

Lendo documentos

Arquivos ODT

(Arquivos OpenDocument Text)

Bibliotecas *odfpy* e *ezodf*

Lendo documentos

Open Document

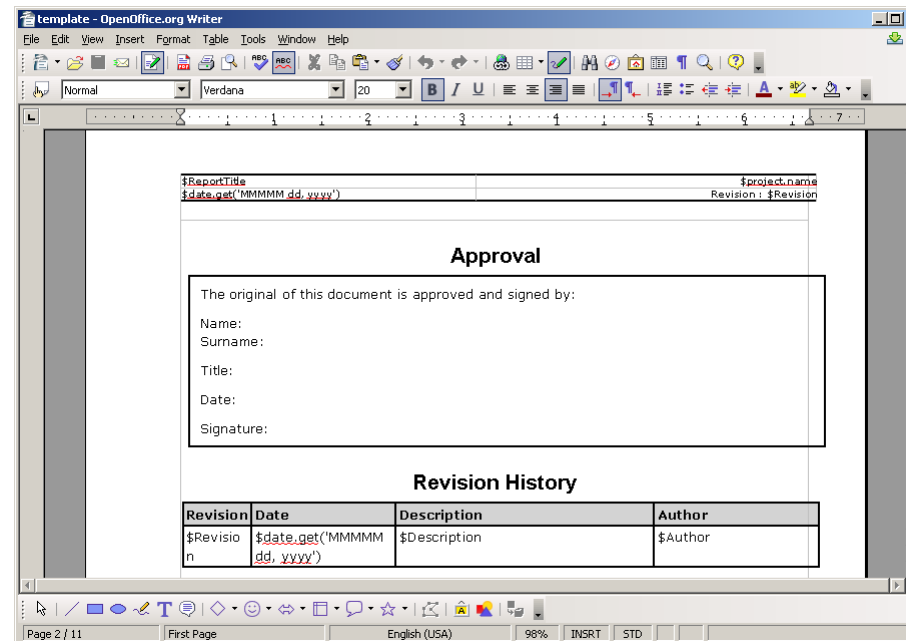
O Open Document Format for Office Applications (ODF), também conhecido como OpenDocument (OD), é uma forma de arquivo usado para armazenamento e troca de documentos de escritório, como textos, planilhas de cálculo, bases de dados, desenhos e apresentações. Este formato foi desenvolvido pelo consórcio OASIS e baseia-se na linguagem XML. O ODF é um formato aberto ao público e foi aprovado como norma ISO/IEC em 8 de Maio de 2006 (ISO/IEC 26300). O ODF foi o primeiro formato de documentos editáveis de escritório a ser aprovado por uma instituição de normalização independente.

O formato ODF foi desenvolvido por uma grande variedade de organizações, sendo possível aceder livremente às respectivas especificações. Isto significa que o ODF pode ser implementado em qualquer sistema, seja ele de código aberto, seja contrário a isto, sem ser necessário efetuar qualquer tipo de pagamento ou estar sujeito a uma licença de uso restrito. O ODF constitui-se como uma alternativa às formas de documentação que são propriedade de empresas privadas, sujeitos a licença de uso restrito ou onerosas, permitindo a organizações e indivíduos escolherem as aplicações para escritório que mais lhes convêm para lidar com os arquivos guardados que o ODF lhes oferece.

Lendo documentos

Arquivos Open Document Text (ODT)

