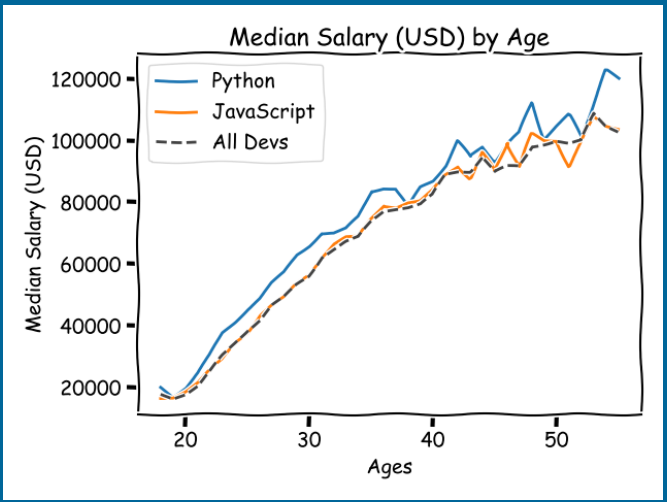
Matplotlib



from matplotlib import pyplot as plt

Customization :

plt.plot() ➔ arguments

1. list of x-values
2. list of y-values
3. **color='#444444'**:
   Specifies the color of the line in hexadecimal format. #444444 is a shade of gray. You can use any color in various formats: named colors ('blue', 'green'), hexadecimal ('#RRGGBB'), RGB tuples, etc.
4. **linestyle='--'**:
   Defines the style of the line. '--' represents a dashed line. Other options include '-' (solid line), ':' (dotted line), and '-.' (dash-dot line).
5. **label='All Devs'**:
   Provides a label for this line plot, which is used in the plot legend. This label will appear in the legend to identify this particular dataset or line in the plot.

- **plt.tight_layout()**
  ensures the layout is adjusted to fit all elements nicely.

- **plt.savefig('plot.png')**
  saves the plot to a file named plot.png.

- **plt.show()**
  displays the plot.

- **plt.legend()**
  function in Matplotlib is used to add a legend to a plot. A legend is an essential part of many plots because it helps to identify what each line, marker, or color represents.

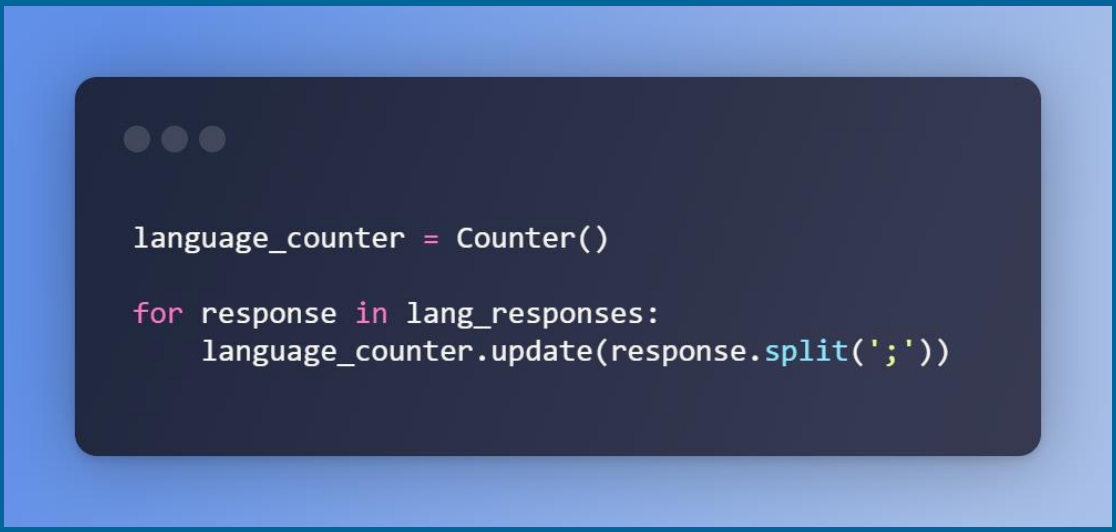
- **style.use("")**

print(plt.style.available)

To explore all available styles

BarCharts :

```
import csv
import numpy as np
import pandas as pd
from collections import Counter
from matplotlib import pyplot as plt
```



```
language_counter = Counter()

for response in lang_responses:
    language_counter.update(response.split(';'))
```

The code snippet you provided is used to count the frequency of each programming language from survey responses.

