Microsoft Office Word document (DOCX)

Report Wizard supports most *DOCX* features. You can place the *VTL* codes inside core (properties) and content of any *DOCX* file. All syntax usable in *RTF* can also be used in *DOCX*.

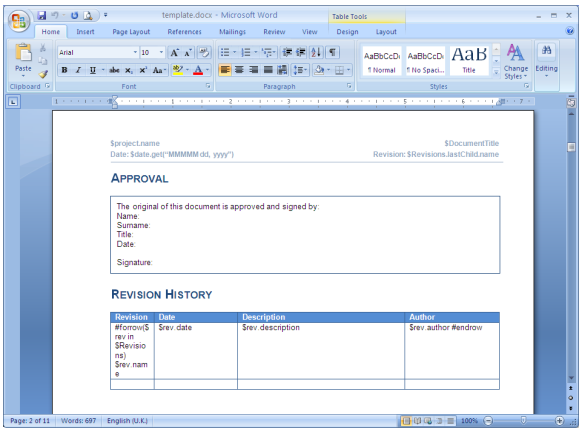

Figure 1: A Sample of DOCX Converted from an RTF Document.

Limitations when used in Microsoft Office Word document

You cannot use multi-line statements in different objects. If you try to use them in *DOCX*, an error message will open. See the following example.

```
1  Using directive in different object
2  #forpage($c in $Class)
       $c.name
       #endpage
3
```

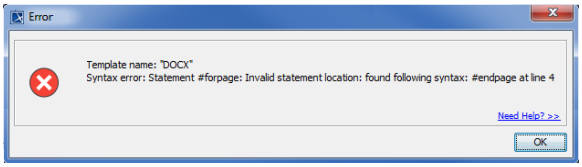

Figure 2: The Error Message of Invalid Usage of the Multi-line Statements in DOCX.

Creating data for multiple columns

#forcol is used for creating data for multiple columns in a row. This statement must be defined in table and can be used in conjunction with the **#forrow** statement. For example:

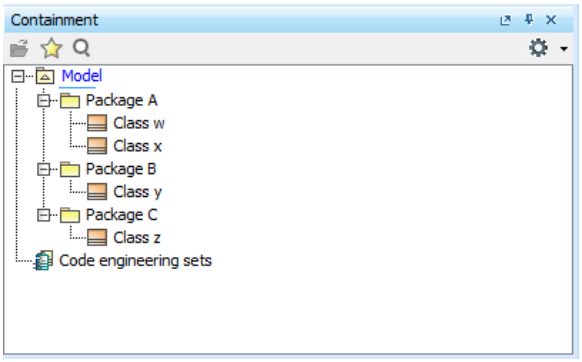

Figure 3: Sample elements in MagicDraw Containment tree.

Template Code:

Name	#forcol($p in $Package)$p.name#endcol

Name	Package A	Package C	Package B

#forcol may create different number of columns in each row. See the example below.

Template Code:

#forrow($p in $sPackage) $p.name	#forcol($o in $sorter.sort($p.ownedElement)) $o.name#endcol#endrow

Output:

Package A	Class w	Class x
Package C	Class z	
Package B	Class y	

#forcol must be defined as the first statement in a column because it is parsed and processed before other statements in the column except **#forrow**. See the example below.

Template Code

Name	#set($p = $Package) #forcol($ap in $p) $ap.name#endcol

Above code will be parsed to

Name	#forcol($ap in $p) #set($p = $Package) $ap.name#endcol

Output:

Name

So, the template code should be:

Template Code:

Name#set($p = $Package)	#forcol($ap in $p) $ap.name#endcol

Output:

Name	Package A	Package C	Package B

#forcol will create dynamic columns in a row. This directive does not create columns for the whole table. See the example below.

Template Code:

Name	#forcol($p in $Package)$p.name#endcol

Output:

Name	Package A	Package C	Package B

Docx supports at most 63 columns per row (Figure 4).

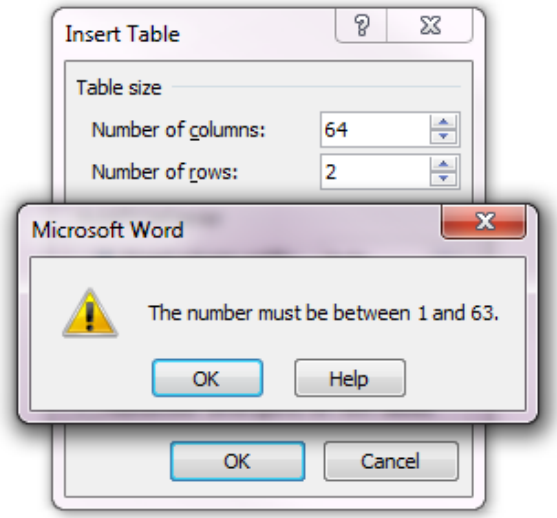

Figure 4: Docx supports at most 63 columns per row.

Creating merged column horizontally for DOCX

ReportWizard provides utility functions for creating table properties. The functions of this module are accessible from templates through **$tableprop**. In this version, ReportWizard provides only **$tableprop.mergeColumns()** to create merged columns horizontally in *DOCX* template.

This statement must be defined in table.

1. $tableprop.mergeColumns(int number)

Merging columns in specified number

Where the parameter is:

- number - the number of columns to merge. This value has to be of type integer and start from 2.

For example:

Template Code:

Project	$tableprop.mergeColumns($Package.size())Packages
	#forcol($p in $Package)$p.name#endcol

Project	Packages		
	Package A	Package C	Package B

2. $tableprop.mergeColumns(String stringNumber)

Merging columns in specified number as String

Where the parameter is:

- stringNumber - the string value of number of columns to be merged. This value has to be of type integer string and start from “2”.

For example:

Template Code:

Project	$tableprop.mergeColumns(“3”) Packages
	#forcol($p in $Package)$p.name#endcol

Output:

Project	Packages		
	Package A	Package C	Package B

$table.mergeColumns() must be defined as the first statement in the column because it is parsed and processed before other statements in column expect **#forrow** and **#forcol.** See the example below.

Template Code:

Project	#set($gValue = $Package.size() + 1)$tableprop.mergeColumns($gValue)Packages	
	#forcol($p in $Package) $p.name#endcol	No Package

Above code will be parsed to

Project	$tableprop.mergeColumns($gValue) #set($gValue = $Package.size() + 1)Packages	
	#forcol($p in $Package) $p.name#endcol	No Package

Output:

Project	Packages			
	Package A	Package C	Package B	No Package

So, the template code should be:

Template Code:

Project#set($gValue = $Package.size() + 1)	$tableprop.mergeColumns($gValue)Packages	
	#forcol($p in $Package)$p.name#endcol	No Package

Project	Packages			
	Package A	Package C	Package B	No Package

$tableprop.mergecolumns() will make a column to be a merged column, it does not change the number of columns in a row. See the following example.

Template Code:

$tableprop.mergeColumns(“5”)Packages		
#forcol($p in $Package)$p.name#endcol		

Output:

Packages						
Package A	Package C	Package B				

The number of columns in the first row is still 3.