**Q1: How can you create a Bokeh plot using Python code?**

To create a Bokeh plot using Python, follow these steps:

1. **Import the required modules:** Use bokeh.plotting for basic plotting.
2. **Create a figure:** Use figure() to create a blank canvas.
3. **Add glyphs:** Use glyph methods (e.g., circle(), line()) to add data to the figure.
4. **Show or save the plot:** Use show() to display the plot in a browser or save() to save it.

Example:

from bokeh.plotting import figure, show

# Create a figure

p = figure(title="Simple Bokeh Plot", x\_axis\_label='X', y\_axis\_label='Y')

# Add glyphs

p.line([1, 2, 3, 4, 5], [6, 7, 2, 4, 5], legend\_label="Line", line\_width=2)

# Show the plot

show(p)

**Q2: What are glyphs in Bokeh, and how can you add them to a Bokeh plot?**

**Glyphs** are visual shapes used to represent data in a Bokeh plot, such as circles, lines, rectangles, and bars. They are added to a figure to visualize data.

Example:

from bokeh.plotting import figure, show

# Create a figure

p = figure(title="Glyphs Example", x\_axis\_label='X', y\_axis\_label='Y')

# Add glyphs

p.circle([1, 2, 3, 4], [4, 7, 2, 6], size=10, color="blue", legend\_label="Circles")

p.line([1, 2, 3, 4], [4, 7, 2, 6], line\_width=2, legend\_label="Line")

# Show the plot

show(p)

**Q3: How can you customize the appearance of a Bokeh plot, including the axes, title, and legend?**

Bokeh provides various options for customization:

* **Title:** Use title.text to modify the title text.
* **Axes:** Customize labels, tick marks, and styles using axis\_label, axis\_line\_color, etc.
* **Legend:** Modify position, background, and other properties using legend attributes.

Example:

from bokeh.plotting import figure, show

# Create a figure

p = figure(title="Customized Bokeh Plot", x\_axis\_label='X-axis', y\_axis\_label='Y-axis')

# Add glyphs

p.circle([1, 2, 3, 4], [4, 7, 2, 6], size=10, color="blue", legend\_label="Circles")

# Customize title

p.title.text = "My Custom Title"

p.title.text\_color = "green"

p.title.text\_font\_size = "20px"

# Customize axes

p.xaxis.axis\_label = "Custom X-axis"

p.yaxis.axis\_label = "Custom Y-axis"

p.xaxis.major\_label\_text\_color = "red"

# Customize legend

p.legend.location = "top\_left"

p.legend.background\_fill\_color = "lightgray"

# Show the plot

show(p)

**Q4: What is a Bokeh server, and how can you use it to create interactive plots that can be updated in real-time?**

A **Bokeh server** enables the creation of interactive web applications with real-time data updates. It integrates Python callbacks with a web interface.

Steps:

1. **Create a script with a Document object.**
2. **Use Python callbacks** to update data in response to user interactions.
3. **Run the Bokeh server** to serve the application.

Example:

from bokeh.plotting import figure, curdoc

from bokeh.models import ColumnDataSource

from bokeh.layouts import column

from bokeh.models import Slider

# Data source

source = ColumnDataSource(data=dict(x=[1, 2, 3, 4], y=[4, 7, 2, 6]))

# Create a figure

p = figure(title="Bokeh Server Example")

p.circle('x', 'y', source=source, size=10, color="blue")

# Slider callback

def update(attr, old, new):

scale = slider.value

source.data = dict(x=[1, 2, 3, 4], y=[i \* scale for i in [4, 7, 2, 6]])

# Add slider

slider = Slider(start=0.1, end=3, step=0.1, value=1, title="Scale")

slider.on\_change('value', update)

# Add to document

curdoc().add\_root(column(slider, p))

Run the server:

bokeh serve --show script\_name.py

**Q5: How can you embed a Bokeh plot into a web page or dashboard using Flask or Django?**

You can embed a Bokeh plot into a web application using the components function to generate HTML and JavaScript.

**Using Flask:**

from flask import Flask, render\_template

from bokeh.plotting import figure

from bokeh.embed import components

app = Flask(\_\_name\_\_)

@app.route('/')

def index():

# Create a plot

p = figure(title="Flask Example")

p.line([1, 2, 3, 4], [6, 7, 2, 4], line\_width=2)

# Embed the plot

script, div = components(p)

return render\_template('index.html', script=script, div=div)

if \_\_name\_\_ == '\_\_main\_\_':

app.run(debug=True)

**Using Django:**

1. Create a view that generates the Bokeh plot:

from django.shortcuts import render

from bokeh.plotting import figure

from bokeh.embed import components

def bokeh\_plot(request):

# Create a plot

p = figure(title="Django Example")

p.line([1, 2, 3, 4], [6, 7, 2, 4], line\_width=2)

# Embed the plot

script, div = components(p)

return render(request, 'bokeh\_plot.html', {'script': script, 'div': div})

1. Create a template (bokeh\_plot.html):

<!DOCTYPE html>

<html>

<head>

<title>Bokeh Plot</title>

<script src="https://cdn.bokeh.org/bokeh/release/bokeh-2.4.0.min.js"></script>

</head>

<body>

<h1>Bokeh Plot Embedded</h1>

{{ script|safe }}

{{ div|safe }}

</body>

</html>