.barh
is used to create horizontal bar charts.

---

pie chart

- explode = [0, 0, 0, 0.1, 0]
  A list indicating which slices to "explode" (offset from the center) to highlight them. Here, only the slice for 'Python' is exploded by 0.1.

- **shadow=True**:
  Adds a shadow to the pie chart to give it a 3D effect.

- **startangle=90**:
  Rotates the start angle of the pie chart to make the first slice start at 90 degrees. This helps with positioning.

- **autopct='%1.1f%%'**:
  Formats the percentage label inside the pie slices. %1.1f%% means one decimal place.

- **wedgeprops={'edgecolor': 'black'}**:
  Adds a black border around each slice for better visual separation.

```
plt.pie(slices, labels=labels, explode=explode,
shadow=True,
startangle=90,
autopct='%1.1f%%',
wedgeprops={'edgecolor': 'black'})
```

Histogram :

- Plot histogram

```
plt.hist(ages, bins=bins, edgecolor='black', log=True)
```

- Plot vertical line for median

```
plt.axvline(median_age, color=color, label='Age Median', linewidth=2)
```

Subplots :

is a function used to create multiple plots in a single figure. It allows you to create a grid of subplots, making it easier to compare multiple plots or visualize different datasets within the same figure.

Key Parameters

- **nrows**: Number of rows in the subplot grid.
- **ncols**: Number of columns in the subplot grid.
- **sharex**: If True, the x-axis will be shared among all subplots.
- **sharey: If True, the y-axis will be shared among all subplots.**
- **figsize**: A tuple specifying the figure size (width, height) in inches.

Return Value

plt.subplots() returns a tuple consisting of:

1. **fig**: The Figure object that contains all the subplots.
2. **ax**: An array of Axes objects representing each subplot. If you create a single subplot (1x1 grid), it returns a single Axes object. If you create a grid, it returns an array of Axes objects.

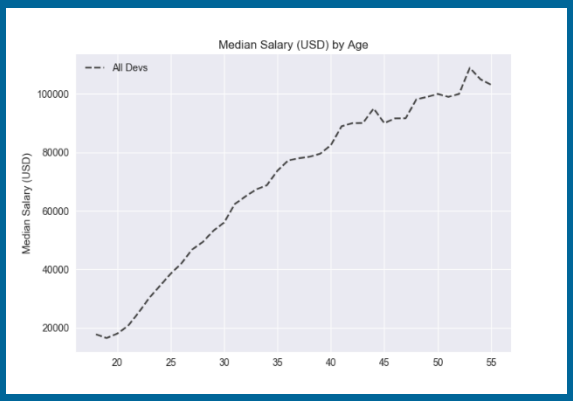
- fig1, ax1 = plt.subplots()
- fig2, ax2 = plt.subplots()
- fig, (ax1, ax2) = plt.subplots(nrows=2, ncols=1)

```
fig1, ax1 = plt.subplots()

fig2, ax2 = plt.subplots()

ax1.plot(ages, dev_salaries,
color='#444444', linestyle='--', label='All
Devs')

ax2.plot(ages, py_salaries, label='Python')
ax2.plot(ages, js_salaries, label='JavaScript')
```




```
ax1.legend()

ax1.set_title('Median Salary (USD) by Age')

ax1.set_ylabel('Median Salary (USD)')
ax2.legend() ax2.set_xlabel('Ages')
ax2.set_ylabel('Median Salary (USD)')
plt.tight_layout()

plt.show()

fig1.savefig('fig1.png')

fig2.savefig('fig2.png')
```