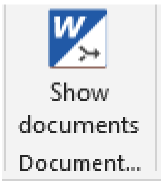
Name ▲

2.6 Using the Document Generation for Mendix MS Word addon

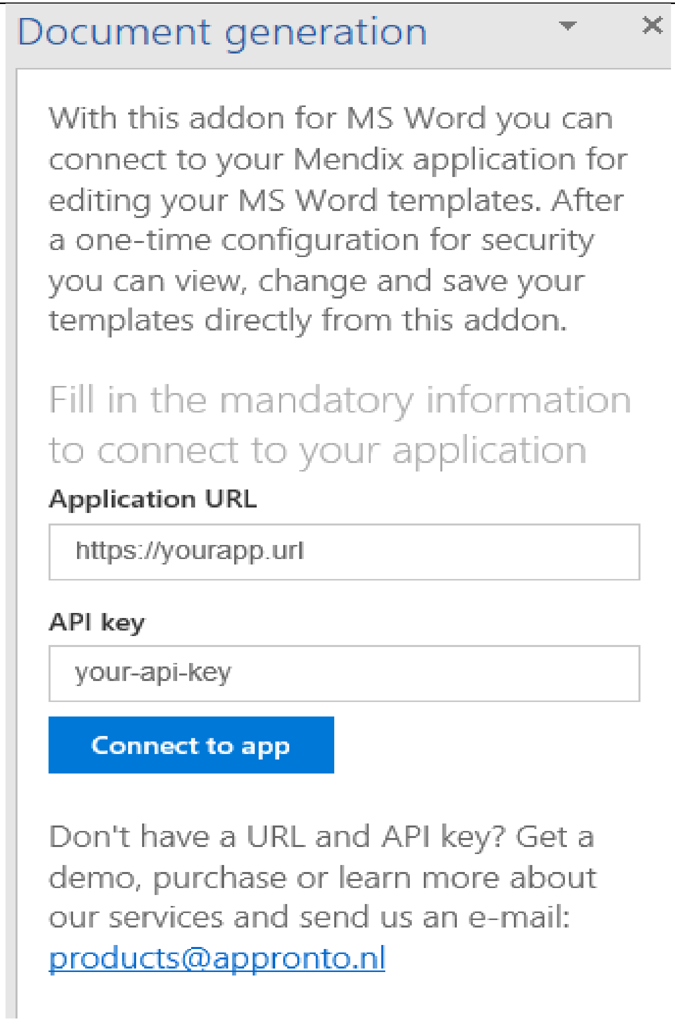
The DocGen addon for MS Word gives the opportunity to create a Word template to drag and drop. Please keep in mind that a TemplateType and a DocumentTemplate already must have been configured. You can upload an empty Word template in the DocumentTemplate.

The following steps must be done to have the addon work properly:

- Add the APIKkey_Overview (can be found at in the DocumentGenerationForMendix -> USE ME -> add to admin nav) to your admin navigation.
- In MS Word you will need the install the add-on: go to the tab Insert -> Add-ins -> Store. A pop up will follow. Type in Mendix in the search field and hit enter. The add-on will be displayed. Click on the Add button. In the Home tab there will be a new icon on the right of the ribbon:



When clicking on the icon the window below will appear on the right side of the screen.

 Document generation With this addon for MS Word you can connect to your Mendix application for editing your MS Word templates. After a one-time configuration for security you can view, change and save your templates directly from this addon. Fill in the mandatory information to connect to your application Application URL: https://yourapp.url API key: your-api-key Connect to app Don't have a URL and API key? Get a demo, purchase or learn more about our services and send us an e-mail: products@appronto.nl	• The application URL can be found under your application Buzz -> Deploy -> Environments • The API Key needs to be generated. You can do this at the APIKkey_Overview and create a new one. Copy this key in the API key field in Word. • Click connect and you will see your configured templates. Now you can start building your Word template.

Once a template has been selected, the tables related to the chosen template will be loaded and shown. By clicking on one of the table names, the fields of the table will be shown (these are the tables and fields configured in the TemplateType Overview Administrator page). By placing the cursor on the preferred position in the document and then clicking on the Fieldname, a mergefield with the name of field will appear in the document. Keep in mind that the Word template must contain the Tablestart:Container and TableEnd:Container. And every Fieldname must be place within the TableStart and TableEnd of the table it belongs to. This can be done easily by clicking on the 'Insert default structure' button.

After the Word template is fully customized with the Fieldnames, just click on the 'Save template to your application' button. This will automatically upload the customized Word template to the right template in your application.

Before use

Disable cloud security in your run configuration <u>(this is only applicable for apps running in the Mendix cloud v3)</u>. This is because the Aspose Module needs certain privileges which are not granted by default by the Mendix cloud. So when you are going to run this in the cloud as well, please file a ticket to ("java.lang.RuntimePermission" "preferences").

