Graficador — Documento Técnico

Kevin Esguerra Cardona

23 de abril de 2025

Índice

1. Introducción

El proyecto **Graficador** es un editor 2D académico desarrollado en Python 3 que permite crear, modificar y exportar figuras geométricas básicas mediante dos familias de algoritmos de rasterizado: *BASIC* (implementaciones manuales para fines didácticos) y *PYGAME* (primitivas nativas de la librería pygame). El código está organizado según el patrón **Modelo–Vista–Controlador (MVC)**, lo cual facilita la mantenibilidad y la extensibilidad del sistema.

- Repositorio: `https://github.com/porgetit/CG-Graficador` *(ejemplo)*
- Lenguaje: Python 3.9 o superior
- Dependencias: pygame, numpy, matplotlib, tkinter (estándar)

2. Modelo

2.1. Canvas

- Mantiene la lista ordenada de objetos `Shape`.
- Serializa/deserializa la escena en JSON (`to_json`, `load_json`).
- Almacena el color de fondo.

2.2. Jerarquía Shape

`Shape` es una clase abstracta con subclases `Line`, `Rectangle`, `Circle`, `Polygon`, `Curve`, `EraseArea`. Cada instancia delega el rasterizado a un objeto `DrawingAlgorithm`.

2.3. Algoritmos de dibujo

- **BASIC**: `DDADrawingAlgorithm`, `MidpointCircleAlgorithm`, `BasicRectangleAlgorithm`, `BasicPolygonAlgorithm`, `BezierCurveAlgorithm`, `EraseAreaAlgorithm`.
- **PYGAME**: `PygameDrawingAlgorithm` (usa primitivas nativas).

3. Controladores

SuperController Punto de orquestación; instancia `DrawingController` y `EventHandler`.

EventHandler Encapsula el reenvío de eventos Pygame hacia `DrawingController`.

DrawingController Gestiona la lógica de la herramienta actual, selección de algoritmo, color, grosor y acciones de archivo.

4. Vistas

CanvasView Pinta el lienzo y recorre la lista de figuras.

ToolbarView Presenta botones con iconos para herramientas, algoritmos, colores y acciones de archivo.

ColorPickerModal Diálogo tkinter para elegir color y grosor.

5. Diagramas

5.1. Diagrama de clases

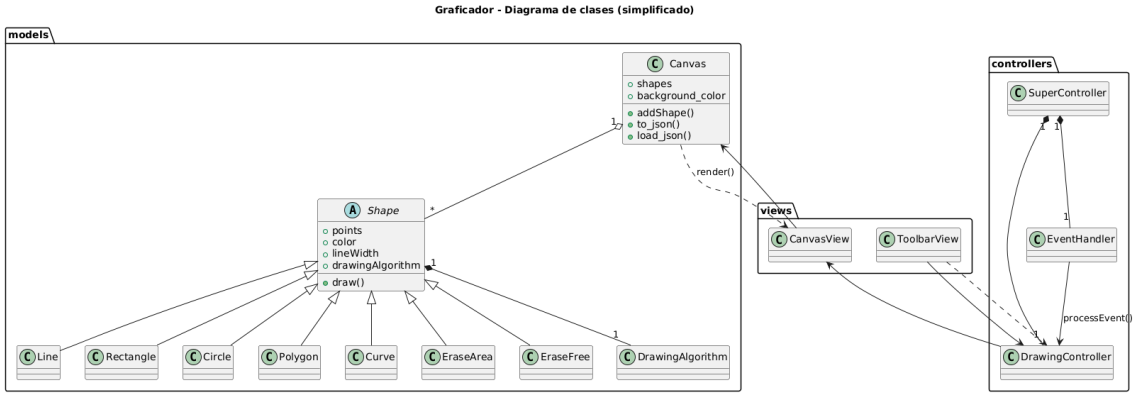

Figura 1: Diagrama de clases (vista parcial)

