WordWriter - Documentation

Before you start:

This is a documentation, how to use the WordWriter Library. WordWriter is an API between Java and Word. So you can use it to generate a .docx Word-Document from your code. It also provides a Placeholder-Substitution which channges a documents content at runtime.

Provided Elements for building Word-Documents:

* Paragraph (with Shortcuts)
* Table
* Image
* Header & Footer
* Numberings
* **Set Default-Values in properties-File**

*-config.properties*

*Don’t change the red marked Values, you may get an Error otherwise!*

#WordWriter Properties  
  
TemplateDocumentName=template.docx  
AfterSubstitutionDocumentName=substituted.docx  
DotxEnding=dotx  
DocxURI=/word/document.xml  
PageBreak=page  
  
#Relationship Properties  
RelationshipIdString=rId  
RelationshipBuilderStartId=3  
  
#List Properties  
SpacingSetLine=220  
SpacingSetLineRule=auto  
  
#Table Properties  
WWTableContentSize=20  
WWTableHeaderRows=1  
WWTableStyle=TableGrid  
ShadingSetColor=auto  
ShadingHeaderFill=FFFFFF  
ShadingRowFill=D8D8D8  
RsidR=00010588  
RsidRDefault=00010588  
VMergeRestart=restart  
VMergeContinue=continue  
HMerge=true  
  
#Paragraph Properties  
Orientation=center  
Font=U001TLig  
WWParagraphSize=20  
  
#Header-Footer Properties  
HeaderStyle=Kopfzeile  
OrientationRight=right  
OrientationCenter=center  
RelativeTo=margin  
Leader=none  
  
#Error Messages  
WpMLLoadError=Beim laden des files ist ein Problem aufgetreten!  
UnMarshallError=Beim unmarshalling ist ein Fehler aufgetreten!  
URIError=Beim Verändern des Content-Types ist ein Fehler aufgetreten!  
NoValueInMapError=Kein Wert für Platzhalter:  
MissingKeyError=Kein Key in Map für Platzhalter:  
  
#Border  
BorderValue=single  
BorderSize=4  
BorderSpace=1  
BorderColor=auto  
  
#Regex  
BoldRegex=<[[u|i]?]\*b[[u|i]?]\*>(.\*?)</[[u|i]?]\*b[[u|i]?]\*>  
ItalicRegex=<[[u|b]?]\*i[[u|b]?]\*>(.\*?)</[[u|b]?]\*i[[u|b]?]\*>  
UnderlineRegex=<[[b|i]?]\*u[[b|i]?]\*>(.\*?)</[[b|i]?]\*u[[b|i]?]\*>

* **Create the Document**

WordWriter word = new WordWriter();

When a WordWriter-Object is created, the WWDefaults-Class gets refreshed. You can change the content of the config.properties File to set your prefered Defaults.

* **Creating a ParagraphFormat**

You can create Instances of this class for several Formats in your Document.

WWParagraphFormat arial12 = new WWParagraphFormat(12, "Arial", "left");  
WWParagraphFormat calibri22 = new WWParagraphFormat(22, "Calibri", "left");

* **Create a Paragraph**

With Parameters

P par = word.createParagraph("Cicero Consulting", 30, "Arial", "center", 220, 1);

WordWriter.createParagraph(String content, Integer size, String font, String orientation, Integer spacing, Integer numId);

Spacing is the distance betweeen two paragraphs, which are one below the other.

The last parameter is called numId. You use it, if you want a Paragraph to be part of a list. Several Paragraphs with the same numId (no matter where in the documet) contain to the same list.

With Format

Instead of typing all the parameters everytime you want to create a Paragraph, you can use the WWParagraphFormat as one parameter.

WWParagraphFormat(Integer size, String font, String orientation, Integer spacing, Integer numId)

WWParagraphFormat arial30 = new WWParagraphFormat(30, "Arial", "center"", 220, null);

P par = word.createParagraph("Cicero Consulting", arial30);

Shortcuts

If you want some parts of the paragraph to be bolt, underlined or italic, use the provided shortcuts.

