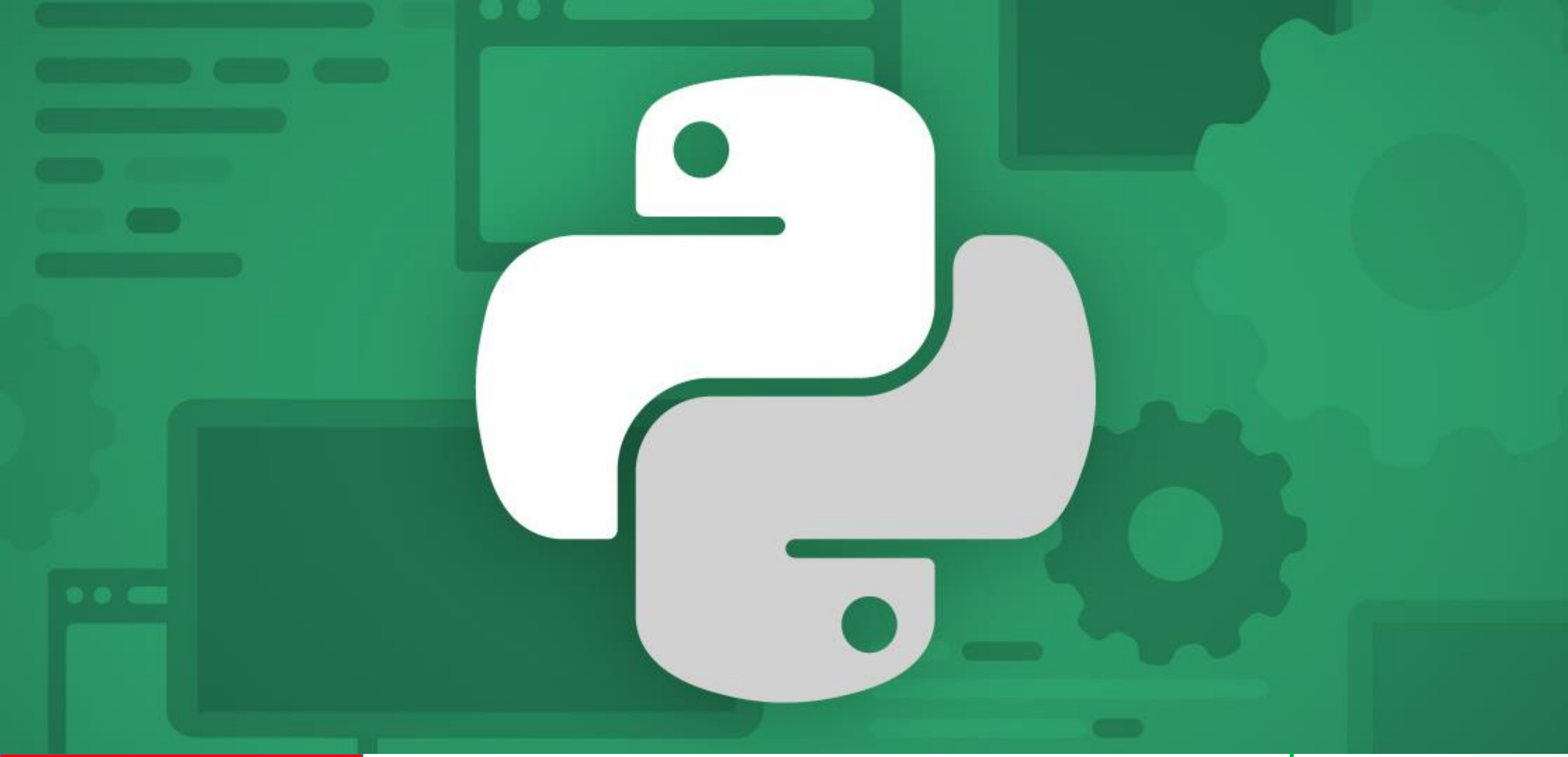

DAY 01

# DATA VISUALIZATION with Matplotlib

ENGINEER's POINT

# Matplotlib


- Matplotlib is capable of creating most kinds of charts, like line graphs, scatter plots, bar charts, pie charts, stack plots, 3D graphs, and geographic map graphs.
- Matplotlib tries to make easy things easy and hard things possible. You can generate plots, histograms, power spectra, bar charts, errorcharts, scatterplots, etc., with just a few lines of code. For example

