```python
# === aesthetics.ipynb ===

#%%
#matplotlib notebook
#get_ipython().run_line_magic('matplotlib', 'notebook')


#%%
# Dependencies
import matplotlib.pyplot as plt
import numpy as np


#%%
# Generate the x values from 0 to 10 using a step of 0.1
x_axis = np.arange(0, 10, 0.1)
sin = np.sin(x_axis)
cos = np.cos(x_axis)


#%%
# Add a semi-transparent horizontal line at y = 0
plt.hlines(0, 0, 10, alpha=0.25)


#%%
# Use dots or other markers for your plots, and change their colors
plt.plot(x_axis, sin, linewidth=0, marker="o", color="blue")
plt.plot(x_axis, cos, linewidth=0, marker="^", color="red")


#%%
# Add labels to the x and y axes
# ------------------------------------------------------------------------
# Adding labels ensures the graphic remains honest and easy to understand,
# even in cases where the visualization is not immediately transparent to
# most viewers.
# .title(), .xlabel() and .ylable() functions take a string as argument
# ------------------------------------------------------------------------
plt.title("Juxtaposed Sine and Cosine Curves")
plt.xlabel("Input (Sampled Real Numbers from 0 to 10)")
plt.ylabel("Value of Sine (blue) and Cosine (red)")


#%%
# ------------------------------------------------------------------------
# Set your x and y limits
# .xlim() and .ylim() are used to set where the axes for the chart should
# begin/end. MatplotLib will naturally create charts with a lot of empty
# space and these methods can help to limit that
# Note: might not be able to control the distance between the tick marks
# ------------------------------------------------------------------------
plt.xlim(0, 10)
plt.ylim(-1, 1)

```

```
56
57 #%%
58 # Set a grid on the plot (very self-explainatory)
59 plt.grid()
60
61
62 #%%
63 # Save the plot and display it
64 plt.savefig("../Images/sin_cos_with_markers.png")
65 plt.show()
66
67
68 #%%
69
70
71
72
```