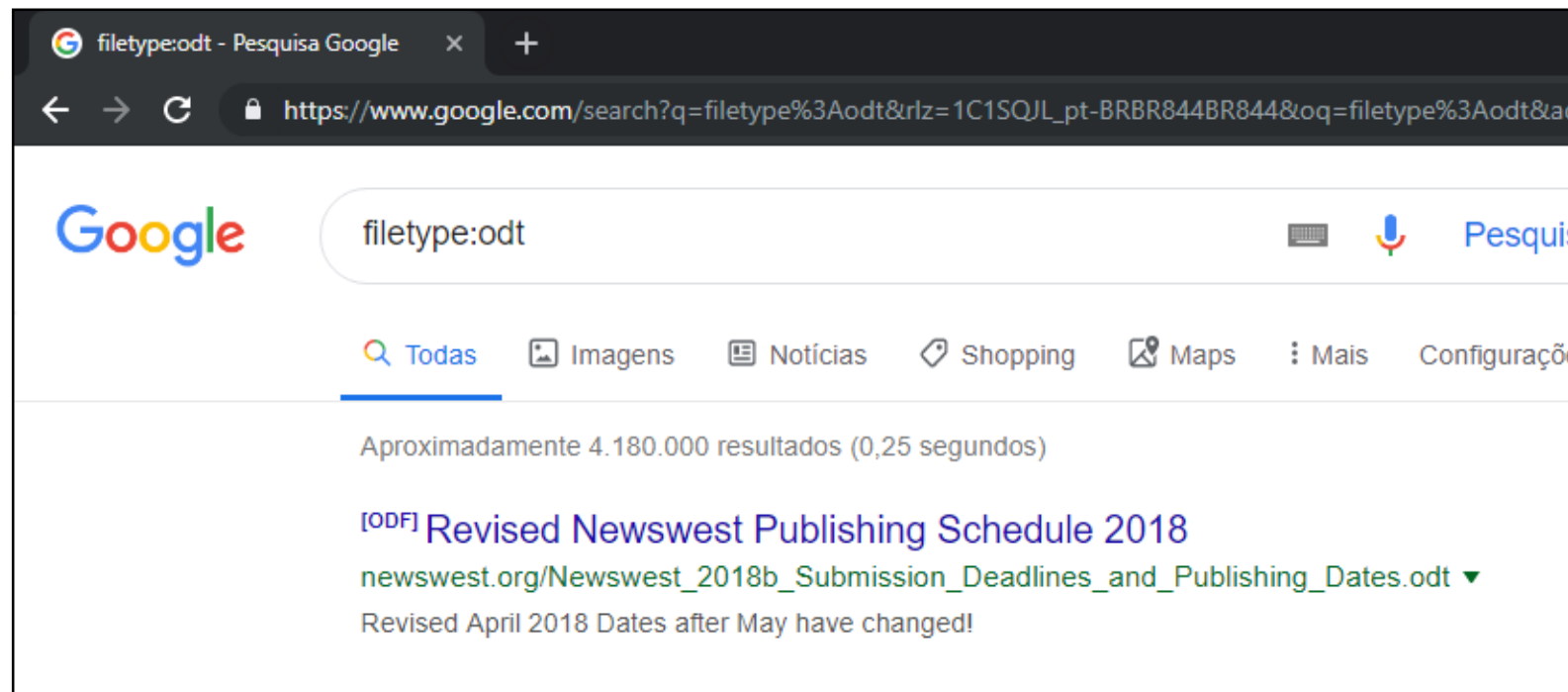
Open Document Text é um formato utilizado para arquivos de texto similares aos arquivos DOC e DOCX do Microsoft Word.



Lendo documentos

Arquivo odt utilizado

Para pegar um documento qualquer de exemplo, pesquisei no Google por “filetype:odt” e peguei o arquivo do primeiro resultado:

