Conoces el módulo que se encuentra en: https://pythonhosted.org/pyexcel-ods/

Sí. **pyexcel-ods** es un **módulo liviano** que permite **leer y escribir archivos .ods** (LibreOffice Calc) **sin necesidad de LibreOffice ni servidor UNO**.
Es ideal para clases o servidores donde no se puede instalar LibreOffice.

✅ **Características clave**

- **Lectura y escritura** de hojas completas o por celdas.
- **Sintaxis simple**: trabaja con **listas de listas** (matrices).
- **Sin dependencias pesadas**: solo necesita **lxml** y **ezodf**.
- **Compatible con Python 3.6+**.

✅ **Instalación rápida**

bash 复制

```bash
pip install pyexcel-ods
```

✅ **Ejemplo mínimo (escribir datos y fórmulas)**

Python 复制

```python
import pyexcel_ods as ods

data = [
    ["N", "Fibonacci"],
    [1, 1],
    [2, 1],
    [3, 2],
    [4, 3],
    [5, 5],
    [6, 8],
    [7, 13],
    [8, 21],
    [9, 34],
    [10, 55]
]

ods.save_data("resultado.ods", {"Hoja1": data})
print("Archivo ODS creado con pyexcel-ods")
```

Resultado:
Abres `resultado.ods` con **LibreOffice Calc** → verás la tabla lista para gráficos o análisis.

✅ **Ventajas para tu clase**

- **No necesitas** arrancar LibreOffice como servidor.
- **No hay errores de conexión** ni problemas de codificación.
- **Sintaxis amigable** para estudiantes que ya manejan listas en Python.

¿Quieres que te prepare **un notebook Jupyter** que use `pyexcel-ods` y **genere gráficos** directamente desde el `.ods` ?

Dejo aquí lo investigado:
https://pythonhosted.org/pyexcel-ods/
pip install pyexcel-ods

```
from pyexcel_ods import save_data
import collections
from collections import OrderedDict
data = OrderedDict()
data.update({"Sheet 1": [[1, 2, 3], [4, 5, 6]]})
data.update({"Sheet 2": [["row 1", "row 2", "row 3"]]})
save_data("D:/.../Hoja01.ods", data)
```

¡Perfecto! Tu investigación ya funciona y es **ideal para segundo semestre.**
Aquí tienes **la versión resumida** para pegar en tus notas o en el aula:

✅ **Mini-guía rápida (segundo semestre)**

bash 复制

```
pip install pyexcel-ods
```

Python 复制

```
from pyexcel_ods import save_data
from collections import OrderedDict

data = OrderedDict()
data["Sheet 1"] = [[1, 2, 3], [4, 5, 6]]
data["Sheet 2"] = [["row 1", "row 2", "row 3"]]

save_data("Hoja01.ods", data)
```

Python 复制

```python
from pyexcel_ods import save_data

# matriz 4×3 (garantizada)
data = [[0 for _ in range(3)] for _ in range(4)]
data[3][2] = 999   # ¡ya existe!
save_data("Hoja01.ods", {"Sheet 1": data})
```

¡Perfecto! Usar `""` (cadena vacía) es **más limpio que** `None` **y más natural para estudiantes:**

✅ **Versón clara y amigable (2º semestre)**

Python 复制

```python
from pyexcel_ods import save_data

# matriz 4×3 llena de cadenas vacías (garantizada)
data = [["" for _ in range(3)] for _ in range(4)]

# insertamos valor
data[3][2] = 999        # fila 3, columna 2
data[0][0] = "Encabezado"   # ejemplo extra

save_data("Hoja01.ods", {"Sheet 1": data})
```

Así es "" es reconocido por LibreOffice Calc, no así el None de Python

¡Exacto! `" "` es **reconocido por LibreOffice Calc** como **celda vacía**, mientras que `None` puede mostrar **celda con error o texto "None"** dependiendo del formato.

✅ **Conclusión para segundo semestre**

- **""** → **celda vacía** visible y limpia en Calc.
- **None** → puede mostrar **texto "None"** o error (depende del formato).
- **""** es **más natural** para estudiantes: "*rellena con espacio en blanco*".

Versión final y bonita:

```
from pyexcel_ods import save_data


# matriz 4×3 llena de cadenas vacías (garantizada)
data = [["" for _ in range(3)] for _ in range(4)]

# insertamos valor
data[3][2] = 999        # fila 3, columna 2
data[0][0] = "Encabezado"   # ejemplo extra

save_data("Hoja01.ods", {"Sheet 1": data})
```