FACULTAD DE INGENIERIA
Universidad de Buenos Aires

Trabajo Práctico 2 — Java

[7507/9502] Algoritmos y Programación III
Curso 2
Segundo cuatrimestre de 2020

Alumno	Padron	Mail
Michael Gustavo Mena Giraldo	102685	mgmena@fi.uba.ar
Juan Ignacio Karabin	104697	jkarabin@fi.uba.ar
Juan Manuel Prato	105224	jprato@fi.uba.ar

Índice

1. Introducción

El presente informe reune la documentación de la solución del segundo trabajo práctico de la materia Algoritmos y Programación III que consiste en desarrollar una aplicacion en Java que permita lo siguiente:

- Crear un personaje que pueda levantar o bajar un lápiz.
- Mover al personaje en ciertas direcciones utilizando bloques.
- Crear un sector dibujo.
- Mover al personaje con su lápiz y realizar un dibujo cuando el mismo se encuentra bajo.
- Crear algoritmos utilizando bloques de repetición.
- Crear algoritmos utilizando bloques de invertir.

Esto será llevado a cabo utilizando Java como lenguaje de programación (tanto para la lógica como la interfaz de usuario), y se utilizarán metodologías ágiles, TDD, e integración contínua con un sistema de control de versiones (Git y GitHub).

2. Supuestos

Los supuestos refieren a funcionalidades no planteadas en el enunciado, sobre las que el equipo de desarrollo tomó una postura y consideró adecuado incluirlas en el proyecto:

- El personaje no puede moverse en diagonal.
- Por defecto, el personaje comienza con el lápiz levantado.
- El tablero es infinito.
- Mientras no se reinicie el juego, o se limpie el tablero, el último dibujo realizado seguirá estando en la pantalla, y la última posición donde quedó el personaje será utilizada como la actual.

3. Modelo de dominio

Para comenzar con nuestro modelo de dominio, daremos una lista de las clases mas relevantes que fueron creadas para luego dar mas detalles acerca de estas:

- Contenedor de Bloques
- Sector Dibujo
- Sector Algoritmo
- Lápiz
- IBloque

ContenedorDeBloques Se implementó esta clase abstracta, ya que varias otras clases necesitan la capacidad de almacenar bloques, e interactuar con estos. Por ejemplo, el sector algoritmo, o los bloques repetir e invertir.

SectorDibujo El sector dibujo es es el encargado de contener tanto el tablero como el personaje. Con esta clase se utilizó el patron Singleton, ya que esta es una clase en la cual solo se una instancia de la misma.

SectorAlgoritmo En esta clase también se implementa el patrón Singleton, ya que solo se requiere una instancia de la misma. Además de esto, esta clase hereda de ContenedorDeBloques, ya que en esta clase se almacenan los bloques del algoritmo a ejecutar.

Lápiz La instancia de esta clase, es decir, el lápiz, puede comportarse de dos formas distintas. Si se encuentra levantado (con su punta hacia arriba), no dibuja ninguna línea. Caso contrario, cuando su punta se encuentra boca abajo, dibuja las respectivas líneas.

IBloque Se crea esta interfaz para que la implemente cada bloque, ya que todos los bloques deberían poder responder a los mensajes declarados en esta.

4. Diagramas de clase

Para una mejor comprensión del modelo propuesto, se dividieron los diagramas en varias partes.

