



Работа с электронными таблицами Excel на Python

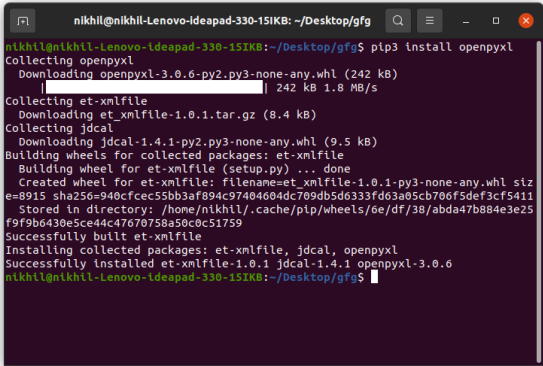
Последнее обновление: 21 августа 2024 г.

Вы все, должно быть, когда-нибудь работали с Excel и, должно быть, испытывали потребность в автоматизации какой-нибудь повторяющейся или утомительной задачи. Не волнуйтесь, в этом руководстве мы узнаем, как работать с Excel с помощью Python или автоматизировать Excel с помощью Python. Мы рассмотрим это с помощью модуля Openpyxl, а также узнаем, как использовать [Питон](#) в Excel.

Начало работы Python Openpyxl

Openpyxl— это библиотека Python, предоставляющая различные методы взаимодействия с файлами Excel с помощью Python. Она позволяет выполнять такие операции, как чтение, запись, арифметические операции, построение графиков и т. д. Этот модуль не входит в стандартную комплектацию Python. Чтобы установить его, введите в терминале эту команду.

```
pip установить openpyxl
```



Чтение файла Excel на Python

Чтобы прочитать файл Excel, вам нужно открыть электронную таблицу с помощью метода `load_workbook()`. После этого вы можете использовать `active` для выбора первого доступного листа и `cell` для выбора ячейки, передавая параметр строки и столбца. Атрибут `value` выводит значение конкретной ячейки. Чтобы лучше понять, прочтите пример ниже.

Примечание: первое число в строке или столбце равно 1, а не 0.

	A	B	C	D	E
1	Name	Course	Branch	Semester	
2	Ankit	B.Tech	CSE	4	
3	Rahul	M.Tech	CSE	2	
4	Priya	MBA	HR	3	
5	Nikhil	B.Tech	CSE	4	
6	Nisha	B.Tech	Biotech	5	
7					

Пример:

В этом примере программа Python использует модуль openpyxl для чтения файла Excel («gfg.xlsx»), открывает главную книгу и получает значение ячейки в первой строке и первом столбце, выводя его на консоль.

```
# import openpyxl module
import openpyxl

# Give the Location of the file
path = "gfg.xlsx"

# To open the workbook
# workbook object is created
wb_obj = openpyxl.load_workbook(path)
```

```
# Get workbook active sheet object

# from the active attribute
sheet_obj = wb_obj.active

cell_obj = sheet_obj.cell(row=1, column=1)

print(cell_obj.value)
```

Выход:

Имя

Python Openpyxl Чтение нескольких ячеек

Возможны два способа считывания данных из нескольких ячеек:

- Чтение строк и столбцов в Excel с помощью openpyxl
- Чтение из нескольких ячеек с использованием имени ячейки

Чтение строк и столбцов в Excel с помощью openpyxl

Мы можем получить количество строк и столбцов, используя **max_row** и **максимальный столбец** соответственно. Мы можем использовать эти значения внутри цикла for, чтобы получить значение нужной строки, столбца или любой ячейки в зависимости от ситуации. Давайте посмотрим, как получить значение первого столбца и первой строки.

В этом примере программа Python, использующая модуль openpyxl, считывает файл Excel («gfg.xlsx»). Она извлекает и выводит общее количество строк и столбцов в активном листе, а затем отображает значения первого столбца и первой строки с помощью итерационных циклов.



```
import openpyxl

# Give the location of the file
path = "gfg.xlsx"

wb_obj = openpyxl.load_workbook(path)

sheet_obj = wb_obj.active

row = sheet_obj.max_row
column = sheet_obj.max_column

print("Total Rows:", row)
print("Total Columns:", column)

print("\nValue of first column")
for i in range(1, row + 1):
    cell_obj = sheet_obj.cell(row=i, column=1)
    print(cell_obj.value)

print("\nValue of first row")
for i in range(1, column + 1):
    cell_obj = sheet_obj.cell(row=2, column=i)
    print(cell_obj.value, end=" ")
```

Выход:

Всего строк: 6
Всего столбцов: 4
Значение первого столбца
Имя
Анкит
Рахул
Прия
Никхил
Ниша
Значение первой строки
Анкит В.Tech CSE 4

