Python COM Automation: win32com.client & Excel

A practical, structured reference covering modules, functions, the Excel object model, constants, examples, setup, and troubleshooting.

1) win32com.client — Modules & Key Functions

Core import:

```
import win32com.client as win32
```

Primary functions (object creation & binding):

- Dispatch(progid_or_clsid) — Create a COM object via PROGID (e.g., "Excel.Application").
- DispatchEx(progid_or_clsid) — Always creates a new instance (doesn’t attach to an existing one).
- GetActiveObject(progid) — Attach to an already running instance.
- EnsureDispatch(progid) — Like Dispatch but uses generated (early-bound) wrappers for better IntelliSense & performance.
- CastTo(obj, interface_name) — Cast a COM object to a specific interface (rare in basic Excel automation).
- WithEvents(obj, event_handler_class) — Bind COM events to a Python class (Excel has limited accessible events).

Supporting modules inside win32com.client:

- win32com.client.constants — Exposes COM constants after makepy/EnsureDispatch (e.g., xlUp, xlPasteValues).
- win32com.client.gencache — EnsureDispatch(progid); EnsureModule(clsid, lcid, major, minor) for cached wrappers.
- win32com.client.makepy — Generates early-binding Python wrappers (run as a script or via command line).
- win32com.client.dynamic — Dynamic dispatch internals (normally not directly used).
- DispatchBaseClass, CDispatch, DispatchWithEvents — base classes underpinning dispatched COM objects.

2) Excel Object Model — What You Manipulate

Hierarchy: Application → Workbooks → Workbook → Worksheets → Range / Cells / Charts / PivotTables / ListObjects

Application (Excel root object)

- Properties: Visible, DisplayAlerts, ScreenUpdating, Calculation.
- Methods: Quit().

```
import win32com.client as win32
from win32com.client import constants
excel = win32.gencache.EnsureDispatch("Excel.Application")
excel.Visible = True
excel.DisplayAlerts = False
```

Workbooks collection & Workbook

- Workbooks.Open(filepath, ReadOnly=False, UpdateLinks=0, ...); Workbooks.Add().
- Workbook.Save(), SaveAs(Filename, FileFormat=...), Close(SaveChanges=True).

- Sheets/Worksheets, Names, FullName, Path, Name.

```
wb = excel.Workbooks.Open(r"C:\data\report.xlsx")
wbNew = excel.Workbooks.Add()
wb.SaveAs(r"C:\data\report_saved.xlsx")
wb.Close(SaveChanges=True)
```

Worksheets collection & Worksheet

- Add(Before=None, After=None, Count=1, Type=None); Item(index_or_name); Delete(); Copy(); Move().
- Properties: Name, Visible, Tab.Color, UsedRange, Range, Cells, Columns, Rows.

```
ws = wb.Worksheets("Sheet1")
ws.Name = "Data"
ws2 = wb.Worksheets.Add(After=ws)
```

Range (workhorse)

- Values & formulas: Range("A1").Value; Range("A1").Formula = "=SUM(B1:B10)"; bulk write with 2D lists.
- Selection & sizing: Cells(row, col); Offset(rOffset, cOffset); Resize(rows, cols).
- Navigation: End(direction), CurrentRegion, SpecialCells(type).
- Formatting: NumberFormat, Font, Interior.Color, Borders, alignments.
- Copy/Paste: Copy(Destination), PasteSpecial(Paste=constants.xlPasteValues, ...).
- Filtering/Sorting: AutoFilter(...); Sort(...).
- Data ops: Clear(), ClearContents(), ClearFormats(), TextToColumns(), RemoveDuplicates(...).
- Find/Replace: Find(...), Replace(...).

```
rng = ws.Range("A1").CurrentRegion
last_row = ws.Cells(ws.Rows.Count, 1).End(constants.xlUp).Row
ws.Range("B2:B" + str(last_row)).NumberFormat = "0.00"
ws.Range("B2:B" + str(last_row)).Copy()
ws.Range("C2").PasteSpecial(Paste=constants.xlPasteValues)
```

Cells, Columns, Rows

- Cells(row, col) — single cell.
- Rows("2:2").Delete(); Columns("B:B").AutoFit(); Columns("C:C").ColumnWidth = 20.

