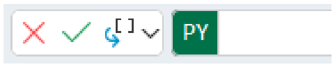
ANACONDA.

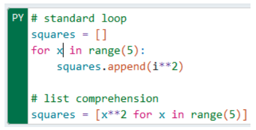
Get Started with Python in Excel

Create a Python cell



Type =PY(or Ctrl+Alt+Shift+P

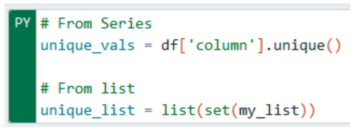
Loop over iterables



```
PY # standard loop
squares = []
for x in range(5):
    squares.append(i**2)

# list comprehension
squares = [x**2 for x in range(5)]
```

Lists, sets, series, DataFrames, dictionaries, etc.

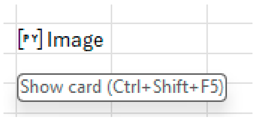
Get unique values



```
PY # From Series
unique_vals = df['column'].unique()

# From list
unique_list = list(set(my_list))
```

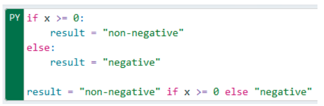
E.g. From a series or from a list

Show the Python Card View



Click [PY] or use Ctrl+Shift+F5

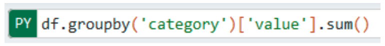
Control flow with conditional logic



```
PY if x >= 0:
    result = "non-negative"
else:
    result = "negative"

result = "non-negative" if x >= 0 else "negative"
```

Use standard if blocks or ternary operators

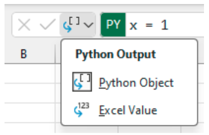
Group and summarize your data



```
PY df.groupby('category')['value'].sum()
```

Use Pandas DataFrame.groupby

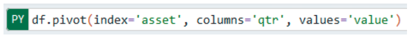
Switch Python Output mode



```
PY x = 1
```

Use drop-down or Ctrl+Alt+Shift+M

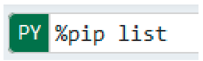
One-line pivot tables



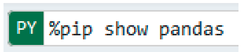
```
PY df.pivot(index='asset', columns='qtr', values='value')
```

Use DataFrame.pivot or DataFrame.pivot_table

Show information about the environment



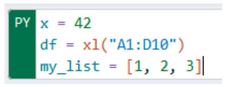
```
PY %pip list
```



```
PY %pip show pandas
```

List the available libraries or details about a specific library

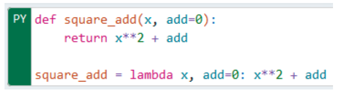
Assign values to variable



```
PY x = 42
df = xl("A1:D10")
my_list = [1, 2, 3]
```

Use a single =

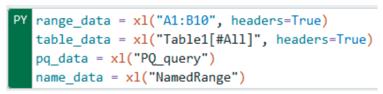
Define custom functions



```
PY def square_add(x, add=0):
    return x**2 + add

square_add = lambda x, add=0: x**2 + add
```

def functions and lambdas

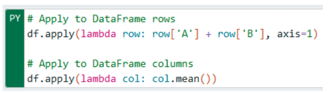
Get data with the xl function



```
PY range_data = xl("A1:B10", headers=True)
table_data = xl("Table1[#All]", headers=True)
pq_data = xl("PQ_query")
name_data = xl("NamedRange")
```

Easily create Pandas DataFrames

Apply functions to a DataFrame



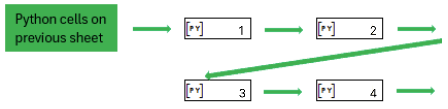
```
PY # Apply to DataFrame rows
df.apply(lambda row: row['A'] + row['B'], axis=1)

# Apply to DataFrame columns
df.apply(lambda col: col.mean())
```

Row-by-row or column-by-column

Python cells

In a Python in Excel workbook, Python cells calculate in sheet order from left to right, then on each sheet in row-major order.

The cell calculations run across a row, and then across each following row down the worksheet.



For more information on Python in Excel

Visit Microsoft support and for more information on the Anaconda package repository visit Anaconda.

Task	Recommended Libraries
Statistics	statsmodels, spicy
Machine Learning	scikit-learn, imbalanced-learn
NLP	nltk, gensim
Regression	stasmodels, scikit-learn
Time Series Analysis	statsmodels, pandas
Regular Expressions	regex, re(built-in)
Data Manipulation	pandas, numpy
Image Processing	Image pillow (PIL)
Data Visualization	seaborn, matplotlib