Чтение из нескольких ячеек с использованием имени ячейки

Мы также можем считывать данные из нескольких ячеек, используя имя ячейки. Это можно рассматривать как нарезку списка на Python. В этом примере программа на Python использует модуль openpyxl для чтения файла Excel

(«gfg.xlsx»). Она создаёт объектную ячейку, указывая диапазон от «A1» до «B6» на активном листе, и выводит значения каждой пары ячеек в этой структуре с помощью цикла for.

```
import openpyxl

# Give the Location of the file
path = "gfg.xlsx"

wb_obj = openpyxl.load_workbook(path)

sheet_obj = wb_obj.active

cell_obj = sheet_obj['A1': 'B6']

for cell1, cell2 in cell_obj:
    print(cell1.value, cell2.value)
```

Выходной сигнал:

Имя Курс
Анкит Б.Техн.
Рахул М.Техн.
Прия MBA
Никхил Б.Техн.
Ниша Б.Техн.

Подробную информацию о чтении файлов Excel с помощью openpyxl можно найти в статье ниже.

- [Чтение файла Excel с использованием модуля Python openpyxl](#)

Python Запись файла Excel

Сначала давайте создадим новую электронную таблицу, а затем запишем некоторые данные в только что созданный файл. Пустую электронную таблицу можно создать с помощью метода **Рабочая тетрадь()** . Давайте рассмотрим пример ниже.

Пример:

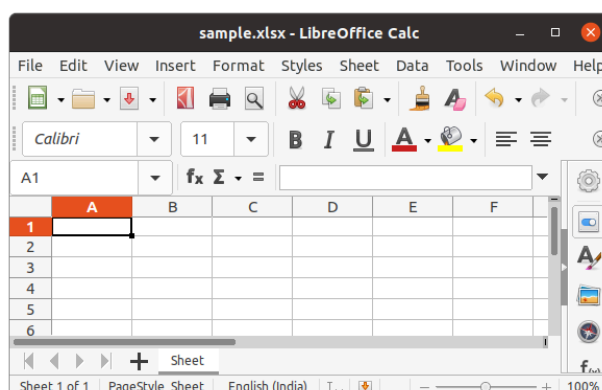
В этом примере новая пустая книга Excel создается с помощью функции библиотеки openpyxl `workbook()` и сохраняется как «sample.xlsx» с помощью `save()` метода. Этот код демонстрирует основные шаги для создания и сохранения файла Excel в Python.

```
from openpyxl import Workbook

workbook = Workbook()

workbook.save(filename="sample.xlsx")
```

Выход:



После создания пустого файла давайте посмотрим, как добавить в него данные с помощью Python. Для добавления данных сначала нужно выбрать активный лист, а затем с помощью метода `cell()` можно выбрать любую конкретную ячейку, передав номер строки и столбца в качестве параметра. Мы также можем писать, используя имена ячеек. Для лучшего понимания см. пример ниже.

Пример:

В этом примере модуль `openpyxl` используется для создания новой книги Excel и заполнения ячеек значениями «Привет», «Мир», «Добро пожаловать» и «Всем». Затем книга сохраняется как «sample.xlsx», что иллюстрирует процесс записи данных в определённые ячейки и сохранения изменений.

```
# import openpyxl module
import openpyxl

wb = openpyxl.Workbook()

sheet = wb.active

c1 = sheet.cell(row=1, column=1)

# writing values to cells
c1.value = "Hello"

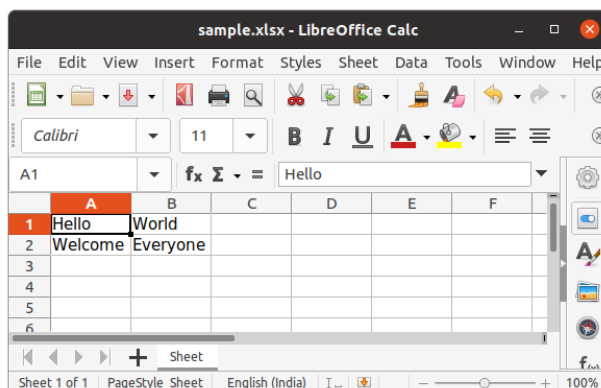
c2 = sheet.cell(row=1, column=2)
c2.value = "World"

c3 = sheet['A2']
c3.value = "Welcome"

# B2 means column = 2 & row = 2.
c4 = sheet['B2']
c4.value = "Everyone"

wb.save("sample.xlsx")
```

Выход:



Подробную информацию о написании текстов в Excel можно найти в статье ниже.

- [Запись в файл Excel с использованием модуля openpyxl](#)

Добавить данные в Excel с помощью Python

В приведенном выше примере вы увидите, что каждый раз, когда вы пытаетесь записать данные в электронную таблицу, существующие данные перезаписываются, и файл сохраняется как новый файл. Это происходит потому, что метод **Рабочая тетрадь()** всегда создает новый объект файла рабочей книги. Чтобы записать данные в существующую рабочую книгу, вы должны открыть файл с помощью метода **load_workbook()**. Мы будем использовать созданную выше рабочую книгу.