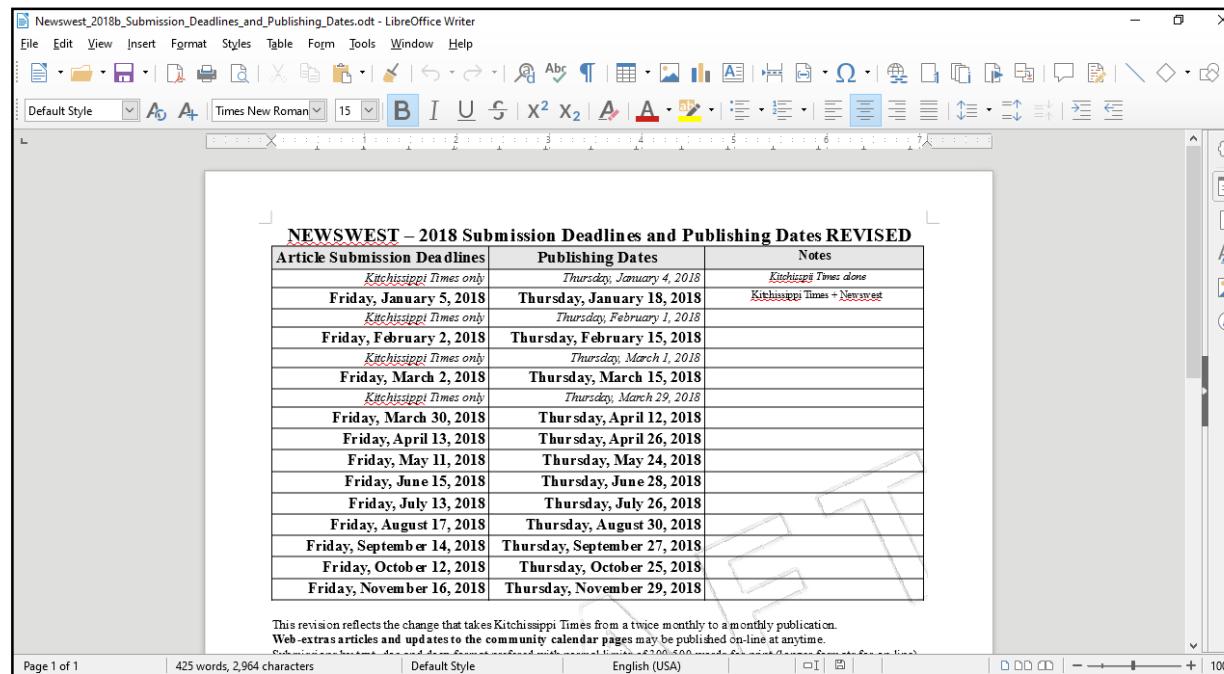


Lendo documentos

Arquivo odt utilizado

Este é o arquivo que vamos utilizar:

http://newswest.org/Newsweb_2018b_Submission_Deadlines_and_Publishing_Dates.odt



The screenshot shows a LibreOffice Writer window with the title "Newsweb_2018b_Submission_Deadlines_and_Publishing_Dates.odt". The document contains a table titled "NEWSWEB – 2018 Submission Deadlines and Publishing Dates REVISED". The table has three columns: "Article Submission Deadlines", "Publishing Dates", and "Notes". The data is organized by month, with each month having a submission deadline on a Friday and a publishing date on a Thursday. A note at the bottom explains that this revision reflects a change from a twice-monthly to a monthly publication.

Article Submission Deadlines	Publishing Dates	Notes
<i>Kitchissippi Times only</i>	<i>Thursday, January 4, 2018</i>	<i>Kitchissippi Times alone</i>
Friday, January 5, 2018	Thursday, January 18, 2018	Kitchissippi Times + Newsweb
<i>Kitchissippi Times only</i>	<i>Thursday, February 1, 2018</i>	
Friday, February 2, 2018	Thursday, February 15, 2018	
<i>Kitchissippi Times only</i>	<i>Thursday, March 1, 2018</i>	
Friday, March 2, 2018	Thursday, March 15, 2018	
<i>Kitchissippi Times only</i>	<i>Thursday, March 29, 2018</i>	
Friday, March 30, 2018	Thursday, April 12, 2018	
Friday, April 13, 2018	Thursday, April 26, 2018	
Friday, May 11, 2018	Thursday, May 24, 2018	
Friday, June 15, 2018	Thursday, June 28, 2018	
Friday, July 13, 2018	Thursday, July 26, 2018	
Friday, August 17, 2018	Thursday, August 30, 2018	
Friday, September 14, 2018	Thursday, September 27, 2018	
Friday, October 12, 2018	Thursday, October 25, 2018	
Friday, November 16, 2018	Thursday, November 29, 2018	

This revision reflects the change that takes Kitchissippi Times from a twice monthly to a monthly publication.
Web-extras articles and updates to the community calendar pages may be published on-line at anytime.

Lendo documentos

Biblioteca Odfpy

Podemos instalar o *odfpy* utilizando o *pip*.

```
Command Prompt
C:\Users\Evaldo>pip install odfpy
Collecting odfpy
  Downloading https://files.pythonhosted.org/packages/85/7d/8f6d1f2a4683be362b101c00232b4c3839e4e4a90e0945d8d43ec6aa671d/odfpy-1.4.0.tar.gz (715kB)
    |████████████████████████████████████████| 716kB 1.3MB/s
Collecting defusedxml (from odfpy)
  Downloading https://files.pythonhosted.org/packages/06/74/9b387472866358ebc08732de3da6dc48e44b0aacd2ddaa5cb85ab7e986a2/defusedxml-0.6.0-py2.py3-none-any.whl
Installing collected packages: defusedxml, odfpy
  Running setup.py install for odfpy ... done
Successfully installed defusedxml-0.6.0 odfpy-1.4.0
C:\Users\Evaldo>
```

Lendo documentos

Biblioteca Odfpy

Odfpy é uma biblioteca para ler e gravar arquivos do OpenDocument v. 1.2.