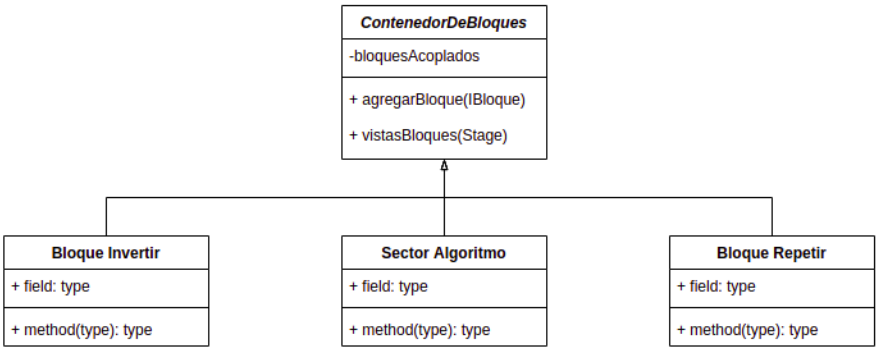

Figura 1: Diagrama de clases general de ContenedorBloques

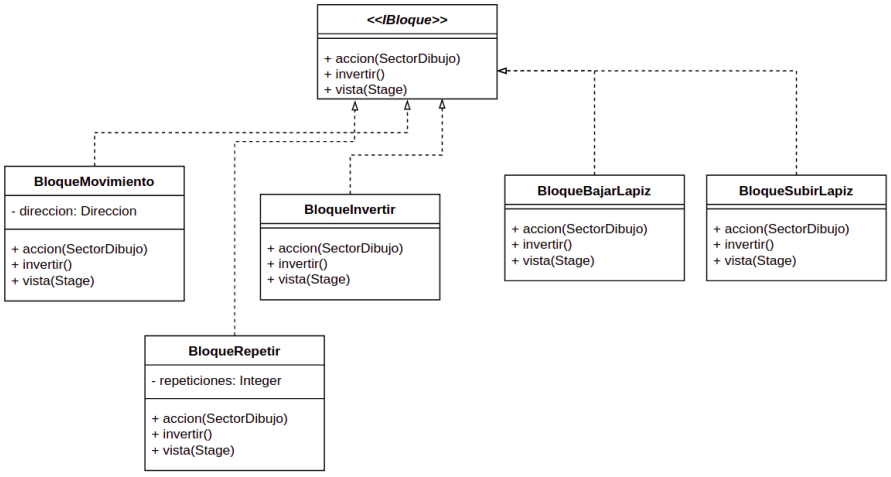

Figura 2: Diagrama de clases general de la Interfaz Bloque

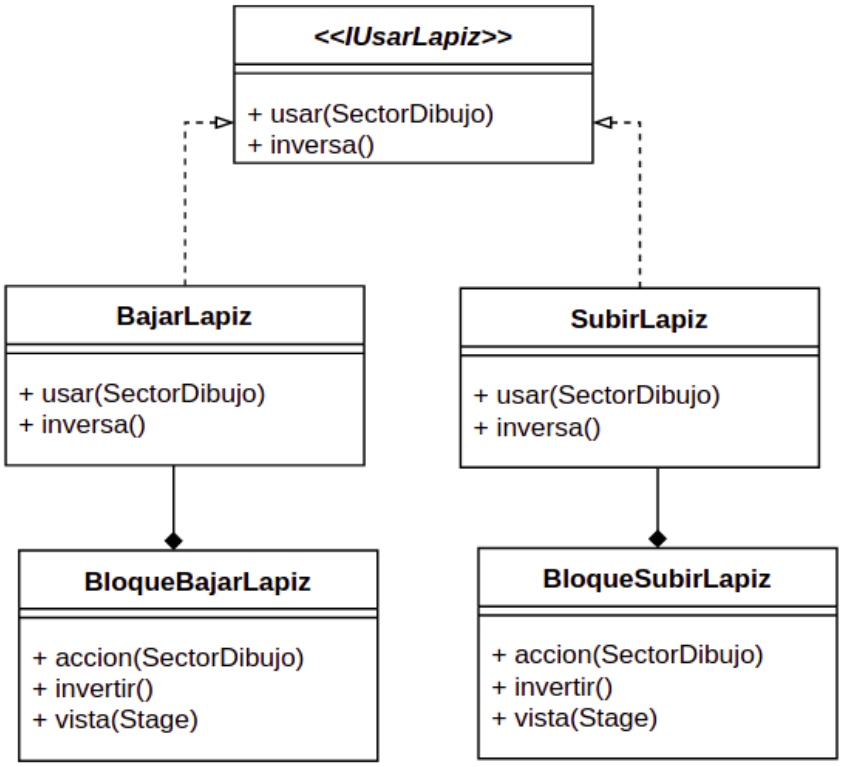

Figura 3: Diagrama de clases general de la Interfaz UsarLapiz

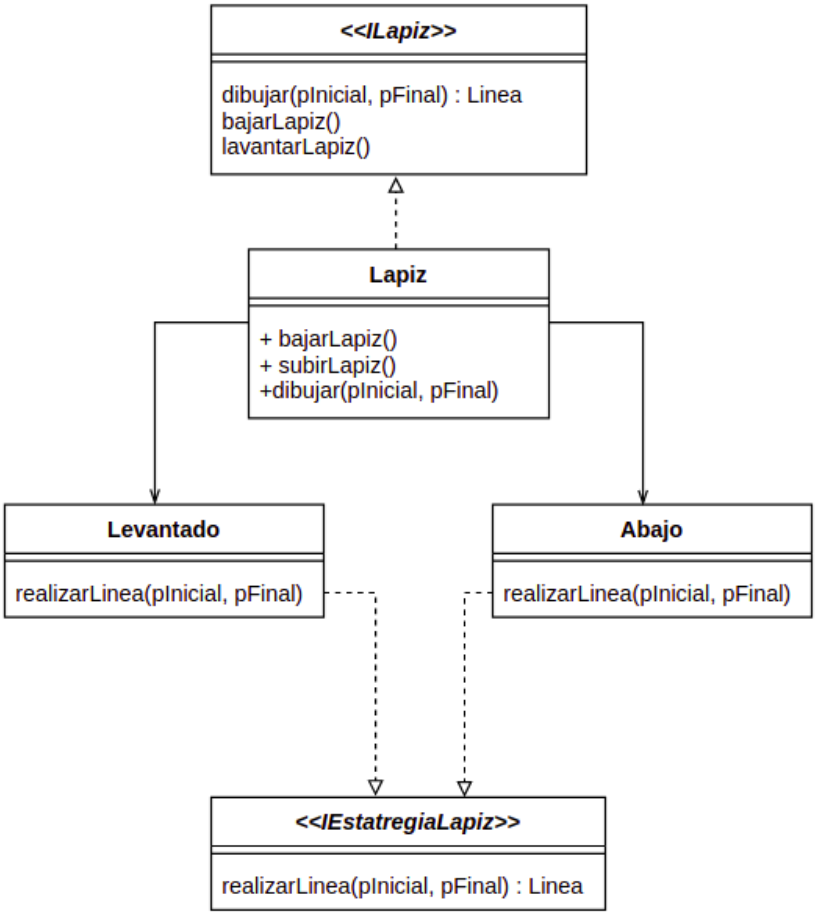

Figura 4: Diagrama de clases general de Lapiz

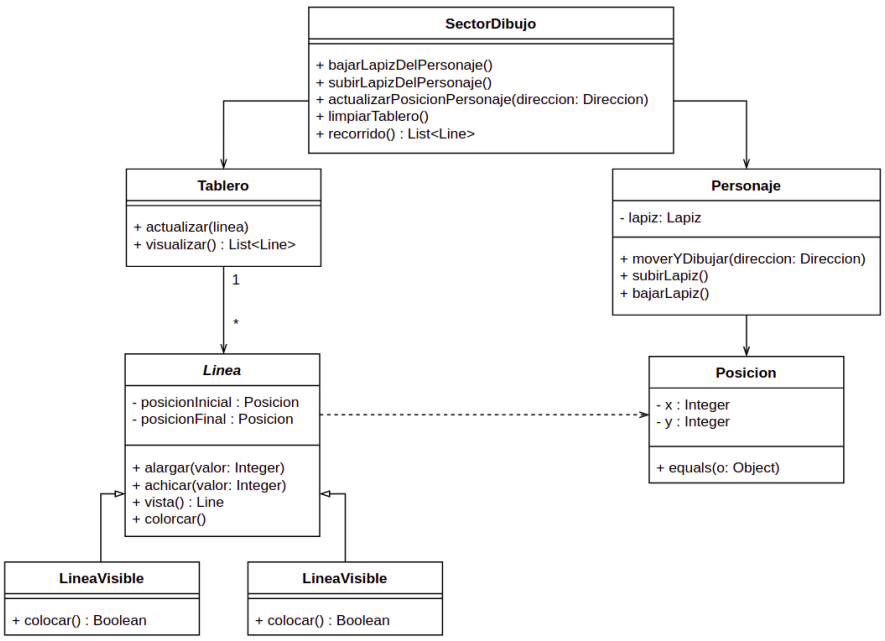

Figura 5: Diagrama de clases general de SectorDibujo, Tablero y Personaje