Пример:

В этом примере модуль `openpyxl` используется для загрузки существующей книги Excel («sample.xlsx»). Программа обращается к ячейке «A3» на активном листе, обновляет её значение на «Новые данные», а затем сохраняет изменённую книгу обратно в «sample.xlsx».

```
# import openpyxl module
import openpyxl

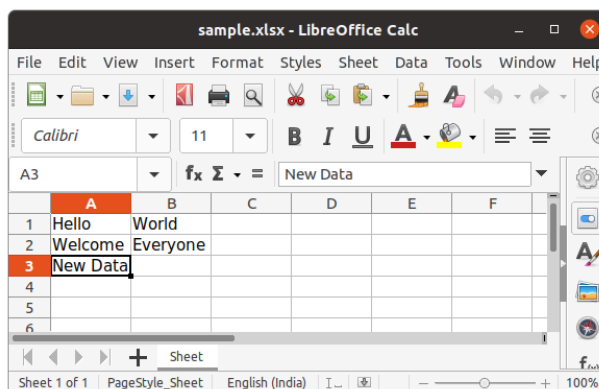
wb = openpyxl.load_workbook("sample.xlsx")

sheet = wb.active

c = sheet['A3']
c.value = "New Data"

wb.save("sample.xlsx")
```

Выход:



Мы также можем использовать метод **добавить()** для добавления нескольких данных в конец листа.

Пример:

В этом примере модуль `openpyxl` используется для загрузки отдельных книг Excel («sample.xlsx»). Двумерная структура данных (кортеж кортежей) определяется и итеративно добавляется к активному листу, фактически добавляя строки со значениями (1, 2, 3) и (4, 5, 6).

```
# import openpyxl module
import openpyxl

wb = openpyxl.load_workbook("sample.xlsx")

sheet = wb.active

data = (
    (1, 2, 3),
    (4, 5, 6)
)

for row in data:
    sheet.append(row)

wb.save('sample.xlsx')
```

Выходной сигнал:

	A	B	C	D
1	Hello	World		
2	Welcome	Everyone		
3	New Data			
4	1	2	3	
5	4	5	6	
6				
7				

Арифметические операции в электронных таблицах

Арифметические операции можно выполнять, вводя формулу в определённую ячейку электронной таблицы. Например, если мы хотим найти сумму, то используется формула **=Sum()** из файла Excel.

Пример:

В этом примере модуль `openpyxl` используется для создания новой книги Excel и заполнения ячеек A1–A5 числовыми значениями. Ячейке A7 присваивается формула для вычисления суммы значений в ячейках A1–A5.

```
# import openpyxl module
import openpyxl

wb = openpyxl.Workbook()

sheet = wb.active

# writing to the cell of an excel sheet
sheet['A1'] = 200
sheet['A2'] = 300
sheet['A3'] = 400
sheet['A4'] = 500
sheet['A5'] = 600

sheet['A7'] = '= SUM(A1:A5)'

# save the file
wb.save("sum.xlsx")
```

Выход:

	A	B
1	200	
2	300	
3	400	
4	500	
5	600	
6		
7	2000	
8		
9		

Подробную информацию об арифметических операциях в электронных таблицах можно найти в статье ниже.


- [Арифметические операции в файле Excel с использованием openpyxl](#)

Настройка строк и столбцов

Объекты Worksheet имеют атрибуты row_dimensions и column_dimensions, которые управляют высотой строк и шириной столбцов. row_dimensions и column_dimensions листа являются значениями, подобными словарю; row_dimensions содержит объекты RowDimension, а column_dimensions содержит объекты ColumnDimension. В row_dimensions можно получить доступ к одному из объектов, используя номер строки (в данном случае 1 или 2). В column_dimensions можно получить доступ к одному из объектов, используя букву столбца (в данном случае A или B).

Пример:

В этом примере модуль openpyxl используется для создания новой книги Excel и установки значений в определённых ячейках. Содержимое «привет» помещается в ячейку A1, а «всем» — в ячейку B2. Кроме того, высота первой строки установлена на 70 единиц, а ширина столбца B — на 20 единиц.



```
# import openpyxl module
import openpyxl

wb = openpyxl.Workbook()

sheet = wb.active

# writing to the specified cell
sheet.cell(row=1, column=1).value = ' hello '

sheet.cell(row=2, column=2).value = ' everyone '

# set the height of the row
sheet.row_dimensions[1].height = 70

# set the width of the column
sheet.column_dimensions['B'].width = 20

# save the file
wb.save('sample.xlsx')
```

Выход:

	A	B	C
1	hello		
2		everyone	
3			
4			

Объединение ячеек

Прямоугольную область ячеек можно объединить в одну ячейку с помощью метода merge_cells(). Аргумент merge_cells() — это одна строка из верхних левых и нижних правых ячеек прямоугольной области, которую нужно объединить.