Ao contrário de outras APIs mais convenientes, o **OdfPy** é essencialmente uma camada de abstração logo acima do formato XML. O foco principal tem sido evitar que o programador crie documentos inválidos. Ele possui verificações que geram uma exceção se o programador adicionar um elemento inválido, adicionar um atributo desconhecido à gramática, esquecer de adicionar um atributo obrigatório ou adicionar texto a um elemento que não o permita.

Lendo documentos

Biblioteca Odfpy

Para trabalhar com OpenDocument Text utilizamos a classe ***opendocument***.

Em nosso primeiro exemplo vamos fazer uma cópia do arquivo. Para isso, vamos carregar o arquivo e salvar com outro nome.

```
from odf.opendocument import load
doc = load("Newwest_2018b_Submission_Deadlines_and_Publishing_Dates.odt")
doc.save("Copia.odt")
```

Load é utilizado para abrir um arquivo e carregar para uma variável, assim, podemos usar o método save para salvar o arquivo informando seu nome.

Lendo documentos

Biblioteca Odfpy

Podemos obter todo texto do arquivo acessando a propriedade text do documento.

```
from odf.opendocument import load
doc = load("Copia.odt")
print(doc.text)
```

Lendo documentos

Biblioteca Odfpy

O método ***getElementsByType(class)*** retorna uma lista com todos elementos de um determinado tipo.

No exemplo a seguir vamos imprimir o conteúdo de todos elementos do tipo parágrafo, usando para isso ***text*** da classe ***odf***.

```
from odf.opendocument import load
from odf import text
doc = load("Copia.odt")
for paragraph in doc.getElementsByType(text.P):
    print(paragraph)
```

Lendo documentos

Biblioteca ezodf

Uma outra alternativa é a biblioteca ezodf, esta possui uma documentação melhor que a odfpy.

Sua documentação fica no seguinte endereço:

<https://pythonhosted.org/ezodf/>

Esta biblioteca pode ser utilizada para trabalhar com todos formatos do padrão ODF, como ODT, ODS, etc, porém, nesta aula vamos trabalhar apenas com documentos de texto.

Lendo documentos

Biblioteca ezodf

Instalando a biblioteca ezodf e a lxml que é uma dependência.

```
C:\Users\Evaldo>pip install ezodf
Collecting ezodf
  Downloading https://files.pythonhosted.org/packages/6f/c5/e966935c26d58d7e3d962270be61be972409084374d4093f478d1f82e8af/ezodf-0.3.2.tar.gz (125kB)
    |████████████████████████████████████████| 133kB 819kB/s
Installing collected packages: ezodf
  Running setup.py install for ezodf ... done
Successfully installed ezodf-0.3.2

C:\Users\Evaldo>
```

```
C:\Users\Evaldo>pip install lxml
Collecting lxml
  Downloading https://files.pythonhosted.org/packages/c6/22/a43126b87020c325fac159bb3b7f4e7ea99e7b2594ce5b8fa23cfa6ee90d/lxml-4.3.4-cp37-cp37m-win_amd64.whl (3.6MB)
    |████████████████████████████████████████| 3.6MB 3.3MB/s
Installing collected packages: lxml
Successfully installed lxml-4.3.4

C:\Users\Evaldo>_
```

Lendo documentos

Biblioteca ezodf

Veja um exemplo de como realizar uma cópia de um arquivo:

```
import ezodf

doc = ezodf.opendoc('copia.odt')
doc.saveas("copia2.odt")
```

Lendo documentos

Biblioteca ezodf

Imprimindo algumas propriedades do documento:

```
import ezodf

doc = ezodf.opendoc('copia.odt')

titulo = doc.meta['title']
print(titulo)
data = doc.meta['date']
print(data)
generator = doc.meta['generator']
print(generator)
description = doc.meta['description']
print(description)
paginas = doc.meta.count['page']
print(paginas)
```