To generate an output, there must be a button or microflow action that generates the output. There is an Example microflow in DocumentGenerationForMendix / Examples / GenerateAdvancedMergeExample. To create a very simple test just:

- Add a microflow button to the CustomDocumentTemplate_Overview grid which calls this example microflow. **Note: do not forget to update the GenerateAdvanced-MergeExample with a RootContextObject if you're expecting one in your root document tables!**
- Restart your project
- Login as an end user and create a CustomDocumentTemplate. Upload your MS word template (see attachments for example) and select the corresponding template type.
- Save it and click the Generate button from the overview.
- Getting errors? Enable logging for ApprontoDocumentGeneration log node to trace level.

The microflow that generates the output must contain:

- Create MergeInput parameter. This is needed for the AdvancedMerge Java Action. You have to set the desired output type (Word, PDF, HTML, BMP, and so on)
- Retrieve a template.
- Call the AdvancedMerge Java action. All parameters are pretty clear from the context. The RootContextObject is not required but if empty nothing will be passed to Root Table microflows.

3. How to's

How to import the DocumentGenerationForMendix module
Please see Installing the DocumentGenerationForMendix module.

How to create a template for the DocGen module
To create a template, just create a Word document. For technical reasons, the document needs to contain the MergeFields: <TableStart:Container> & <TableEnd:Container>

See the invoice_template for an example.

How to select the data for the document output
In the Template Overview page the microflow that retrieves the data must be imported in the left column. When the microflow is selected in the left column, in the right column the data from the microflow needs to be mapped to the fields. The name of a field must be exactly the same as the name of the merge field within the Word template.

How to set that the right data is used for the document generator action
In the microflow that will be called when generating a document, there should be a Retrieve object action. In this Retrieve action the name of the template should be set.

How to set the output type
In the microflow call when a document needs to be generated, there must be a Create object (MergeInput) action and a Create object (Output) action. Within the create MergeInput action you can set the value for the member output. Also, in the create Output object the value of the member output needs to be set.

How to render an image from the Mendix application to the generated document
One of the methods is to convert the image to a base64 string after the retrieve. We recommend using the java action (Base64EncodeFile) from the Community Commons module. The return variable can be used as a value of an attribute that will be used for a merge field. Your merge field should look like this «Image:*Fieldname*». The microflow should look something like:

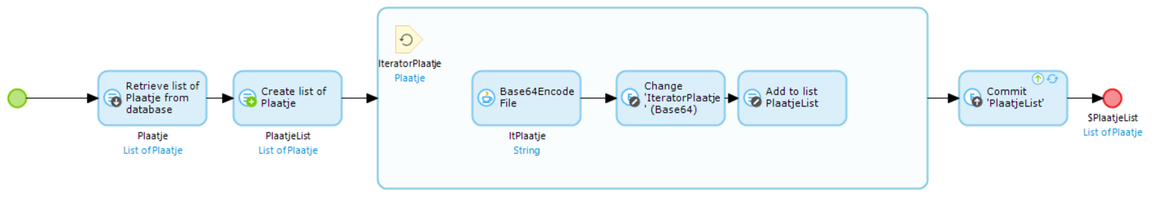


Figure 9

Another method is to use the 'Alias: _Image_'. For this method you will need to refresh your MXReflex with the module System checked. The microflow that needs to be used in the table should look something like this:

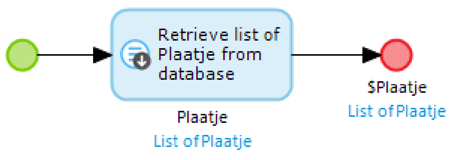


Figure 10

The Fieldname should be set as _Image_ with as selected X path ‘Contents’. The Mergefield should look like «Image:_Image_».

4. FAQ

Do I need to use the TableStart: Container and TableEnd:Container when I use SimpleMerge?
When using SimpleMerge, you don't need to use the TableStart:Container and TableEnd:Container.

Is it possible to show all the data in a list on the output document with SimpleMerge?
Yes, but you will have to manually configure the exact amount of merge fields as the amount of records in the list. We recommend using AdvancedMerge, as you can work with tables and rows.

Can I use any Word template with DocumentGenerationForMendix ?
Yes, but make sure the merge fields in the Word template correspond with the fields in your TemplateType.

How do I delete the Aspose watermark?
With an active license the watermark will disappear. Please contact info@**appronto**.nl for a license.

Which java action should be used when data is needed from different entities?
This depends on the preferred outcome. If the output only needs to display single records, SimpleMerge will do the job. If the output should display all records from a list, then the AdvancedMerge should be used.

When a mergefield is empty, the line in the document is still visible. Is it possible to not show it when empty?
The IF ELSE statement can determine whether a mergefield is shown and can also determine if a line needs to be added. See also If Else Mergefield.

Can I use the API key which I generated in Sprintr for DocumentGenerationForMendix MS Word addon?
No, follow the steps described in Setting up the MS Word template using the Document Generator 4 Mendix MS Word addon to generate a API key.