Пример:

В этом примере модуль openpyxl используется для создания новой книги Excel. Программа объединяет ячейки A2 и

D4, создавая одну ячейку, охватывающую несколько столбцов и строк, и устанавливает в ней значение «Двенадцать ячеек объединяются». Кроме того, ячейки C6 и D6 объединяются, и в полученную объединённую ячейку помещается текст «Две ячейки объединяются».

```
import openpyxl
wb = openpyxl.Workbook()
sheet = wb.active

sheet.merge_cells('A2:D4')

sheet.cell(row=2, column=1).value = 'Twelve cells join together.'

# merge cell C6 and D6
sheet.merge_cells('C6:D6')

sheet.cell(row=6, column=6).value = 'Two merge cells.'

wb.save('sample.xlsx')
```

Выход:

	A	B	C	D	E	F	G	H
1								
2	Twelve cells join together.							
3								
4								
5								
6			Two merge cells.					
7								
8								
9								

Разделение ячеек

Чтобы разделить ячейки, вызовите метод листа `unmerge_cells()`.

Пример:

В этом примере модуль `openpyxl` используется для загрузки существующей книги Excel («sample.xlsx»). Затем программа разъединяет ранее объединённые ячейки, в частности ячейки A2–D4 и ячейки C6–D6.

```
import openpyxl

wb = openpyxl.load_workbook('sample.xlsx')
sheet = wb.active

# unmerge the cells
sheet.unmerge_cells('A2:D4')

sheet.unmerge_cells('C6:D6')

wb.save('sample.xlsx')
```

Выход:

	A	B	C	D	E	F	G	H
1								
2	Twelve cells join together.							
3								
4								
5								
6			Two merge cells.					
7								
8								
9								

Установка стиля шрифта

Чтобы настроить стили шрифтов в ячейках, важно импортировать функцию **Шрифт()** из модуля `openpyxl.стили`.

Пример:

В этом примере модуль `openpyxl` используется для создания новой книги Excel. Программа устанавливает значения в разных ячейках с текстом «GeeksforGeeks» и применяет различные стили шрифта к каждой ячейке.



```
import openpyxl

# import Font function from openpyxl
from openpyxl.styles import Font

wb = openpyxl.Workbook()
sheet = wb.active

sheet.cell(row = 1, column = 1).value = "GeeksforGeeks"

# set the size of the cell to 24
sheet.cell(row = 1, column = 1).font = Font(size = 24 )

sheet.cell(row = 2, column = 2).value = "GeeksforGeeks"

# set the font style to italic
sheet.cell(row = 2, column = 2).font = Font(size = 24, italic = True)

sheet.cell(row = 3, column = 3).value = "GeeksforGeeks"

# set the font style to bold
sheet.cell(row = 3, column = 3).font = Font(size = 24, bold = True)

sheet.cell(row = 4, column = 4).value = "GeeksforGeeks"

# set the font name to 'Times New Roman'
sheet.cell(row = 4, column = 4).font = Font(size = 24, name = 'Times New Roman')

wb.save('sample.xlsx')
```

Выход:

	A	B	C	D	E	F	G
1	GeeksforGeeks						
2		GeeksforGeeks					
3			GeeksforGeeks				
4				GeeksforGeeks			
5							
6							

Обратитесь к приведенной ниже статье, чтобы получить подробную информацию об изменении строк и столбцов.

- [Настройка строк и столбцов файла Excel с помощью модуля openpyxl](#)

Построение графиков

Диаграммы основаны как минимум на одной серии с одной или несколькими точками данных. Сами серии основаны на ссылках на различные ячейки. Чтобы построить диаграмму на листе Excel, сначала создайте диаграммы объектов определённого класса (например, гистограмму, линейную диаграмму и т. д.). После создания диаграмм объектов вставьте в них данные и, наконец, запишите эти объектные диаграммы на лист.

