POLIGONO EN BLENDER

¿Qué es esta práctica?

Esta práctica consiste en dibujar un polígono 2D usando código en Blender, en lugar de dibujarlo a mano con el mouse.

Usamos Python y la API de Blender (bpy) para:

- Crear una malla
- Calcular vértices matemáticamente
- Conectar los vértices con aristas
- Mostrar el polígono en la escena

Al final, el código crea un hexágono (6 lados) plano.

Instalación de Blender en Mac

Dado que tienes un procesador M4 (Apple Silicon), es crucial descargar la versión optimizada para obtener el máximo rendimiento.

1. **Ve al sitio oficial:** Entra a blender.org/download.
2. **Selecciona la versión correcta:**
    - La página suele detectar tu sistema automáticamente, pero asegúrate de que diga macOS Apple Silicon (no Intel).
    - Si ves un menú desplegable, elige "macOS (Apple Silicon)".
3. **Descargar e Instalar:**
    - Descarga el archivo .dmg.

- Ábrelo y arrastra el icono de Blender naranja a la carpeta de Applications (Aplicaciones).

4. **Primer inicio:**
    - Abre Blender desde tus aplicaciones. Al ser la primera vez, macOS te preguntará si confías en la app. Dile que sí ("Open").
    - En la pantalla de bienvenida, puedes dejar la configuración por defecto y hacer clic en cualquier parte vacía para cerrar el menú.

Preparando el entorno de trabajo

Blender tiene una interfaz compleja, pero para lo que haremos solo necesitas el área de programación.

1. En la barra superior de menús, verás pestañas como "Layout", "Modeling", etc. Busca y haz clic en la pestaña Scripting.
2. Verás que la interfaz cambia. La ventana más grande (generalmente a la derecha o al centro) es el Editor de Texto.
3. Haz clic en el botón New (Nuevo) en la parte superior de ese editor para crear un archivo de texto en blanco donde pegaremos tu código.

Código Explicado

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

¿Qué hace?

- bpy: permite controlar Blender con Python
- math: permite usar funciones matemáticas (seno, coseno, π)

Función para crear el polígono

```
4  def crear_poligono_2d(nombre, lados, radio):
5      """
6      Crea un polígono regular plano en el origen (0,0,0).
7      Parámetros:
8        - nombre: Nombre del objeto en Blender.
9        - lados: Cantidad de vértices (ej. 6 para hexágono).
10       - radio: Distancia del centro a los vértices.
11     """
```

Explicación

Esta función:

- nombre: nombre del objeto en Blender
- lados: número de lados del polígono
- radio: tamaño del polígono

Ejemplo:

- 3 lados → triángulo
- 4 lados → cuadrado
- 6 lados → hexágono

Crear la malla y el objeto

```
13     # 1. CREACIÓN DE DATOS (Malla y Objeto)
14     # Creamos una malla vacía (la estructura de puntos)
15     malla = bpy.data.meshes.new(nombre)
16     # Creamos el objeto que contendrá esa malla
17     objeto = bpy.data.objects.new(nombre, malla)
```

Explicación

- **Malla**: estructura geométrica (vértices y aristas)
- **Objeto**: lo que aparece en la escena

Agregar el objeto a la escena

```
19      # 2. VINCULACIÓN
20      # Para que el objeto sea visible, debemos ponerlo en la colección de la escena
21      bpy.context.collection.objects.link(objeto)
```

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

Listas para vértices y aristas

```
23      vertices = []
24      aristas = [] # Las líneas que unen los puntos
```

- vertices: puntos del polígono
- aristas: líneas que conectan los puntos

Cálculo de los vértices (parte matemática)

```
26      # 3. CÁLCULO MATEMÁTICO (De Polar a Cartesiano)
27      # Iteramos tantas veces como lados tenga el polígono
28      for i in range(lados):
29          # Calculamos el ángulo en radianes para este vértice específico.
30          # 2*pi es un círculo completo (360 grados).
31          angulo = 2 * math.pi * i / lados
32
33          # Usamos trigonometría básica para encontrar la posición X y Y
34          x = radio * math.cos(angulo)
35          y = radio * math.sin(angulo)
36
37          # Agregamos la coordenada (x, y, 0). Z es 0 porque es 2D.
38          vertices.append((x, y, 0))
39
```

Explicación simple

- El círculo completo tiene 360° o 2π
- Se divide entre el número de lados
- Se calculan coordenadas usando:
    - coseno → eje X
    - seno → eje Y
- Z = 0 → figura plana (2D)

Esto coloca los puntos uniformemente en un círculo

Crear las aristas (líneas)

```
40      # 4. DEFINIR CONEXIONES (Aristas)
41      # Unimos el vértice actual 'i' con el siguiente '(i+1)'
42      # El operador módulo (%) sirve para que el último punto se una con el primero (0)
43      for i in range(lados):
44          aristas.append((i, (i + 1) % lados))
```

Explicación

- Conecta cada vértice con el siguiente
- % lados hace que el último se conecte con el primero
- Así se cierra el polígono 🔒

Cargar datos en la malla

```
46      # 5. CONSTRUCCIÓN FINAL
47      # Le decimos a la malla: "Usa estos vértices y estas aristas"
48      # El tercer parámetro [] es para caras (faces), que por ahora dejamos vacío.
49      malla.from_pydata(vertices, aristas, [])
50
51      # Actualizamos la geometría para que Blender valide los cambios
52      malla.update()
```

Aquí Blender recibe:

- Vértices
- Aristas
- (No hay caras, por eso está vacío [])

Limpiar la escena antes de dibujar

```
54  # --- EJECUCIÓN DEL SCRIPT ---
55
56  # A. LIMPIEZA DE ESCENA
57  # Seleccionamos todo lo que existe en la escena actual
58  bpy.ops.object.select_all(action='SELECT')
59  # Lo borramos (Cuidado: esto borra todo tu trabajo previo en la escena)
60  bpy.ops.object.delete()
```

Esto:

- Selecciona todo
- Borra todo
- Evita que se acumulen figuras

Llamar a la función (crear el polígono)

```
62  # B. LLAMADA A LA FUNCIÓN
63  # Creamos un hexágono (6 lados) de radio 5
64  crear_poligono_2d("Poligono2D", lados=6, radio=5)
```

Resultado:

- Nombre: Poligono2D
- 6 lados → hexágono
- Tamaño: radio 5

Ejecutar el código

1. En el Editor de texto
2. Presiona:
    - ▶ **Run Script**
    - o ⌥ Option + P