# Plotting a basic line

- he .plot method of pyplot to plot some coordinates. This .plot takes many parameters, but the first two here are 'x' and 'y' coordinates,
- The plt.plot will "draw" this plot in the background, but we need to bring it to the screen when we're ready,

```
import matplotlib.pyplot as plt

plt.plot([1,2,3],[5,7,4])
plt.show()
```

# Legends, Titles, and Labels with Matplotlib

- label allows us to assign a name to the line, which we can later show in the legend.
- With plt.xlabel and plt.ylabel, we can assign labels to those respective axis.
- Next, we can assign the plot's title with plt.title, and then we can invoke the default legend with plt.legend().

```
plt.plot(x, y, label='First Line')
plt.plot(x2, y2, label='Second Line')
plt.xlabel('Plot Number')
plt.ylabel('Important var')
plt.title('graph1')
plt.legend()
plt.show()
```

# Bar Charts and Histograms

- The plt.bar creates the bar charts.
- You can use color to color just about any kind of plot, using colors like g for green, b for blue, r for red, and so on. You can also use hex color codes, like #191970

```
import matplotlib.pyplot as plt
plt.bar([1,3,5,7,9],[5,2,7,8,2], label="Example one")
plt.bar([2,4,6,8,10],[8,6,2,5,6], label="Example two",
color='g')
plt.legend() plt.xlabel('bar number')
plt.ylabel('bar height')
plt.title(‘bar graph')
plt.show()
```

# Matplotlib customization options

| character | color |
|---|---|
| ‘b’ | blue |
| ‘g’ | green |
| ‘r’ | red |
| ‘c’ | cyan |
| ‘m’ | magenta |
| ‘y’ | yellow |
| ‘k’ | black |
| ‘w’ | white |

| character | description |
|---|---|
| '-' | solid line style |
| '--' | dashed line style |
| '-.' | dash-dot line style |
| ':' | dotted line style |
| '.' | point marker |
| ',' | pixel marker |
| 'o' | circle marker |
| 'v' | triangle_down marker |
| '^' | triangle_up marker |
| '<' | triangle_left marker |
| '>' | triangle_right marker |
| '1' | tri_down marker |
| '2' | tri_up marker |
| '3' | tri_left marker |
| '4' | tri_right marker |
| 's' | square marker |
| 'p' | pentagon marker |
| '*' | star marker |
| 'h' | hexagon1 marker |
| 'H' | hexagon2 marker |
| '+' | plus marker |
| 'x' | x marker |
| 'D' | diamond marker |
| 'd' | thin_diamond marker |
| '\|' | vline marker |
| '_' | hline marker |

# Histograms

- Histograms are very much like a bar chart, tend to show distribution by grouping segments together. Examples of this might be age groups, or scores on a test.
- Rather than showing every single value, this shows a range of values

# Scatter Plots

- Scatter plots are usually to compare two variables, or three if you are plotting in 3 dimensions, looking for correlation or groups.

```
import matplotlib.pyplot as plt
x = [1,2,3,4,5,6,7,8]
y = [5,2,4,2,1,4,5,2]
plt.scatter(x,y, label='skitscat',
color='k', s=25, marker="o")
plt.xlabel('x')
plt.ylabel('y')
plt.title('Scatter Plot')
plt.legend()
plt.show()
```

# Pie Charts

- Pie Chart is used to show parts to the whole, and often a % share.
- Matplotlib handles the sizes of the slices and everything, we just feed it the numbers.

```
import matplotlib.pyplot as plt
slices = [7,2,2,13]
activities = ['sleeping','eating','working','playing']
cols = ['c','m','r','b']
plt.pie(slices,
labels=activities,
colors=cols,
startangle=90,
shadow= True,
explode=(0,0.1,0,0),
autopct='%1.1f%%')
plt.title('Pie Charts')
plt.show()
```