indexnext |previous |python-docx 0.8.6 documentation »

Working with Documents

python-docx allows you to create new documents as well as make changes to existing ones. Actually, it only lets you make changes to existing documents; it’s just that if you start with a document that doesn’t have any content, it might feel at first like you’re creating one from scratch.

This characteristic is a powerful one. A lot of how a document looks is determined by the parts that are left when you delete all the content. Things like styles and page headers and footers are contained separately from the main content, allowing you to place a good deal of customization in your starting document that then appears in the document you produce.

Let’s walk through the steps to create a document one example at a time, starting with two of the main things you can do with a document, open it and save it.

Opening a document

The simplest way to get started is to open a new document without specifying a file to open:

from docx import Document

document = Document()

document.save('test.docx')

This creates a new document from the built-in default template and saves it unchanged to a file named ‘test.docx’. The so-called “default template” is actually just a Word file having no content, stored with the installed python-docx package. It’s roughly the same as you get by picking the Word Document template after selecting Word’s File > New from Template... menu item.

REALLY opening a document

If you want more control over the final document, or if you want to change an existing document, you need to open one with a filename:

document = Document('existing-document-file.docx')

document.save('new-file-name.docx')

Things to note:

You can open any Word 2007 or later file this way (.doc files from Word 2003 and earlier won’t work). While you might not be able to manipulate all the contents yet, whatever is already in there will load and save just fine. The feature set is still being built out, so you can’t add or change things like headers or footnotes yet, but if the document has them python-docx is polite enough to leave them alone and smart enough to save them without actually understanding what they are.

If you use the same filename to open and save the file, python-docx will obediently overwrite the original file without a peep. You’ll want to make sure that’s what you intend.

Opening a ‘file-like’ document

python-docx can open a document from a so-called file-like object. It can also save to a file-like object. This can be handy when you want to get the source or target document over a network connection or from a database and don’t want to (or aren’t allowed to) interact with the file system. In practice this means you can pass an open file or StringIO/BytesIO stream object to open or save a document like so:

f = open('foobar.docx', 'rb')

document = Document(f)

f.close()

# or

with open('foobar.docx', 'rb') as f:

source\_stream = StringIO(f.read())

document = Document(source\_stream)

source\_stream.close()

...

target\_stream = StringIO()

document.save(target\_stream)

The 'rb' file open mode parameter isn’t required on all operating systems. It defaults to 'r' which is enough sometimes, but the ‘b’ (selecting binary mode) is required on Windows and at least some versions of Linux to allow Zipfile to open the file.

Okay, so you’ve got a document open and are pretty sure you can save it somewhere later. Next step is to get some content in there ...

Table Of Contents

Working with Documents

Opening a document

REALLY opening a document

Opening a ‘file-like’ document

Previous topic

Quickstart

Next topic

Working with Text

Useful Links

python-docx @ GitHub

python-docx @ PyPI

Issue Tracker

Quick search

Go

indexnext |previous |python-docx 0.8.6 documentation »

© Copyright 2013, Steve Canny. Created using Sphinx 1.5.3.

Theme based on Read The Docs

v: latest