**Python read excel file**

Excel is a spreadsheet application which is developed by Microsoft. It is an easily accessible tool to organize, analyze, and store the data in tables. It is widely used in many different applications all over the world. From Analysts to CEOs, various professionals use Excel for both quick stats and serious data crunching.

**Excel Documents**

An Excel spreadsheet document is called a workbook which is saved in a file with **.xlsx** extension. The first row of the spreadsheet is mainly reserved for the header, while the first column identifies the sampling unit. Each workbook can contain multiple sheets that are also called a worksheets. A box at a particular column and row is called a cell, and each cell can include a number or text value. The grid of cells with data forms a sheet.

The active sheet is defined as a sheet in which the user is currently viewing or last viewed before closing Excel.

**Reading from an Excel file**

First, you need to write a command to install the **xlrd** module.

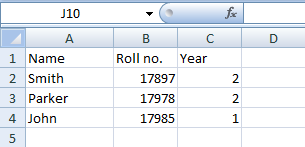
1. pip install xlrd

**Creating a Workbook**

A workbook contains all the data in the excel file. You can create a new workbook from scratch, or you can easily create a workbook from the excel file that already exists.

**Input File**

We have taken the snapshot of the workbook.



**Code**

1. # Import the xlrd module
2. import xlrd
4. # Define the location of the file
5. loc = ("path of file")
7. # To open the Workbook
8. wb = xlrd.open\_workbook(loc)
9. sheet = wb.sheet\_by\_index(0)
11. # For row 0 and column 0
12. sheet.cell\_value(0, 0)

**Explanation:** In the above example, firstly, we have imported the xlrd module and defined the location of the file. Then we have opened the workbook from the excel file that already exists.

**Reading from the Pandas**

Pandas is defined as an open-source library which is built on the top of the NumPy library. It provides fast analysis, data cleaning, and preparation of the data for the user and supports both xls and xlsx extensions from the URL.

It is a python package which provides a beneficial data structure called a data frame.

**Example**

1. Example -
2. import pandas as pd
4. # Read the file
5. data = pd.read\_csv(".csv", low\_memory=False)
7. # Output the number of rows
8. print("Total rows: {0}".format(len(data)))
10. # See which headers are available
11. print(list(data))

**Reading from the openpyxl**

First, we need to install an openpyxl module using pip from the command line.

1. pip install openpyxl

After that, we need to import the module.

We can also read data from the existing spreadsheet using openpyxl. It also allows the user to perform calculations and add content that was not part of the original dataset.

**Example**

1. import openpyxl
2. my\_wb = openpyxl.Workbook()
3. my\_sheet = my\_wb.active
4. my\_sheet\_title = my\_sheet.title
5. print("My sheet title: " + my\_sheet\_title)

**Output:**

My sheet title: Sheet

To learn more about openpyxl, visit our complete tutorial [Click Here](https://www.javatpoint.com/python-openpyxl)

. We have discussed essential detail in this tutorial.

# Python Write Excel File

The Python write excel file is used to perform the multiple operations on a spreadsheet using the **xlwt** module. It is an ideal way to write data and format information to files with .xls extension.

If you want to write data to any file and don't want to go through the trouble of doing everything by yourself, then you can use a for loop to automate the whole process a little bit.

## Write Excel File Using xlsxwriter Module

We can also write the excel file using the **xlsxwriter** module. It is defined as a Python module for writing the files in the XLSX file format. It can also be used to write text, numbers, and formulas to multiple worksheets. Also, it supports features such as charts, formatting, images, page setup, auto filters, conditional formatting, and many others.

We need to use the following command to install xlsxwriter module:

1. pip install xlsxwriter

#### Note- Throughout XlsxWriter, rows, and columns are zero-indexed. The first cell in a worksheet is listed as, A1 is (0,0), B1 is (0,1), A2 is (1,0), B2 is (1,1)......,and so on.

## Write Excel File Using openpyxl Module

It is defined as a package which is generally recommended if you want to read and write .xlsx, xlsm, xltx, and xltm files. You can check it by running **type(wb)**.

The load\_workbook() function takes an argument and returns a workbook object, which represents the file. Make sure that you are in the same directory where your spreadsheet is located. Otherwise, you will get an error while importing.

You can easily use a for loop with the help of the range() function to help you to print out the values of the rows that have values in column 2. If those particular cells are empty, you will get None.

## Writing data to Excel files with xlwt

You can use the xlwt package, apart from the XlsxWriter package to create the spreadsheets that contain your data. It is an alternative package for writing data, formatting information, etc. and ideal for writing the data and format information to files with .xls extension. It can perform multiple operations on the spreadsheet.

It supports features such as formatting, images, charts, page setup, auto filters, conditional formatting, and many others.

Pandas have excellent methods for reading all kinds of data from excel files. We can also import the results back to pandas.

## Writing Files with pyexcel

You can easily export your arrays back to a spreadsheet by using the save\_as() function and pass the array and name of the destination file to the dest\_file\_name argument.

It allows us to specify the delimiter and add dest\_delimiter argument. You can pass the symbol that you want to use as a delimiter in-between " ".

**Code**

1. # import xlsxwriter module
2. import xlsxwriter
4. book = xlsxwriter.Book('Example2.xlsx')
5. sheet = book.add\_sheet()
7. # Rows and columns are zero indexed.
8. row = 0
9. column = 0
11. content = ["Parker", "Smith", "John"]
13. # iterating through the content list
14. for item in content :
16. # write operation perform
17. sheet.write(row, column, item)
19. # incrementing the value of row by one with each iterations.
20. row += 1
22. book.close()

**Output:**