P par = word.createParagraph("<biu>Cicero Consulting</biu>", 30, "Arial", "center", 1);

<b>some text</b> 🡪 **some text**

<u>some text</u> 🡪 some text

<i>some text</i> 🡪 *some text*

<bu>some text</bu> 🡪 **some text**

<bi>some text</bi> 🡪 ***some text***

<iu>some text</iu> 🡪 *some text*

<biu>some text</biu> 🡪 ***some text***

Adding a Frame to Paragraph

test.addHLine(par, "top");  
test.addHLine(par, "bottom");  
test.addHLine(par, "left");  
test.addHLine(par, "right");

MS-Word sometimes has some problems with these frames. It could be the case that your frame isn’t visible in the Document, although it’s correctly nested inside the code.

* **Add a Table to the Document**

WWTable t = word.createWWtable(4, 4);

Set TableGrid

List<Integer> integers = Arrays.*asList*(1000, 1500, 1000, 2000);  
t.createTableGrid(integers);

You don’t have to set the tableGrid by yourself, you just have to create it. It gets automatically set when the table gets saved.

Shading

You can use shading, to get a better overview of the table. Select the rows/cols, you want to shade, put them into a list and use it as parameter. The selected rows/cols are getting shaded then.

You can use row-shading and col-shading.

List<Integer> shadings = Arrays.*asList*(2, 4, 5);  
t.setColShading(shadings);

t.setRowShading(shadings);

Add Content

//-- String in Table-Cell

t.addCell(0, 0, "hallo");

t.addCell(0, 2, "Welt", "left");

t.addCell(0, 1, "meine", null, null);  
t.addCell(0, 2, "Welt", null, null, "left");

//-- Table in Table-Cell

t.addCell(1, 2, Tbl);

t.addCell(1, 2, Tbl, "center");

t.addCell(1, 2, Tbl, 2, null);

t.addCell(1, 2, Tbl, null, 3, "center");

//-- Paragraph in Table-Cell

t.addCell(1, 2, P);

t.addCell(1, 2, P, "right");

t.addCell(1, 2, P, 2, null);

t.addCell(1, 2, P, null, null, "right");

t.addCell(Integer row, Integer col, **String content**, Integer vmerge, Integer hmerge, String orientation);

t.addCell(Integer row, Integer col, **Tbl content**, Integer vmerge, Integer hmerge, String orientation);

t.addCell(Integer row, Integer col, **P content**, Integer vmerge, Integer hmerge, String orientation);

Integers vMerge & hMerge: number of cells which should get merged (at least 2 to merge one cell)

If you want to merge cells **vertically**, you don’t have to add those cells, which are getting merged.

If you add them anyway, they’re not visible.

If you want to merge cells **horizontally**, you must not add those cells, which are getting merged, otherwise your table will expand in it’s width.

It’s also possible to add several objects into one table-cell. They’re getting viewed one below the other.

*Shortcuts are also provided for generating table-Content.*

Borders

*You can set default values for every border in the config.properties-file. If you want to specify the borders of one cell use the provided function for setting borders. The first two parameters are indexes which point to the cell you want to change.*

t.setTcBorders(3, 4, "single", "single", "double", "single");

Cast WWTable to docx4j Table

Tbl x = t.saveTable();

Add the Table to the Document

word.getMainPart().addObject(x);

* **Add a LineBreak into the Document**

word.addLineBreak();

If you want a lineBreak within a Paragraph

P einzelvertrag = test.createParagraph("Einzelvertrag", 17, "U001TLig", "left");  
einzelvertrag.getContent().add(test.getLineBreak());

* **Add a PageBreak into the Document**

word.addPageBreak();

* **Add a horizontal Line to the Document**

If you want to devide some parts of your Document, you can use a horizontal Line between them.

test.addHorizontalLine();

* **Add a numbered List to the Document**

List<String> l = Arrays.*asList*("hallo", "das", "ist", "ein", "test");  
word.add(l, "1");

This method is not recommendet. If you want to make a list, use the numId parameter when creating it. This allows you to add content between enumerations.