5.2. Diagrama de secuencia: «Dibujar línea»

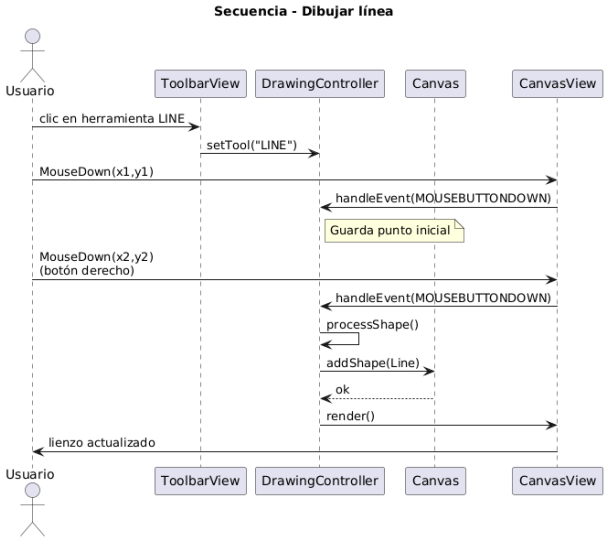

Figura 2: Diagrama de secuencia para el escenario «dibujar línea»

5.3. Diagrama de flujo general

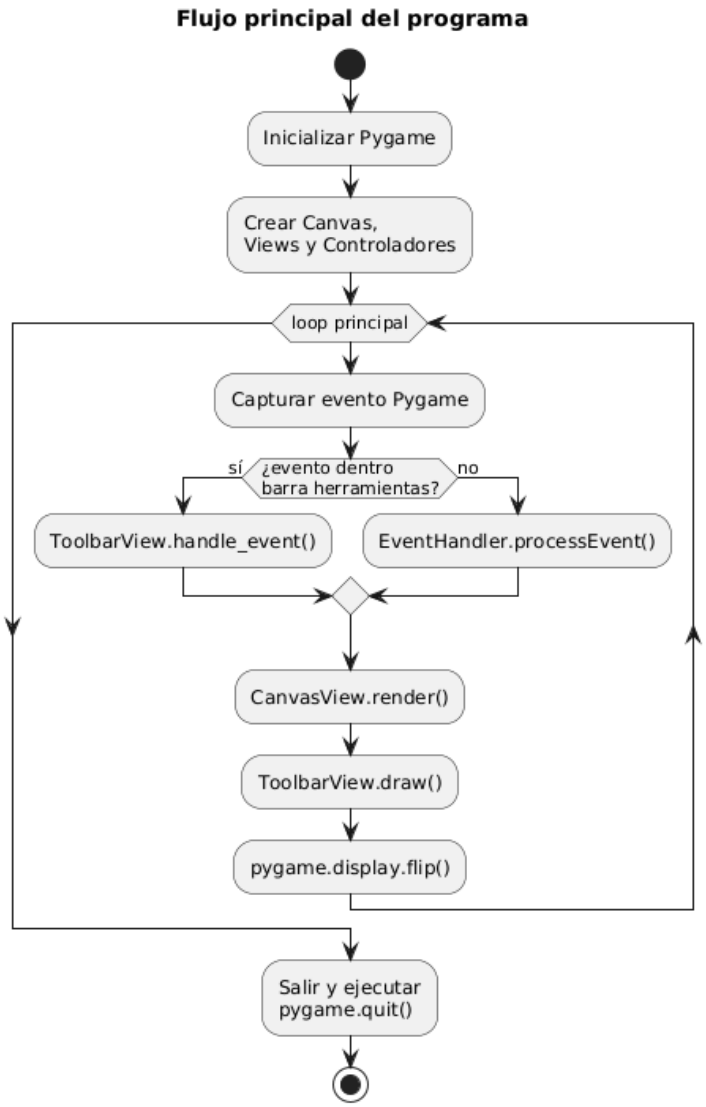

Figura 3: Flujo de ejecución en `main.py`

6. Guía de instalación

1. Crear entorno virtual: `python -m venv venv`
2. Instalar dependencias:
   `pip install pygame numpy matplotlib`
3. Ejecutar `python main.py`.

7. Convenciones de código

- Estilo PEP 8 reforzado con black.

- Separación estricta MVC: *modelo* sin dependencias PyGame, *vista* sin lógica de negocio, *controlador* como mediador puro.

8. Conclusiones y trabajo futuro

El **Graficador** cumple su objetivo pedagógico al ilustrar dos aproximaciones de rasterizado y al mantener la lógica desacoplada en capas bien definidas. Para iteraciones futuras se propone:

1. Implementar *Undo/Redo* con pilas de *snapshots* del `Canvas`.
2. Borrado real de figuras mediante filtrado de *bounding boxes*.
3. Herramientas de selección y transformación (mover, escalar, rotar).