Charts

- ChartObjects.Add(Left, Top, Width, Height); Charts.Add().
- Chart.SetSourceData(Source=rng); Chart.ChartType = constants.xlColumnClustered, etc.
- HasTitle, ChartTitle.Text, Axes, SeriesCollection.

```
co = ws.ChartObjects().Add(Left=100, Top=40, Width=400, Height=300)
chart = co.Chart
chart.ChartType = constants.xlLine
chart.SetSourceData(ws.Range("A1:B20"))
chart.HasTitle = True
chart.ChartTitle.Text = "Sales Trend
```

PivotTables

- PivotCaches.Create(SourceType, SourceData) → CreatePivotTable(TableDestination, TableName).
- Configure fields: PivotTables("PivotName").PivotFields("Field").Orientation = constants.xlRowField, etc.

```
pc = wb.PivotCaches().Create(SourceType=constants.xlDatabase, SourceData=ws.Range("A1").CurrentRegion
pc.CreatePivotTable(TableDestination=ws.Range("E3"), TableName="Pivot1")
pt = ws.PivotTables("Pivot1")
pf = pt.PivotFields("Category")
pf.Orientation = constants.xlRowField
pf.Position = 1
```

```
pt.AddDataField(pt.PivotFields("Amount"), "Sum of Amount", constants.xlSum)
```

Tables (ListObjects)

- ListObjects.Add(SourceType=..., Source=..., XlListObjectHasHeaders=...).
- Access DataBodyRange, HeaderRowRange; format columns, etc.

```
tbl = ws.ListObjects.Add(SourceType=constants.xlSrcRange,
                         Source=ws.Range("A1").CurrentRegion,
                         XlListObjectHasHeaders=constants.xlYes)
tbl.Name = "SalesTbl"
tbl.DataBodyRange.Columns(2).NumberFormat = "0.00
```

Names / Named Ranges

- wb.Names.Add(Name="MyRange", RefersTo="=Sheet1!$A$1:$B$10"); wb.Names("MyRange").RefersTo.

3) Common Excel Constants (after makepy/EnsureDispatch)

- Directions: xlUp, xlDown, xlToRight, xlToLeft.
- Paste: xlPasteValues, xlPasteFormats.
- Sort order: xlAscending, xlDescending.
- Chart types: xlColumnClustered, xlLine, xlPie, etc.
- Pivot: xlRowField, xlColumnField, xlDataField, xlSum.
- SpecialCells: xlCellTypeVisible, xlCellTypeConstants, xlCellTypeFormulas.
- Headers: xlYes, xlNo.

```
from win32com.client import constants
rng.End(constants.xlDown)
```

4) End-to-End Example: Open → Edit → Format → Chart → Save

```
import win32com.client as win32
from win32com.client import constants

excel = win32.gencache.EnsureDispatch("Excel.Application")
excel.Visible = True
excel.DisplayAlerts = False

wb = excel.Workbooks.Open(r"C:\data\sales.xlsx")
ws = wb.Worksheets("Sheet1")

# Write values
ws.Range("A1").Value = "Month"
ws.Range("B1").Value = "Revenue"
ws.Range("A2").Resize(6, 1).Value = [["Jan"], ["Feb"], ["Mar"], ["Apr"], ["May"], ["Jun"]]
ws.Range("B2").Resize(6, 1).Value = [[1200], [1500], [900], [1800], [2100], [2000]]

# Format
ws.Range("A1:B1").Font.Bold = True
ws.Columns("A:B").AutoFit()
ws.Range("B2:B7").NumberFormat = "#,##0"

# Chart
co = ws.ChartObjects().Add(Left=250, Top=20, Width=400, Height=300)
chart = co.Chart
chart.ChartType = constants.xlColumnClustered
chart.SetSourceData(Source=ws.Range("A1:B7"))
chart.HasTitle = True
chart.ChartTitle.Text = "Revenue by Month"

# Save & close
wb.SaveAs(r"C:\data\sales_output.xlsx")
wb.Close(SaveChanges=True)
excel.Quit()
```

5) Installation & Setup

```
pip install pywin32
python -m win32com.client.makepy  # choose Microsoft Excel Object Library for early binding
```

6) Troubleshooting Tips

• Ensure Python and Office bitness (32/64-bit) are compatible.

• GetActiveObject may fail if Excel is not registered; start Excel first or use DispatchEx.

• Performance: minimize cross-boundary calls; write 2D blocks; temporarily disable ScreenUpdating; set Calculation to manual.

• Events: Excel exposes limited application-level events; WithEvents is more common for Outlook/Word.