Trabajo Práctico 2 — Java

[7507/9502] Algoritmos y Programación III
Curso 2 - Grupo 5
Segundo Cuatrimestre de 2020

Alumno:	Filipovskis	Nieva	Aceval	Saavedra
Número de padrón:	105231		104082	104302

Índice

1. Introducción

Este informe contiene la documentacion de la solución del segundo trabajo práctico de la materia Algoritmos y Programación III que consiste en desarrollar un juego que permite aprender los conceptos básicos de programación, armando algoritmos utilizando bloques visuales. Usando todos los conceptos vistos en el curso, utilizando un lenguaje de tipado estático (Java) con un diseño del modelo orientado a objetos y trabajando con las técnicas de TDD e Integración Continua vistos hasta ahora en el curso.

2. Supuestos

El dibujo no tiene limites: El dibujo y los movimientos asociados se harán sin limites en el area de dibujo, aún cuando se deje de ver en el espacio designado a esto en la interfaz.

Reseteo tablero y bloque : los bloques correspondientes a estas dos acciones no se toman en el modelo realizado, ya que en las especificaciones del trabajo no se mencionan en ninguna circunstancia.

3. Diagramas de clases

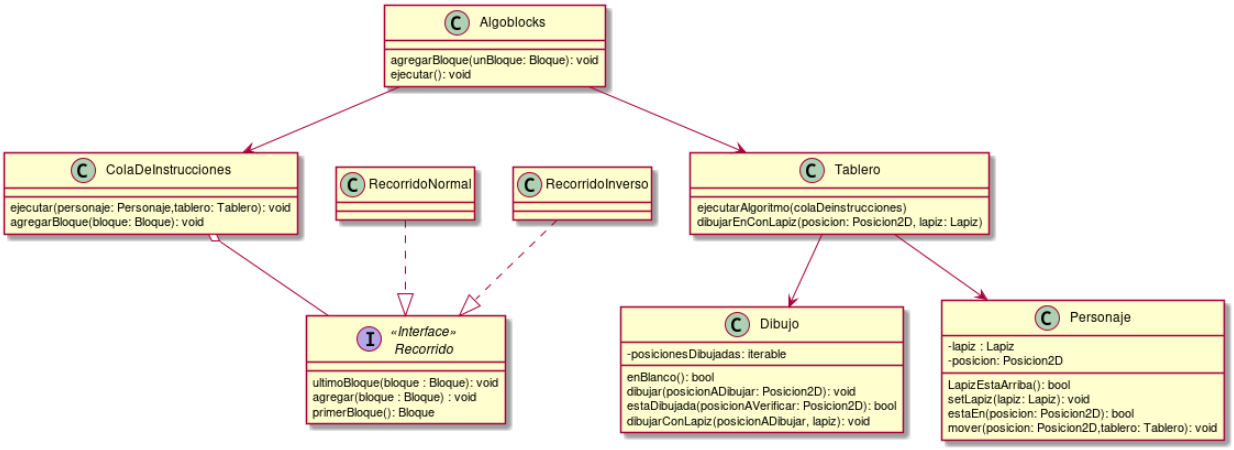


Figura 1: Diagrama de clases de relación entre AlgoBlocks - Tablero - Dibujo - Personaje - ColaDeInstrucciones - Recorrido

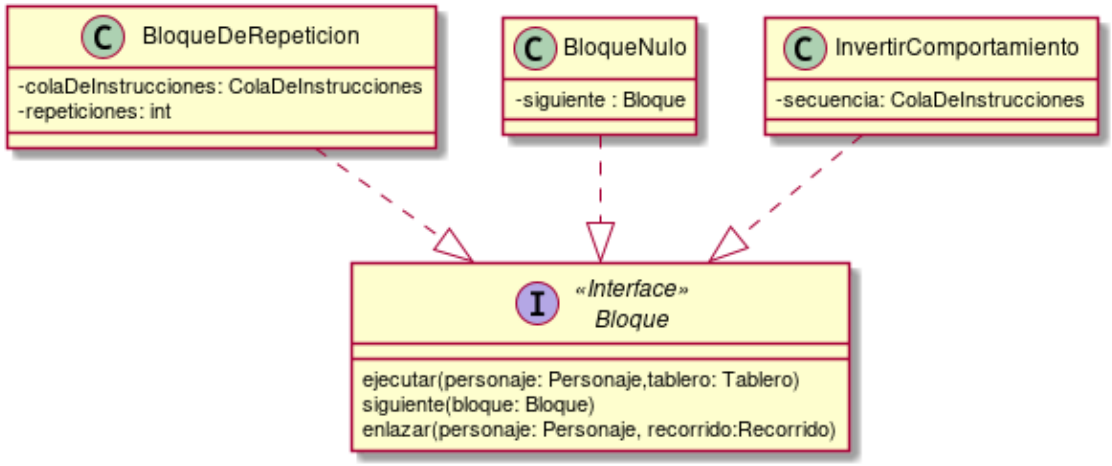


Figura 2: Diagrama de clases de relación entre la interfaz Bloque y algunos de sus bloques que brindan comportamiento