5. Detalles de implementación

5.1. Clase Lápiz

En el caso de esta clase, se utilizó el patron strategy, ya que el lápiz debía poder comportase como un lápiz levantado, y en tal caso no debe dibujar una linea, y como uno apoyado, en ese caso, debiendo dibujar la línea. Para esto, el lápiz puede usar la estrategia del lápiz lavantando o apayodo. Este cambio de estrategia es realizado externamente, segun los bloques que se utilizen.

5.2. Aclaraciones

- Se utilizó setStage ya que fue la forma que encontramos para refrescar la escena por esto lo considemos un violación al tell dont ask aceptable.
- Se implementó el método vista en los bloques, a pesar de ser parte de la interfaz gráfica, ya que está se define según el estado interno del bloque. Por este motivo, pensamos que era mejor que sea parte del mismo.

6. Diagramas de secuencia

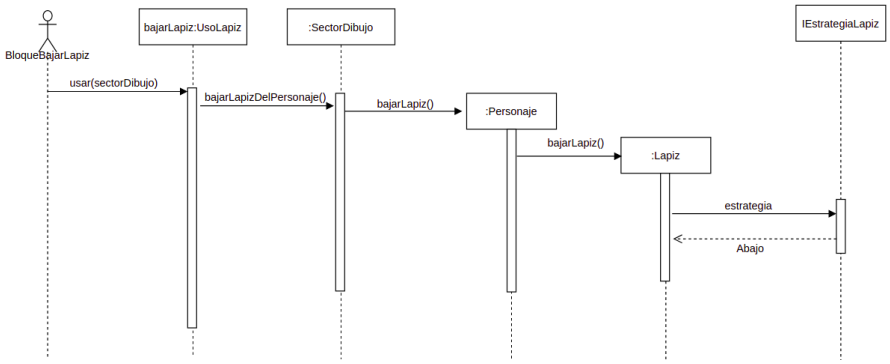

Figura 6: Diagrama de secuencia en el caso en el que lo primero que se hace al iniciar el juego es bajar el lápiz del personaje.