Пример 1. Создание и настройка гистограмм в Excel с помощью openpyxl

В этом случае модуль openpyxl используется для создания новых книг Excel. Числовые значения от 0 до 9 для определения первого столбца активного листа. Затем создаётся объект BarChart, и данные для построения графика отображаются с помощью класса Reference. График введён с помощью заголовка, оси названия X и оси Y. Наконец, график добавляется на лист и привязывается к ячейке E2.



```
# import openpyxl module
import openpyxl

# import BarChart class from openpyxl.chart sub_module
from openpyxl.chart import BarChart, Reference

wb = openpyxl.Workbook()

sheet = wb.active

# write 0 to 9 in 1st column of the active sheet
for i in range(10):
    sheet.append([i])

# create data for plotting
values = Reference(sheet, min_col=1, min_row=1,
                    max_col=1, max_row=10)

# Create object of BarChart class
```



```

# create object of BarChart class
chart = BarChart()

# adding data to the Bar chart object
chart.add_data(values)

# set the title of the chart
chart.title = " BAR-CHART "

# set the title of the x-axis
chart.x_axis.title = " X_AXIS "

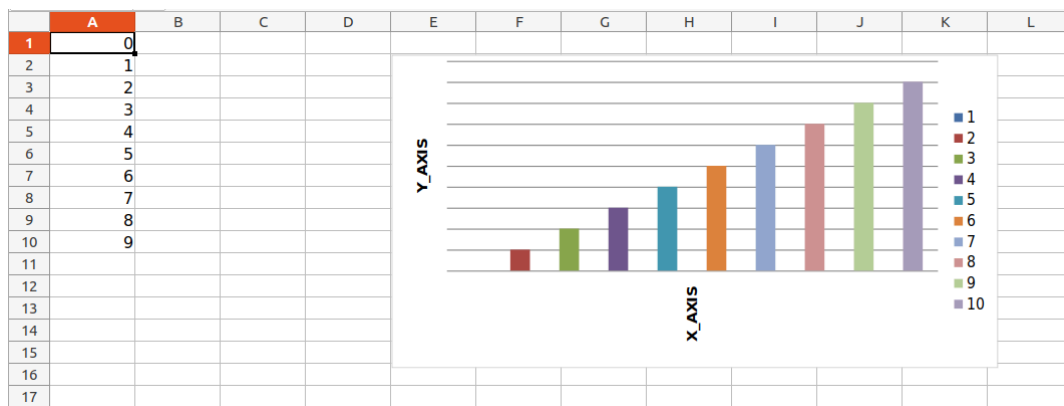
# set the title of the y-axis
chart.y_axis.title = " Y_AXIS "

sheet.add_chart(chart, "E2")

# save the file
wb.save("sample.xlsx")

```

Выход:



Пример 2: Создание и настройка линейной диаграммы в Excel с помощью openpyxl

В этом примере модуль openpyxl используется для создания новой книги Excel. Числовые значения от 0 до 9 записываются в первый столбец активного листа. Затем создаётся объект LineChart, а данные для построения графика указываются с помощью класса Reference. Диаграмма настраивается с помощью заголовка, заголовка оси X и заголовка оси Y. Наконец, диаграмма добавляется на лист и привязывается к ячейке E2.



```

# import openpyxl module
import openpyxl

# import LineChart class from openpyxl.chart sub_module
from openpyxl.chart import LineChart, Reference

wb = openpyxl.Workbook()
sheet = wb.active

# write 0 to 9 in 1st column of the active sheet
for i in range(10):
    sheet.append([i])

values = Reference(sheet, min_col=1, min_row=1,
                    max_col=1, max_row=10)

# Create object of LineChart class
chart = LineChart()

chart.add_data(values)

# set the title of the chart
chart.title = " LINE-CHART "

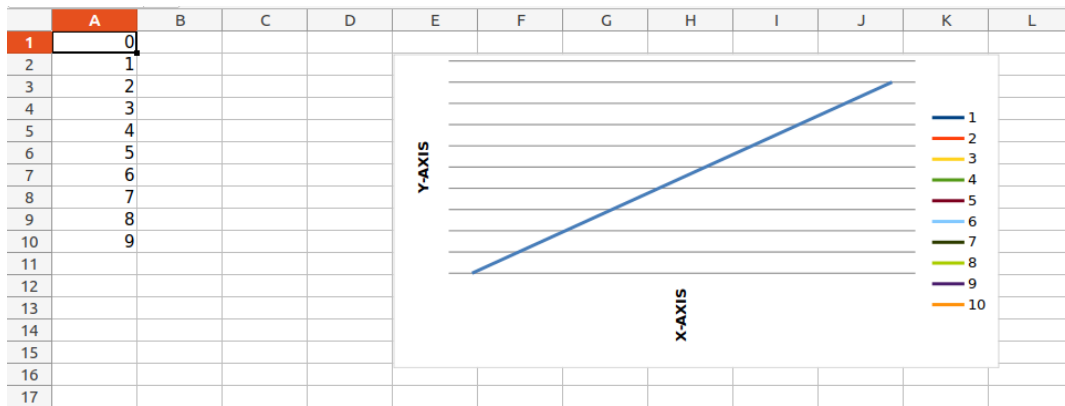
# set the title of the x-axis
chart.x_axis.title = " X-AXIS "

# set the title of the y-axis
chart.y_axis.title = " Y-AXIS "
sheet.add_chart(chart, "E2")

# save the file
wb.save("sample.xlsx")

```

Выход:



Подробную информацию о построении графиков в Excel с помощью Python можно найти в статьях ниже.

