

Python - Docx Edition

Technical documentation

Disclaimer:

Due to constant changes in information technology systems and applications, the information contained in this site is provided on an "as is" basis with no guarantees of completeness, accuracy, usefulness or timeliness.

1. Introduction

python-docx is a Python library for creating and updating Microsoft Word (.docx) files.

1. Installation

To be able to use python-docx, you need to install the module. First, update pip:

pip install --upgrade pip setuptools

If you're using a proxy, then add the option --proxy, for example:

pip --proxy http://${PROXY\_IP}:${PORT} install --upgrade pip

Now you can install the module:

pip install python-docx

If you face an error like the one hereunder, make sure that you updated the setuptools module.

error: can't copy 'docx/templates/default-docx-template':  
doesn't exist or not a regular file

1. Word Style Template
2. Usage
   1. Document Creation

To create a document, first import the python-docx module:

from docx import Document

To open an existing presentation, do:

document = Document(docx\_template)

Or a new one:

document = Document()

* 1. Text in a Document

To insert a text in a document, do:

document.add\_paragraph('Hello World')

* 1. Styles and Layout

To apply a specific style for a text, do:

document.add\_paragraph(text, style)

For example:

document.add\_paragraph('The Lord of the Rings', 'Title')  
document.add\_paragraph('The Fellowship of the Ring', 'Subtitle')  
document.add\_paragraph('Prologue', 'Heading 1')  
document.add\_paragraph('Concerning Hobbits', 'Heading 2')

content = """This book is largely concerned with Hobbits, and  
from its pages a reader may discover much of their character and  
a little of their history."""

document.add\_paragraph(content, 'Normal')

If you want to create a new style, first import the necessary libraries:

from docx.shared import Pt, RGBColor  
from docx.enum.style import WD\_STYLE\_TYPE

Now you can create the new style, start by declaring it, giving it a name:

document.styles.add\_style('Titulo', WD\_STYLE\_TYPE.PARAGRAPH)

Then set it the way you want:

title\_style = document.styles['Titulo']  
title\_style.font.name = "Arial"  
title\_style.font.size = Pt(28)  
title\_style.font.color.rgb = RGBColor(0x00, 0x00, 0x00)  
title\_style.paragraph\_format.space\_after = Pt(28)

To add a page break, do:

document.add\_page\_break()

* 1. Table in a Document

To create a new table, do:

table = document.add\_table(r, c, s)

With:

* r the number of rows in the table
* c the number of columns in the table
* s the style of the table (optional)

To add content in a cell, do:

table.cell(row\_id, col\_id).text = 'Firstname'

For example, if I wanted to create a table like the following one:

+-----------+----------+  
| Firstname | Lastname |  
+-----------+----------+  
| Bilbo | Baggins |  
+-----------+----------+

I would do:

table = document.add\_table(2, 2, 'Table Grid')  
table.cell(0, 0).text = 'Firstname'  
table.cell(0, 1).text = 'Lastname'  
table.cell(1, 0).text = "Bilbo"  
table.cell(1, 1).text = "Baggins"

To set the background color to a specific cell, import the necessary libraries:

from docx.oxml.shared import OxmlElement, qn

Add the following function:

def shade\_cells(cell, shade):  
 tcPr = cell.\_tc.get\_or\_add\_tcPr()  
 tcVAlign = OxmlElement("w:shd")  
 tcVAlign.set(qn("w:fill"), shade)  
 tcPr.append(tcVAlign)

Now, you can use the function like:

shade\_cells(table.cell(x, y), "FF6666")

* 1. Picture in a Document

To add a picture in a document, do:

document.add\_picture(p, w)

With:

* p the picture file you want to insert
* w the width of the picture

You can also add the height of the picture, instead or in addition to the width of the picture. However, if you add both the width and the height, the proportions of the picture won't be kept.

If you want to use inches for the width or height of the picture, import the library:

from docx.shared import Inches

Then you can declare the width like:

graph\_width = Inches(6.0)

* 1. Save the Document

To save the document, do:

document.save(docx\_output)

1. Sources

* Canny, S. python-docx. Retrived from https://python-docx.readthedocs.io/en/latest/#

Authors

Quentin Petit (editor)

Luxembourg

EMail: <petit.quent@gmail.com>