1

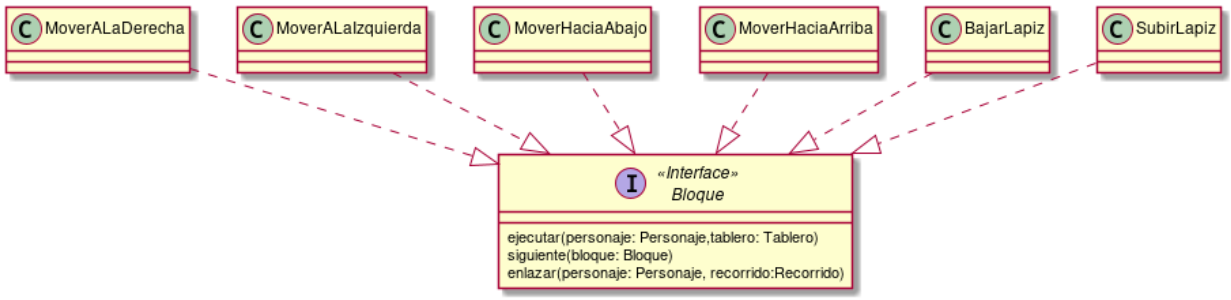
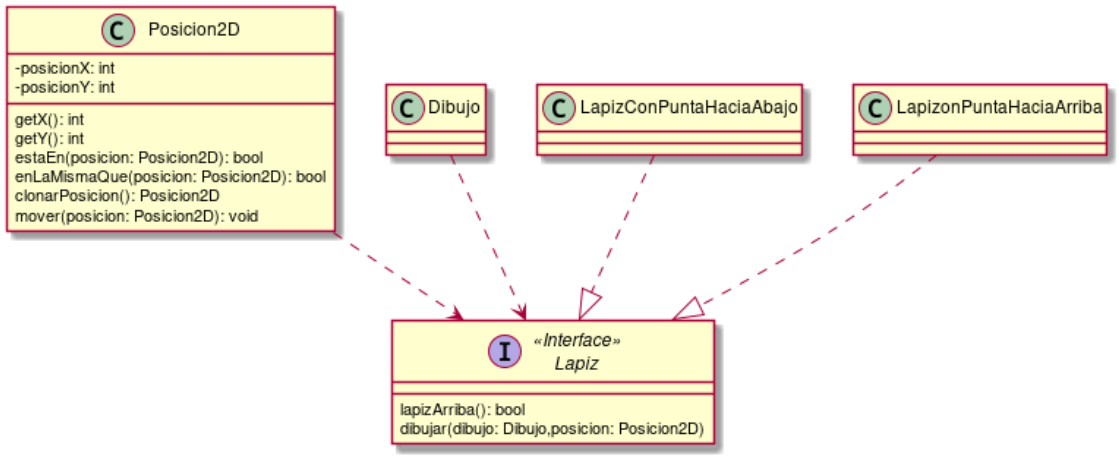


Figura 3: relación entre interfaz Bloque y el resto de las clases que lo implementan



Se muesta la relacion entre Lapiz y las clases que lo implementan y las dependencias con las clases Dibujo - Posicion2D

Las clases que implementan la interfaz bloque en este último diagrama no afectan a la posición de un personaje, como sí lo hacen los planteados en la Figura 2. En este caso, las clases se encargarán de instanciar un tipo de lápiz en base a la instrucción que se reciba.