- [Построение диаграмм в таблице Excel с использованием модуля openpyxl | Набор 1](#)
- [Построение диаграмм в таблице Excel с использованием модуля openpyxl | Набор 2](#)
- [Построение диаграмм в таблице Excel с использованием модуля openpyxl | Набор 3](#)

Добавление изображений

Для импорта изображений в наш рабочий лист мы будем использовать `openpyxl.рисунки.изображение.Изображение`. Метод является оболочкой для метода `PIL.Image`, найденного в библиотеке `PIL (pillow)`. Поэтому для использования этого метода необходимо установить библиотеку `PIL (pillow)`.

Использованное изображение:



Пример:

В этом примере модуль `openpyxl` используется для создания новой книги Excel. Строка данных добавляется на активный лист, чтобы отличить её от изображения. Затем на лист добавляется изображение («`geek.jpg`») с помощью класса `openpyxl.drawing.image.Image`, и оно размещается в ячейке `A2`.

```
import openpyxl
from openpyxl.drawing.image import Image

wb = openpyxl.Workbook()

sheet = wb.active


sheet.append([10, 2010, "Geeks", 4, "life"])

img = Image("geek.jpg")

sheet.add_image(img, 'A2')

# Saving the workbook created
wb.save('sample.xlsx')
```

Выход:

	A	B	C	D	E
1	10	2010	Geeks		4 life
2					
3					
4					
5					
6					

Подробную информацию о добавлении изображений можно найти в статье ниже.

- [Openpyxl – Добавление изображения](#)

Еще немного функциональности Excel с использованием Python

- [Как удалить одну или несколько строк в Excel с помощью Openpyxl?](#)
- [Тригонометрические операции в файле Excel с использованием openpyxl](#)
- [Как скопировать данные из одного листа Excel в другой](#)
- [Как автоматизировать таблицу Excel на Python?](#)

Работа с таблицами Excel на Python – часто задаваемые вопросы

Как использовать Excel с Python?

Чтобы использовать Excel с Python, вы можете использовать такие библиотеки, как `pandas`, `openpyxl`, или `xlrd`/`xlwt`. Эти библиотеки помогают читать, записывать и обрабатывать файлы Excel. `pandas` особенно полезен для анализа и обработки данных и может легко считывать и записывать файлы Excel с помощью функций `read_excel()` и `to_excel()`.

Пример с пандами:

```
import pandas as pd

# Чтение файла Excel
df = pd.read_excel('filename.xlsx')

# Обработка данных
df['new_column'] = df['existing_column'] * 10

# Запись в файл Excel
df.to_excel('modified_filename.xlsx', index=False)
```

Можно ли автоматизировать Excel с помощью Python?

Да, вы можете автоматизировать задачи Excel с помощью Python, используя библиотеки, например, `openpyxl` для обработки `.xlsx` файлов или `xlwings` для взаимодействия с приложениями Excel. Эти инструменты позволяют автоматизировать повторяющиеся задачи, такие как форматирование ячеек, вставка данных и создание диаграмм.

Пример с openpyxl:

```
from openpyxl import Workbook

wb = Workbook()
ws = wb.active

# Добавление данных
ws['A1'] = "Привет"
ws['A2'] = "Мир!"

# Сохранение рабочей книги
wb.save("example.xlsx")
```

В чем разница между pandas и openpyxl?

- **панды** — это высокоуровневый инструмент обработки данных, разработанный в первую очередь для анализа данных. Он предоставляет функциональные возможности для чтения и записи в файлы Excel, но не способен взаимодействовать с файлами Excel на низком уровне (например, изменять стили ячеек или другие функции форматирования).
- **openpyxl** — это библиотека, предназначенная для чтения и записи в `.xlsx` файлы Excel. Она обеспечивает более детальный контроль над файлами Excel, например, настройку стилей ячеек, фильтров и формул, что `pandas` напрямую не позволяет.

Можно ли импортировать файл Excel в Python?

Да, вы можете импортировать файлы Excel в Python, используя несколько библиотек. `pandas` — это самая распространённая библиотека для работы со структурированными файлами данных, включая Excel, поскольку она может быстро загружать файл Excel в `DataFrame`, что позволяет выполнять обширные операции с данными и их анализ.

Пример с пандами:

```
import pandas as pd

# Загрузка файла Excel в DataFrame
df = pd.read_excel('your_file.xlsx')
print(df)
```

Можно ли использовать Python в Excel вместо VBA?

Да, Python можно использовать в Excel как альтернативу VBA через библиотеки, такие как `xlwings`. Эта библиотека позволяет коду Python напрямую взаимодействовать с Excel, используя синтаксис и возможности Python для автоматизации, анализа данных и визуализации в Excel. Это особенно полезно для пользователей, которые хотят использовать расширенные возможности Python, не выходя из среды Excel.