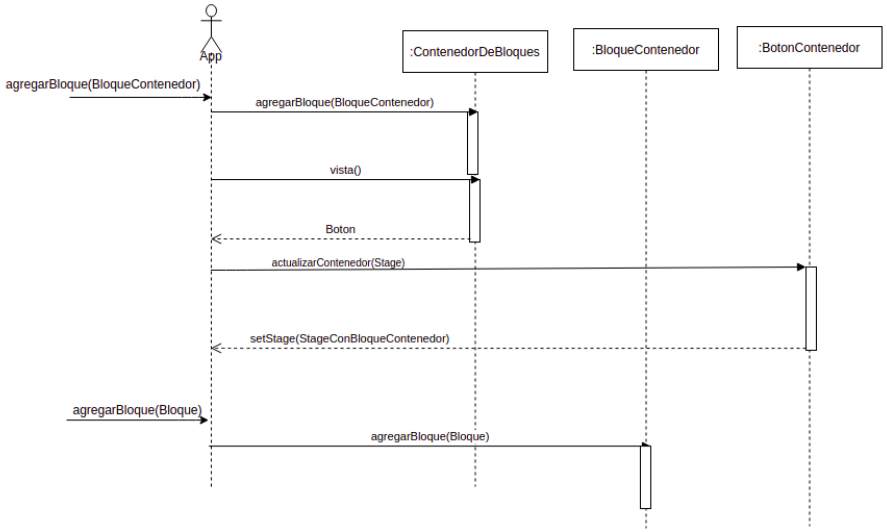

Figura 7: Diagrama de secuencia en el caso en el que se agrega y utiliza un bloque contenedor.

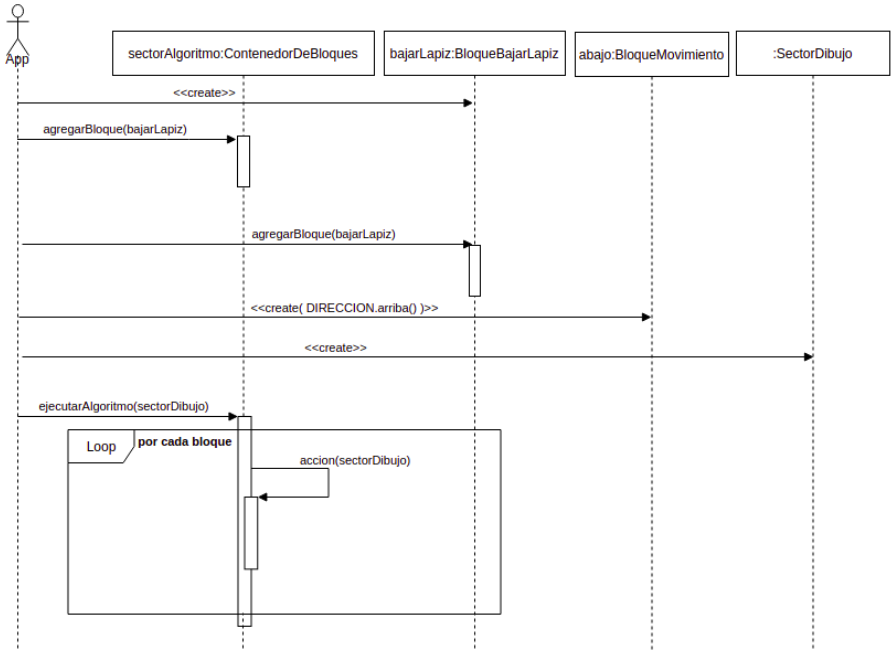

Figura 8: Diagrama de secuencia en el caso en el que el personaje se mueve hacia arriba, se baja el lapiz y se agregan bloques.

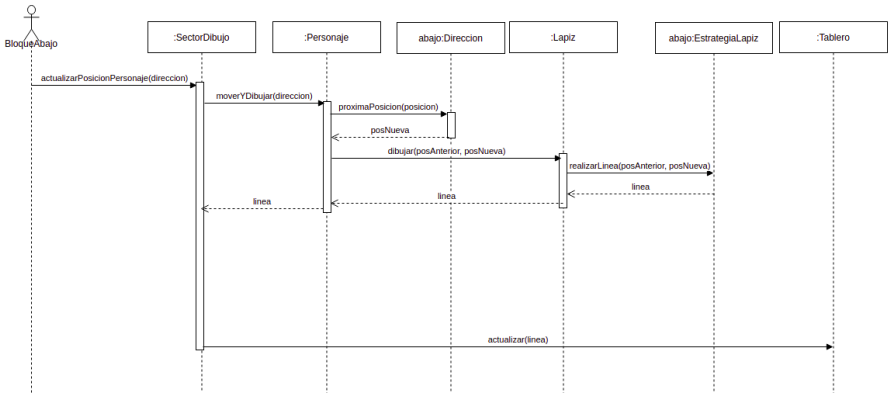

Figura 9: Diagrama de secuencia en el caso en el que al personaje se le actualiza su posición y dibuja.