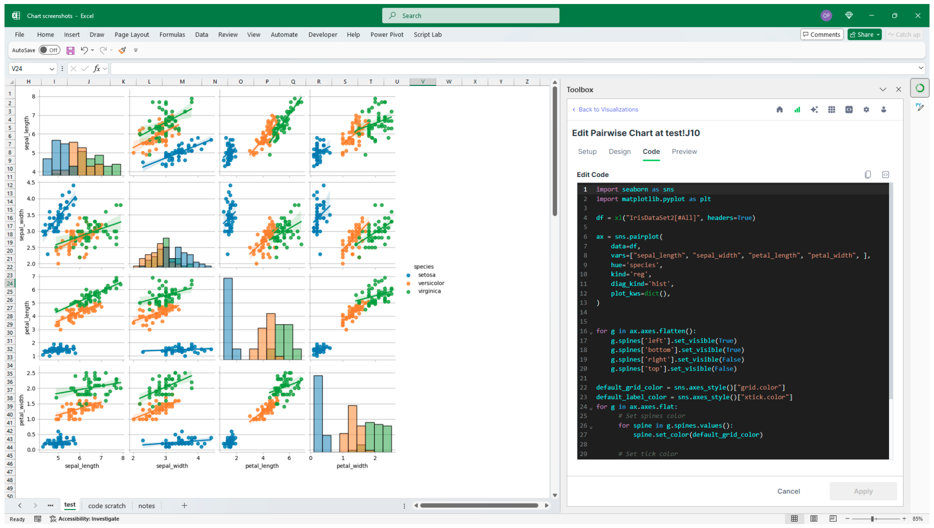
Additional Python in Excel Resources:

Anaconda's Get Started with Python in Excel Course
LinkedIn's Python in Excel for Financial Professionals
Microsoft's Introduction to Python in Excel
Anaconda Learning Courses

For more resources, check out our Resource Guide

Microsoft Support | Microsoft AppSource | Anaconda.com | Resource Guide

Create insightful visualizations with no code

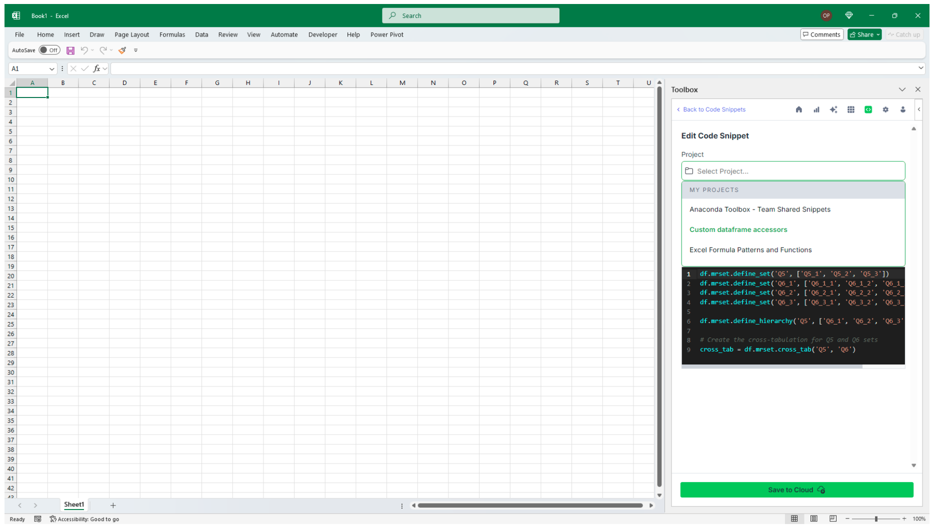


Edit Pairwise Chart at test!J10

Edit Code

```
import seaborn as sns
import matplotlib.pyplot as plt

df = xl("IrisDataSet2[#All]", headers=True)

ax = sns.pairplot(
    data=df,
    vars=["sepal_length", "sepal_width", "petal_length", "petal_width", ],
    hue='species',
    kind='reg',
    diag_kind='hist',
    plot_kws=dict(),
)


for g in ax.axes.flatten():
    g.spines['left'].set_visible(True)
    g.spines['bottom'].set_visible(True)
    g.spines['right'].set_visible(False)
    g.spines['top'].set_visible(False)

default_grid_color = sns.axes_style()["grid.color"]
default_label_color = sns.axes_style()["xtick.color"]
for g in ax.axes.flat:
      # Set spines color
      for spine in g.spines.values():
          spine.set_color(default_grid_color)

      # Set tick color
```

Organize and share your Python code snippet



Edit Code Snippet

Project: Select Project...

MY PROJECTS
- Anaconda Toolbox - Team Shared Snippets
- Custom dataframe accessors
- Excel Formula Patterns and Functions

```
df.mrset.define_set('Q5', ['Q5_1', 'Q5_2', 'Q5_3'])
df.mrset.define_set('Q6_1', ['Q6_1_1', 'Q6_1_2', 'Q6_1_
df.mrset.define_set('Q6_2', ['Q6_2_1', 'Q6_2_2', 'Q6_2_
df.mrset.define_set('Q6_3', ['Q6_3_1', 'Q6_3_2', 'Q6_3_

df.mrset.define_hierarchy('Q5', ['Q6_1', 'Q6_2', 'Q6_3'

# Create the cross-tabulation for Q5 and Q6 sets
cross_tab = df.mrset.cross_tab('Q5', 'Q6')
```

Discover Next Level Python in Excel Capabilities with Anaconda Toolbox for Excel

Anaconda Toolbox enables anyone, regardless of their Python experience level, to quickly generate code and visualizations in Microsoft Excel while learning Python along the way. By using Anaconda Toolbox, Python in Excel users can take full advantage of all the capabilities Microsoft Excel and Python have to offer. Plus, you can share data and collaborate with Python experts in Anaconda.cloud notebooks.

You can download the free add-in by visiting Microsoft AppSource.