* **Add a Header or Footer to the Document**

*Add String as Header or Footer (Orientation = left)*

word.addHeader("Document-Title");  
word.addFooter("11.07.2016");

*Create a Paragraph as Header or Footer*

List<String> l1 = Arrays.*asList*("Datum", "Name", "Ort");  
P header = word.createHeaderOrFooterParagraph(l1);

P footer = word.createHeaderOrFooterParagraph(l1, arial30);

You can also give a WWParagraphFormat as parameter to adjust format-settings.

*Add the Paragraph as Header or Footer*

test.addHeader(header);  
test.addFooter(footer);

Remind that this Creation of a header- or a footer-Paragraph is static. The first Element is stored left, the second in the center and the third one right. That means that you have to add at least 3 Elements to the List.

Example:

If you want the visible parts of the header to be stored in center and on the right, you have to remind that the method is static. In this case you have to add a List with an empty string as first element. The second element will be in the middle while the third will be on the right side.

* **Add an image to the Document**

For adding images, use the WWImage-Class.

There ara a few possibilities how you can add an image to your document. The simplest way is to store it as an inline Object.

The inline image doesn’t overlap any other Objects in the document. It just needs much space.

WWImage image = new WWImage("image.jpg", "", 0.2, test.getMyWord());

If you want the image to be overlapping (in front or behind the text) you have to make the inline anchor.

image.setAnchor(true);

Now you can choose if you want the image to be behind the text or in front of it.

image.setImageIsBehindDocument(false);

Positioning

There are two ways how you can influence the possitioning of the image. First is to set the Alingments.

image.sethAllignment(STAlignH.*RIGHT*);  
image.setvAllignment(STAlignV.*TOP*);

The second possibility is more accurate. By setting the Offsets in pixel\* the image can be placed in a more speciffic way.

Remind that setting offsets and alignments is only available for anchor images.

Text-Wrapping

If you want the image to be inside or Paragraph (wrapped by some text) you have to set the Wrapping to true.

image.setWrapping(true);

After setting the formatings, use the buildImage() function, and store it’s return-value inside the Document’s manipart.

test.getMainPart().addObject(image.buildImage());

* **Placeholder-Substitution**

You have also the possibility to generate a document with placeholders, which are getting replaced at runtime. To use shortcuts use two hashes around the keyword.

##customer\_surname##

You can also use an existing template with placeholders.

Placeholder-Substitution-Classs provides substitution for .docx and .dotx documents.

The placeholder-substitution works with a Hash-Map where every placeholder is a key and has got it’s value. So you have to build your customized Hash-Map to give it as a parameter for the placeholder-substitution.

Use the placeholder-substitution like this:

Map map = new HashMap();  
map.put("##MIETER\_X-first\_name-surname##", "Dieter Schoass");  
map.put("##MIETER\_X-street-house\_no##", "Peterwardeinstrasse 8");  
map.put("##MIETER\_X-post\_code-city##", "9073 Viktring");  
map.put("##amt\_annual\_old##", "1.234,00");  
map.put("##amt\_used##", "123,14");  
map.put("##request\_no##", "01923");  
map.put("##MIETOBJEKT-street-house\_no##", "Jauerburggasse 17");  
map.put("##MIETOBJEKT-post\_code-city##", "8010 Graz");  
map.put("##MIETER\_X-salutation-title-surname##", "Manuel Hess");  
map.put("##time\_frame\_used##", "21.07.2008-12.01.2016");  
map.put("##time\_frame\_unused##", "12.01.2016-25.07.2016");  
map.put("##amt\_annual\_reduced##", "1.234,00");  
map.put("##amt\_service##", "499,00");  
map.put("##amt\_annual\_new##", "249,00");  
map.put("##amt\_credit\_note##", "29,00");  
map.put("##nav\_user##", "der Inder");  
map.put("##amt\_unused##", "129,00");  
map.put("##rent\_deposit\_reduced##", "Tarif E46");  
map.put("##rent\_deposit##", "Tarif B32");  
map.put("##doc\_date##", "25.07.2016");  
  
try {

WWPlaceholderSubstitution.*renamePlaceholdersFromHashMap*("tochange.dotx", "changed.docx", map);  
   
} catch (IOException e) {  
 e.printStackTrace();  
}