7. Diagrama de estados

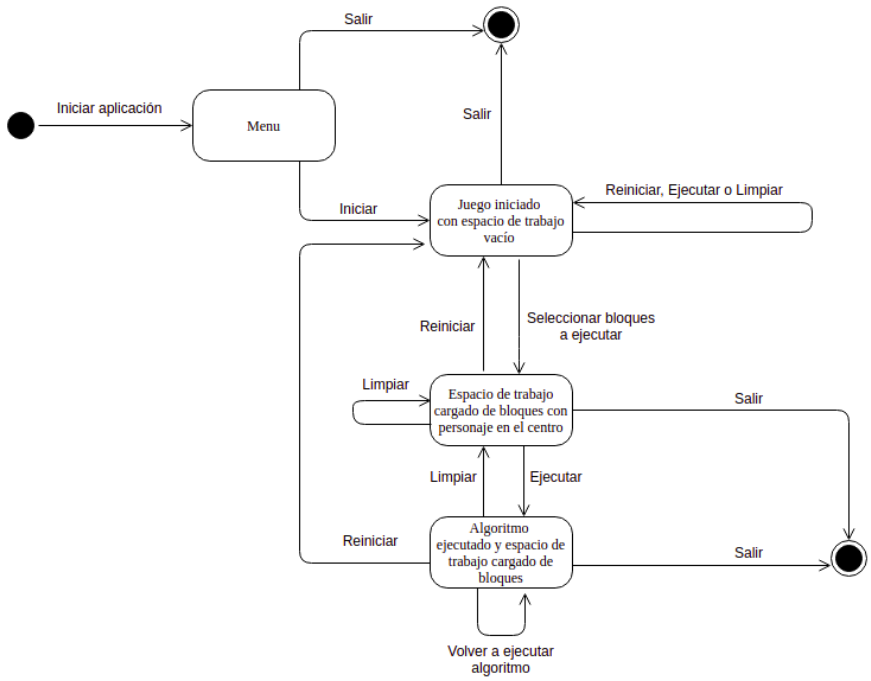

Figura 10: Diagrama de estados

8. Diagrama de paquetes

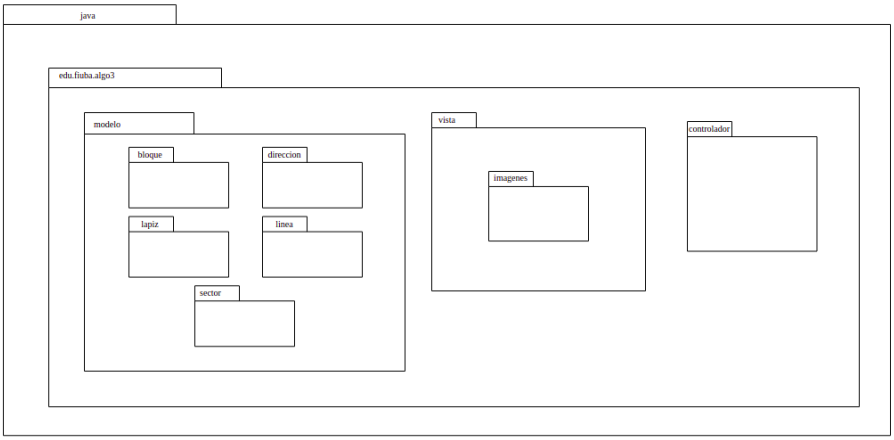

Figura 11: Diagrama de paquetes