Documentación Técnica del Proyecto
Juego más fácil del mundo

Juan David Mayorga López – 20232020116
Samuel Andrés Barrera Pulido – 20232020156
Mariam Betin Escobar – 20232020300

Para la clase de Modelos de Programación
Profesor: Alejandro Daza
October 29, 2025

Contents

1 Resumen del Proyecto

Este proyecto implementa un pequeño juego donde el jugador controla un cubo que debe esquivar bolas que caen desde la parte superior de la pantalla. El objetivo es sobrevivir el mayor tiempo posible sin ser golpeado por ninguna bola.

El juego fue desarrollado en Python utilizando la librería Pygame, y tiene como propósito aplicar los patrones de diseño Command y Chain of Responsibility, integrándolos en una estructura simple pero funcional.

2 Arquitectura General

El proyecto está organizado en diferentes clases y componentes, que interactúan para gestionar el estado del juego:

- **Clase** `Game`**:** Controla el bucle principal del juego, las actualizaciones de pantalla, la interacción entre la serpiente, la comida y el marcador. Gestiona los decoradores que modifican el aspecto visual.
- **Clase** `JuegoSimple`**:** Clase principal que coordina la ejecución del juego.
- **Clase** `Player`**:** Representa el cubo que se mueve por la pantalla.
- **Clase** `Enemy`**:** Representa las bolas que caen desde la parte superior.
- **Clase** `Command`**:** Define la estructura de los comandos.
- **Clases** `MoverUp, MoverDown, MoverLeft, MoverRight`: Comandos concretos para mover al jugador.
- **Clase** `InputHadler`**:** Clase base del patrón Chain of Responsibility.

3 Patrones de Diseño Utilizados

El proyecto utiliza principalmente el patron **Command,** el cual se utiliza para desacoplar los controles del jugador de la lógica de movimiento. Cada tecla o acción del jugador se traduce en un comando, y el jugador ejecuta ese comando sin saber de dónde proviene la orden.

Adicionalmente esta el patrón **Chain of Responsibility** que se implementó para manejar las colisiones, límites y otros eventos del juego en una cadena. Cada manejador tiene una responsabilidad específica, y si no puede procesar el evento, lo pasa al siguiente en la cadena. De esta forma, se evita tener una función monolítica que controle todo, y se logra una estructura más limpia y ordenada.

4 Descripción de Funcionamiento

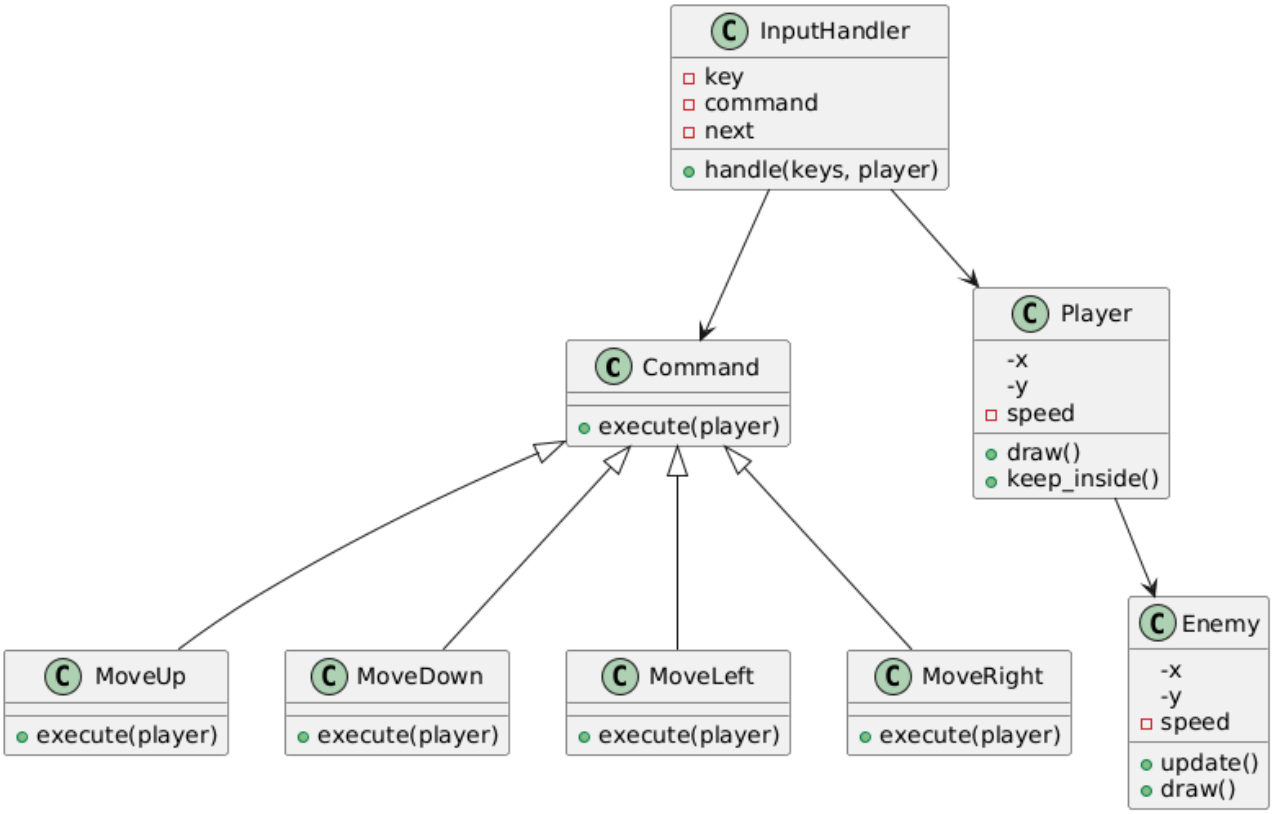
Al iniciar el programa, se abre una ventana de color negro con el cubo del jugador ubicado en el centro, entonces el jugador puede moverse usando las teclas **W, A, S, D**. Cuando realice la acción de alguna de las letras las bolas comienzan a caer desde la parte superior. El cubo no puede salirse de los límites de la pantalla y si una bola toca al jugador, el juego se detiene y se

muestra el mensaje de fin.

Los patrones **Command** y **Chain of Responsibility** garantizan un flujo de control ordenado y flexible durante toda la ejecución.

5 Diagrama Uml

El diagrama UML muestra las clases principales del sistema y sus relaciones con los patrones de diseño aplicados y explicados anteriormente.



6 Conclusión

El proyecto permitió comprender la aplicación práctica de los patrones de diseño **Command** y **Chain of Responsibility** dentro de un contexto lúdico. Con su uso, el código del juego se mantuvo limpio, modular y fácil de ampliar. Asi este enfoque demuestra nuevamente que los patrones de diseño ofrecen ventajas significativas en la organización, mantenibilidad y escalabilidad del software.