4. Diagramas de secuencia

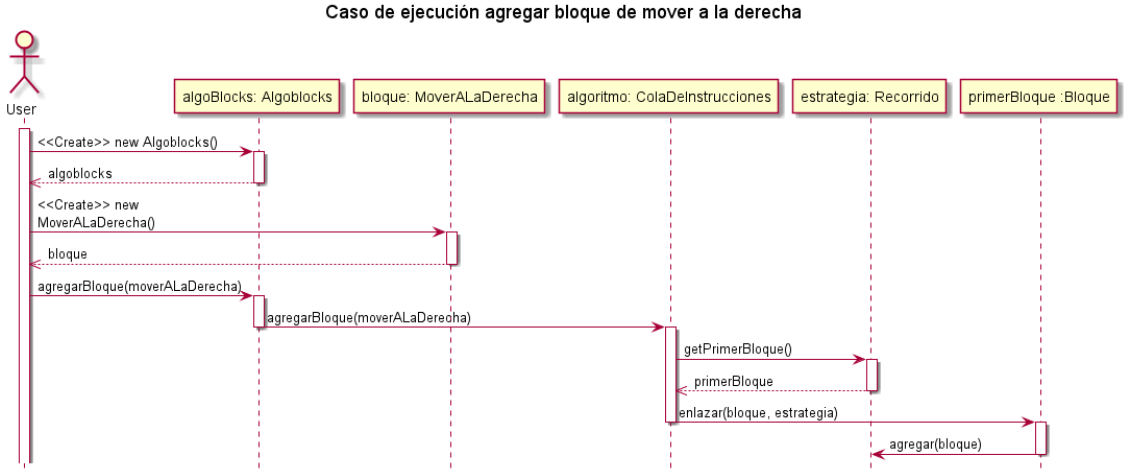


Figura 4: Agregar bloque mover a la derecha.

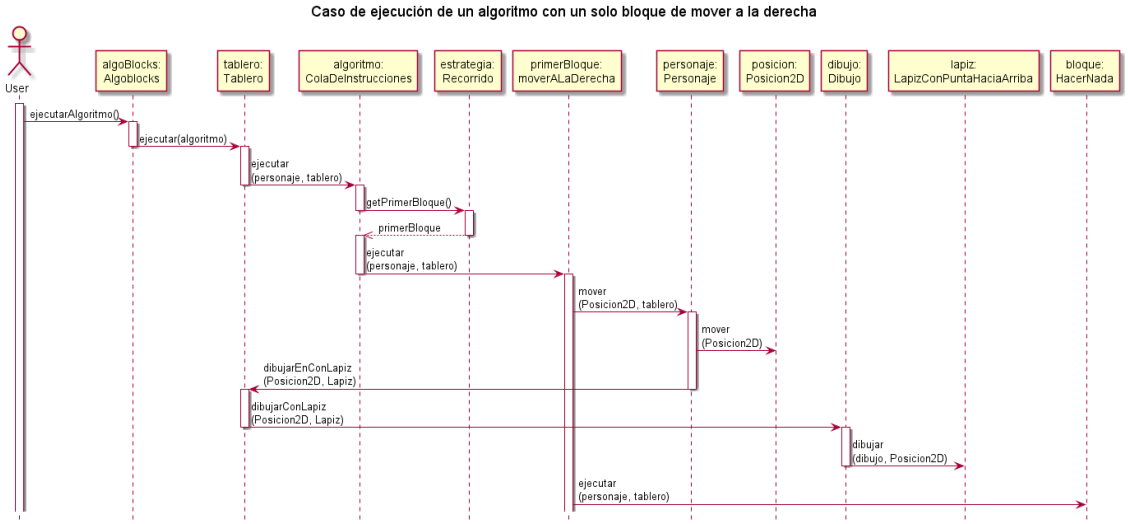


Figura 5: Ejecutar un solo bloque mover a la derecha.

5. Diagramas de paquetes

6. Diagramas de estado

7. Detalles de implementación

7.1. Estrategias utilizadas para puntos conflictivos

Manejo de posición del personaje El objetivo principal de plantear la Posicion de un Personaje de la manera realizada, es obtener posiciones genéricas y modificables a futuro. En principio se tiene a un Personaje que tiene una posicion en dos dimensiones representada con la clase llamada Posicion2D. Las posiciones de dicha clase se ven afectadas únicamente por objetos de la clase Bloque. Estos se encargarán de instanciar posiciones que representen movimientos, y que al ser enviadas a la posición del personaje mediante el método mover() permitan que se determine la nueva posición.

De la clase moverALaDerecha, que implementa Bloque:

```
public void ejecutar(Personaje personaje) {
    personaje.mover(new Posicion2D(1, 0));
}
```

De la clase Personaje:

```
public void mover(Posicion2D posicion) {
    this.posicion.mover(posicion);
}
```

De la clase Posicion2D:

```
public void mover(Posicion2D posicion) {
    this.x += posicion.getX();
    this.y += posicion.getY();
}
```

7.2. Patrones de diseño utilizados

Facade Descripción 1.

8. Excepciones