Lendo documentos

Biblioteca ezodf

Vamos agora criar um novo documento e adicionar alguns parágrafos, salvando o mesmo em seguida.

```
import ezodf
from ezodf import Paragraph

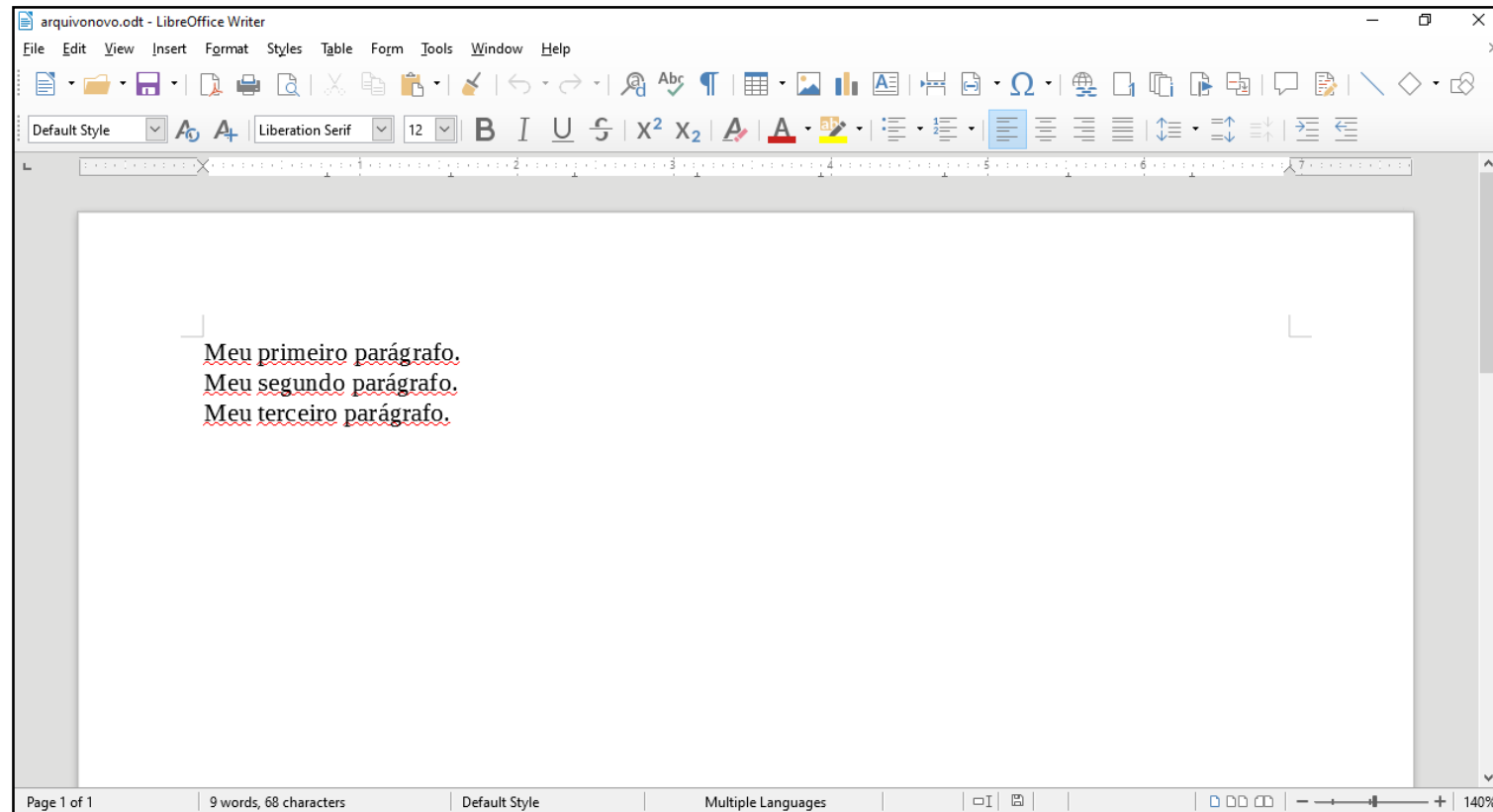
# Criando um documento novo
documento = ezodf.newdoc(doctype="odt", filename="arquivonovo.odt", template=None)
# Adicionando um parágrafo e armazenando o objeto na variável p1
p1 = documento.body.append(Paragraph('Meu terceiro parágrafo.'))
# Adicionando um parágrafo antes do parágrafo p1
documento.body.insert_before(p1, Paragraph('Meu segundo parágrafo.'))
# inserindo um parágrafo passando o índice 0, ele vai ser inserido no
# início do documento
documento.body.insert(0, Paragraph('Meu primeiro parágrafo.'))

# Salvando o documento
documento.save()
```

Lendo documentos

Biblioteca ezodf

Veja como o documento vai ficar.



Lendo documentos

Biblioteca ezodf

Vamos agora adicionar uma lista de itens em nosso documento.