[Учебное пособие по Python](#) [Вопросы для интервью](#) [Тест на Python](#) [Проекты на Python](#) [Практикуйте Python](#) [Наука о данных с помощью Python](#) [Веб-разработчик Р](#)

```
import xlwings as xw

# Подключиться к существующей книге Excel
wb = xw.Book('example.xlsx')
sheet = wb.sheets['Sheet1']

# Записать данные в Excel
sheet.range('A1').value = 'Привет, Excel!'

# Выполнить формулы Excel
sheet.range('A2').value = '=SUM(1, 2, 3)'

wb.save()
wb.close()
```

Эти инструменты и методы делают Python мощным помощником при работе с Excel, позволяя выполнять как простые манипуляции с данными, так и сложные рабочие задачи.

Комментарий

Дополнительная информация

Рекламируйтесь у нас

Следующая статья

[Работа с CSV-файлами в Python](#)

Похожие чтения

Работа с электронными таблицами Excel на Python

Вы все, должно быть, работали с Excel в какой-то момент своей жизни и, должно быть, чувствовали необходимость автоматизировать какую-то повторяющуюся или утомительную задачу. Не волнуйтесь, в этом уроке мы узнаем, как работать с Excel с помощью Python или автоматизировать Excel с помощью Python. Мы...

15 мин чтения

Работа с CSV-файлами в Python

Python — одна из важных областей для специалистов по работе с данными и многих программистов, где необходимо обрабатывать различные данные. CSV (значения, разделённые запятыми) — один из распространённых и доступных форматов файлов для хранения и обмена табличными данными. В статье...

10 мин чтения

Работа с файлами Excel на Python с использованием Xlwings

Xlwings — это библиотека Python, которая позволяет легко вызывать Python из Excel и наоборот. Она легко создает чтение и запись в Excel и из него с помощью Python. Ее также можно модифицировать для работы в качестве сервера Python для Excel для синхронного обмена данными между Python и Excel. Xlwings...

3 мин чтения

Работа с файлами Excel с использованием Pandas

Таблицы Excel интуитивно понятны и удобны для пользователя, что делает их идеальными для работы с большими наборами данных даже для менее технически подкованных людей. Если вы ищете места, где можно научиться манипулировать и автоматизировать работу с файлами Excel с помощью Python, т...

7 мин чтения

Преобразование любых дат в электронных таблицах с помощью Python

В этой статье мы рассмотрим, как преобразовать любые даты в электронных таблицах с помощью Python. Используемый файл/этот файл содержит один...

В этой статье мы рассмотрим, как преобразовать любые даты в электронных таблицах с помощью Python. Используемый файл: Этот файл содержит один столбец под названием «Дата» и хранит случайные даты 2021 года в различных форматах. Подход: начнем с импорта библиотеки pandas. Давайте посмотрим...

3 мин чтения

Запись в таблицу Excel с использованием Python

Используя модуль xlwt, можно выполнять множество операций с электронными таблицами. Например, запись или изменение данных можно выполнить в Python. Также пользователю может потребоваться просмотреть различные таблицы и извлечь данные на основе некоторых критериев или изменить некоторы...

2 мин чтения

Чтение файла Excel с помощью Python

Можно получить информацию из электронной таблицы. Чтение, запись или изменение данных в Python можно выполнять с помощью различных методов. Кроме того, пользователю может потребоваться просмотреть различные листы и получить данные по определённым критериям или изменить некоторые...

4 min read

How to replace a word in excel using Python?

Excel is a very useful tool where we can have the data in the format of rows and columns. We can say that before the database comes into existence, excel played an important role in the storage of data. Nowadays using Excel input, many batch processing is getting done. There may be the requirement o

3 min read

How to Read Excel Multiple Sheets in Python Pandas

Reading multiple sheets from an Excel file into a Pandas DataFrame is a basic task in data analysis and manipulation. It allows us to work with data spread across different sheets efficiently within the Pandas framework. Read Excel Multiple Sheets in PandasWe use the pd.read_excel() function to read

4 min read

Python for Spreadsheet Users

Excel is one of the popular spreadsheets for data management. While it is tedious to write and update the data in a long Excel sheet, Python helps in minimizing this task and helps easy creation, reading, writing of Excel sheet. This can be done by various Python libraries,3 of which will be discuss

5 min read

Reading and Writing CSV Files in Python

CSV (Comma Separated Values) format is the most common import and export format for spreadsheets and databases. It is one of the most common methods for exchanging data between applications and popular data format used in Data Science. It is supported by a wide range of applications. A CSV file stor

4 min read

Python | Working with Pandas and XlsxWriter | Set - 1

Python Pandas is a data analysis library. It can read, filter and re-arrange small and large datasets and output them in a range of formats including Excel. Pandas writes Excel files using the XlsxWriter modules. XlsxWriter is a Python module for writing files in the XLSX file format. It can be used

