🏠 Architectural Model Generator

📝 Describe your building style

A cool skyscraper in modernist style, inspired by the architecture of Shanghai's Bund, Shrink inward fro

🛠️ Generate Parameters

🔧 Adjust Parameters

totalHeight

300.00

200.00 500.00

baseWidth

50.00

30.00 70.00

layerHeight

3.00

2.00 5.00

shrinkFactor

0.95

0.90 0.98

layersPerShrink

10.00

5.00 15.00

Get Rhino Code 💻

Rhino Modeling Steps

To use a generated Rhino Python script for modeling, follow these simple steps. This guide assumes you have a basic understanding of using Rhino 3D but no prior experience with scripting in Rhino.

Step 1: Open Rhino and Access the Python Script Editor

1. **Launch Rhino 3D**: Open the Rhino application on your computer.
2. **Open the Python Script Editor:** Use the shortcut `Alt` + `Shift` + `P` to open the Python script editor in Rhino. Alternatively, you can type `EditPythonScript` in the command line and press `Enter`.

Step 2: Create or Open the Python Script

1. **Create a New Script**: In the Python script editor, click on `File` > `New` to create a new script file.
2. **Copy the Script**: Paste the generated Python script into the editor. This script will use the parameters provided in JSON format to create a model.

Step 3: Understand the Script Parameters

The script uses the following parameters to generate a model:

- `totalHeight`: The total height of the model (300.0 units).
- `baseWidth`: The width of the base layer (50.0 units).
- `layerHeight`: The height of each layer (3.0 units).
- `shrinkFactor`: The factor by which each layer shrinks (0.95).
- `layersPerShrink`: The number of layers before applying the shrink factor (10.0).

Step 4: Run the Script

1. **Execute the Script**: Click on the `Run` button (green triangle) in the Python script editor to execute the script.
2. **Observe the Model**: The script will generate a 3D model in the Rhino workspace based on the parameters provided.

Step 5: Troubleshoot and Adjust

1. **Check for Errors**: If there are any errors in the script, they will be displayed in the output panel of the Python script editor.
2. **Provide Feedback**: Return here to describe any issues or errors you encounter. You can adjust the script through a conversation with the AI to resolve any problems.

By following these steps, you should be able to use the generated Rhino Python script to model effectively. If you encounter any issues, feel free to ask for further assistance.

Rhino Python Code

✅ Script auto-saved at: `C:\Users\heyus/Desktop/RhinoScripts\rhino_model_2025-05-25_10-46-13_PM.py`

```
def create_pyramid(totalHeight, baseWidth, layerHeight, shrinkFactor, layersPerShr
    import rhinoscriptsyntax as rs

    def create_layer(center, width, height):
        half_width = width / 2.0
        points = [
            rs.PointAdd(center, [-half_width, -half_width, 0]),
            rs.PointAdd(center, [half_width, -half_width, 0]),
            rs.PointAdd(center, [half_width, half_width, 0]),
            rs.PointAdd(center, [-half_width, half_width, 0]),
            rs.PointAdd(center, [-half_width, -half_width, 0])
        ]
        return rs.AddPolyline(points)
    current_height = 0.0
    current_width = baseWidth
    center = [0, 0, 0]
    layers = int(totalHeight / layerHeight)

    for i in range(layers):
        layer = create_layer(center, current_width, current_height)
        rs.MoveObject(layer, [0, 0, current_height])
        current_height += layerHeight

        if (i + 1) % layersPerShrink == 0:
            current_width *= shrinkFactor

    rs.AddTextDot("Pyramid Complete", [0, 0, totalHeight])

create_pyramid(300.0, 50.0, 3.0, 0.95, 10.0)
```

Code Feedback and Adjustment

Want to adjust the code or got an Error? Enter your query here:

the shape is make sense but it didn't give me 4 facades, just lines , fix this

Improve Code

📝 Summary of Changes

- The `create_layer` function was modified to remove the `height` parameter, as it was not being used in the function's logic.
- A list named `layer_curves` was introduced to store each layer's polyline, allowing for the creation of a lofted surface from these curves.
- A loft operation (`rs.AddLoftSrf`) was added to create surfaces between the stored layer curves, enhancing the 3D representation of the pyramid.

🧾 Modified Rhino Script

✅ Modified script auto-saved at: `C:\Users\heyus/Desktop/RhinoScripts\rhino_model_2025-05-25_10-46-21_PM.py`

```
def create_pyramid(totalHeight, baseWidth, layerHeight, shrinkFactor, layersPerShr
    import rhinoscriptsyntax as rs

    def create_layer(center, width):
        half_width = width / 2.0
        points = [
            rs.PointAdd(center, [-half_width, -half_width, 0]),
            rs.PointAdd(center, [half_width, -half_width, 0]),
            rs.PointAdd(center, [half_width, half_width, 0]),
            rs.PointAdd(center, [-half_width, half_width, 0]),
            rs.PointAdd(center, [-half_width, -half_width, 0])
        ]
        return rs.AddPolyline(points)

    current_height = 0.0
    current_width = baseWidth
    center = [0, 0, 0]
    layers = int(totalHeight / layerHeight)
    layer_curves = []

    for i in range(layers):
        layer_curve = create_layer(center, current_width)
        rs.MoveObject(layer_curve, [0, 0, current_height])
        layer_curves.append(layer_curve)
        current_height += layerHeight

        if (i + 1) % layersPerShrink == 0:
            current_width *= shrinkFactor

    # Loft the curves to create surfaces
    if len(layer_curves) > 1:
        rs.AddLoftSrf(layer_curves)

    # Optionally, add a text dot to indicate completion
    rs.AddTextDot("Pyramid Complete", [0, 0, totalHeight])

create_pyramid(300.0, 50.0, 3.0, 0.95, 10)
```