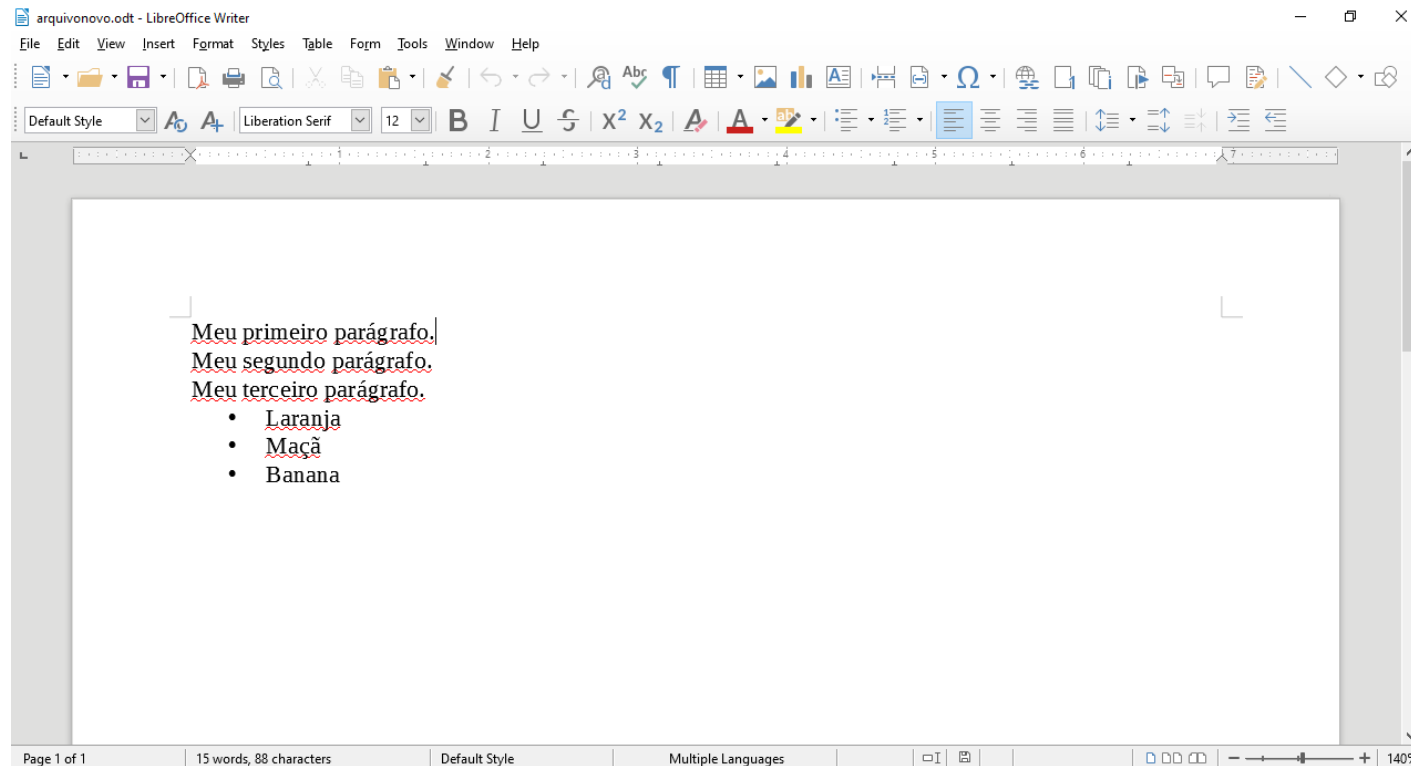
```
import ezodf

documento = ezodf.opendoc('arquivonovo.odt')
# Criando uma lista
lista = ezodf.ezlist(['Laranja', 'Maçã', 'Banana'])
documento.body.append(lista)
documento.save()
```

Lendo documentos

Biblioteca ezodf

Veja agora o documento com a lista.



Lendo documentos

Biblioteca ezodf

Agora vamos imprimir o primeiro elemento do nosso texto e imprimir o total de elementos.

```
import ezodf

documento = ezodf.opendoc('arquivonovo.odt')
# Pegando o primeiro elemento do texto
p1 = documento.body[0]
print(p1)
print(p1.text)
# Quantidade de elementos do documento
count = len(documento.body)
print(count)
```

Lendo documentos

Biblioteca ezodf

Em nosso último exemplo vamos abrir o documento que baixamos e vamos percorrer os elementos, imprimindo os textos dos parágrafos.

```
import ezodf

documento = ezodf.opendoc('Newswest_2018b_Submission_Deadlines_and_Publishing_Dates.odt')

for obj in documento.body:
    if type(obj) is ezodf.text.Paragraph and obj.text is not None:
        print(obj.text)
```

FIM