Manual básico del código (Blender + Python)

¿Qué hace este programa?

- ✓ Borra todos los objetos de la escena
- ✓ Calcula los puntos de un polígono regular
- ✓ Crea una malla en Blender
- ✓ Dibuja un polígono 2D con el número de lados que tú quieras

1. Importación de librerías

```
1  import bpy
2  import math
```

- import bpy
- bpy es la librería de Blender
- Permite:
  - Crear objetos
  - Crear mallas
  - Borrar cosas
  - Manipular la escena

Sin bpy, no puedes controlar Blender con código

- import math
- math es una librería de Python para matemáticas
- Aquí se usa para:

  sin() → seno

  cos() → coseno

  pi → π (3.1416...)

Se necesita para calcular la forma circular del polígono

2. Definición de la función

```
3  def crear_poligono_2d(nombre, lados, radio):
```

¿Qué es esto?

Es una función llamada crear_poligono_2d.

Parámetros:

nombre → nombre del objeto en Blender

lados → cuántos lados tendrá el polígono

radio → tamaño del polígono

Ejemplo mental:

lados = 3 → triángulo

lados = 4 → cuadrado

lados = 6 → hexágono

3. Creación de la malla y el objeto

```
4     malla=bpy.data.meshes.new(nombre)
5     objeto=bpy.data.objects.new(nombre, malla)
```

- meshes.new
- Crea una malla vacía
- Aún no tiene forma
- objects.new
- Crea un objeto usando esa malla
- Es lo que aparece en la escena

La malla es la forma, el objeto es lo visible

4. Añadir el objeto a la escena

```
7     bpy.context.collection.objects.link(objeto)
```

- Esto hace que el objeto:

  Sea visible

  Aparezca en la colección actual

Si no haces esto, el objeto existe pero no se ve

5. Listas para vértices y aristas

¿Qué son?

- vértices
- Lista de puntos (x, y, z)
- Son las esquinas del polígono
- aristas
- Conectan vértices
- Son las líneas del polígono

6. Cálculo de los vértices (la parte matemática)

```
for i in range(lados):
```

- Este ciclo se repite tantas veces como lados tenga el polígono

```
angulo=2 * math.pi*i/lados
```

¿Qué hace?

- Divide el círculo completo (360°) en partes iguales
- Cada parte corresponde a un vértice

Esto garantiza que el polígono sea regular

```
14          x= radio * math.cos(angulo)
15          y= radio * math.sin(angulo)
```

- Usa trigonometría para colocar los puntos en círculo
- radio controla el tamaño

```
16          vertices.append((x, y, 0))
```

- Agrega el vértice
- z = 0 → es un polígono 2D

7. Creación de las aristas

```
18      for i in range(lados):
19          aristas.append((i, (i + 1) % lados))
```

¿Qué significa esto?

- Conecta:
- Vértice i con el siguiente
- % lados sirve para:
- Conectar el último vértice con el primero

Así el polígono se cierra

8 Crear la geometría real

```
22      malla.from_pydata(vertices, aristas, [])
```

Esta línea es CLAVE:

- Usa los datos:
- vértices
- aristas
- No se crean caras ([])

Por eso es solo un contorno 2D

```
23      malla.update()
```

- Actualiza la malla
- Hace visibles los cambios

9. Limpiar la escena

```
26  bpy.ops.object.select_all(action='SELECT')
27  bpy.ops.object.delete()
```

¿Qué hace?

- Selecciona todos los objetos
- Los borra

Evita que se acumulen objetos viejos

10. Llamada a la función

```
29  crear_poligono_2d("Poligono2D", lados=6, radio=5)
```

Esto crea:

- Un objeto llamado Poligono2D
- Con 6 lados
- Tamaño radio = 5

Resultado: un hexágono 2D