3 min read

Python | Working with Pandas and XlsxWriter | Set – 3

Prerequisite: : Python working with pandas and xlsxwriter | set-1 Python Pandas is a data analysis library. It can read, filter and re-arrange small and large datasets and output them in a range of formats including Excel. Pandas writes Excel files using the XlsxWriter modules. XlsxWriter is a Pytho

5 min read

Python | Working with Pandas and XlsxWriter | Set – 2

Prerequisite: : Python working with pandas and xlsxwriter | set-1 Python Pandas is a data analysis library. It can read, filter and re-arrange small and large datasets and output them in a range of formats including Excel. Pandas writes Excel files using the XlsxWriter modules. XlsxWriter is a Pytho

4 min read

How to Automate an Excel Sheet in Python?

Before you read this article and learn automation in Python....let's watch a video of Christian Genco (a talented programmer and an entrepreneur) explaining the importance of coding by taking the example of automation. You might have laughed loudly after watching this video and you surely, you might

8 min read

Add New Sheet to Excel Using Pandas

The article aims to implement a function in Python using Pandas to add a new sheet to an existing Excel file, facilitating efficient data organization and management within Excel workbooks. Using openpxl to add a new sheet to Excel Using PandasPandas offer a seamless interface for manipulating Excel

2 min read

Store Google Sheets data into SQLite Database using Python

In this article, we are going to store google sheets data into a database using python. The first step is to enable the API and to create the credentials, so let's get started. Enabling the APIs and creating the credentialsGo to Marketplace in Cloud Console.Click on ENABLE APIS AND SERVICESThen Searc

5 min read

Save user input to a Excel File in Python

In this article, we will learn how to store user input in an excel sheet using Python. What is Excel? Excel is a spreadsheet in a computer application that is designed to add,

display, analyze, organize, and manipulate data arranged in rows and columns. It is the most popular application for account

4 min read

How to Read an Excel File using polars

The Polars is a fast, efficient DataFrame library in Python, designed for processing large datasets with low memory usage and high performance. While Polars is more commonly used with CSV, Parquet, and JSON files, we can also work with Excel files, though this requires an additional setup as Polars

4 min read

Using Google Sheets as Database in Python

In this article we'll be discussing how we can use Google Sheets to run like a database for any Python file. Google Spreadsheets:Google Spreadsheets is free online web based application that resembles Microsoft Excel. You can use it to create and edit tables for various projects such as Contact Lis

4 min read



Corporate & Communications Address:

A-143, 7th Floor, Sovereign Corporate
Tower, Sector- 136, Noida, Uttar Pradesh
(201305)

Registered Address:

K 061, Tower K, Gulshan Vivante
Apartment, Sector 137, Noida, Gautam
Buddh Nagar, Uttar Pradesh, 201305

Advertise with us

Company

About Us
Legal
Privacy Policy
In Media
Contact Us
Advertise with us
GFG Corporate Solution
Placement Training Program
GeeksforGeeks Community

Languages

Python
Java
C++
PHP
GoLang
SQL
R Language
Android Tutorial
Tutorials Archive

DSA

Data Structures
Algorithms
DSA for Beginners
Basic DSA Problems
DSA Roadmap
Top 100 DSA Interview
Problems
DSA Roadmap by Sandeep
Jain
All Cheat Sheets

Data Science & ML

Data Science With Python
Data Science For Beginner
Machine Learning
ML Maths
Data Visualisation
Pandas
NumPy
NLP
Deep Learning

Web Technologies

HTML
CSS
JavaScript
TypeScript
ReactJS
NextJS
Bootstrap
Web Design

Python Tutorial

Python Programming
Examples
Python Projects
Python Tkinter
Web Scraping
OpenCV Tutorial
Python Interview Question
Django

Computer Science

Operating Systems
Computer Network
Database Management
System
Software Engineering
Проектирование цифровой
логики
Инженерная математика
Разработка программного
обеспечения
Тестирование программного
обеспечения

DevOps

Гит
линукс
ABC
Докер
Кубернетес
Лазурный
GCP
Дорожная карта DevOps

Проектирование системы

Проектирование высокого
уровня
Низкоуровневое
проектирование
Диаграммы UML
Руководство по
собеседованию
Шаблоны проектирования
ООАД
Учебный лагерь по
проектированию систем
Вопросы для интервью

Подготовка к собеседованию

Конкурсное
программирование
Лучший DS или Алгоритм
для CP
Процесс подбора персонала
в компании
Подготовка на уровне
компании
Подготовка к способностям
Пазлы

Школьные предметы

Математика
Физика
Химия
Биология
Социальные науки
Грамматика английского
языка
Коммерция
Мировой ГК

Видео GeeksforGeeks

ДСА
Питон
Ява
C++
Веб-разработка
Наука о данных
Предметы CS