FACULTAD DE INGENIERIA
Universidad de Buenos Aires

[75.07 / 95.02]
Algoritmos y programación III

Trabajo práctico 2

Integrantes:

NORESE, Florencia - 83.920
STRNISKO, Juan pablo - 93.242
STROIA, Lautaro - 100.901
SUAREZ, Martín - 101.540

Introducción

El presente informe reúne la documentación de la solución del segundo trabajo práctico de la materia Algoritmos y Programación III que consiste en desarrollar una aplicación de manera grupal aplicando todos los conceptos vistos en el curso utilizando un lenguaje de tipado estático, Java.

Sistema de turnos

La clase SistemaTurnos se encarga de recibir a los jugadores limitando el tope impuesto por el modelo (en este caso 2 jugadores). Posee la clase ListaJugadores la cual se encarga de almacenar los jugadores y poder ir iterándolos. Además, la lista comienza a iterar en una posición aleatoria por lo que permite que el jugador que comienza la partida sea elegido de esta manera. A futuro, será el sistema de turnos el encargado de manejar los turnos del juego y permitir, o no, que cada jugador actúe.

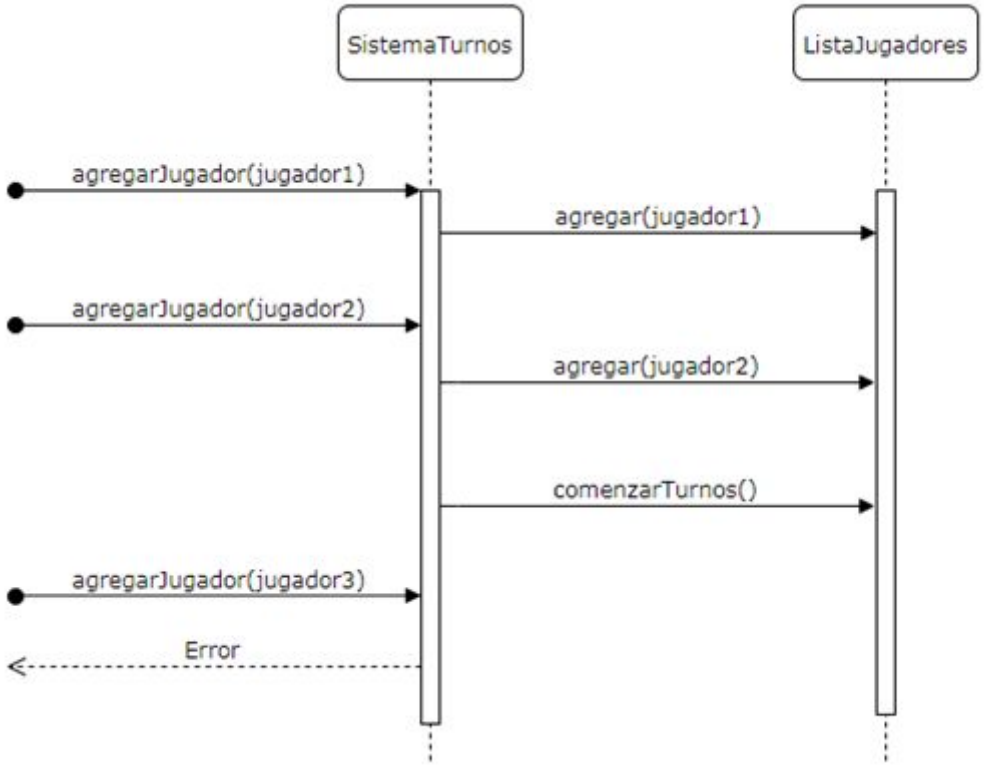

Diagrama de secuencia en el que se intentan agregar tres jugadores al sistema de turnos.

Clase Edificios

Los edificios poseen dentro una clase vida para evitar cargar de responsabilidades la clase base. A futuro se debe refactorizar aún más y delegar en otras clases las funciones del edificio. Por ejemplo, para el castillo, agregarle una clase Atacante que se encargue de los ataques y una clase ProductorDeUnidades, que se encargue de producir las unidades.

Clase Mapa

Modelo de dominio

El desarrollo de nuestra solución se basa en la creación de una objeto mapa, el cual se ocupa de crear, con las dimensiones pasadas por parámetro, un objeto zonaDeJuego. El objeto zonaDeJuego se ocupa de crear una matriz de objetos tipo Celda. Los objetos tipo Celda son los encargados de posicionar las unidades y edificios e identificar si estan ocupados o no.

Implementación

La clase mapa es la encargada de crear el objeto zonaDeJuego de acuerdo a los valores pasados por parámetro. La creación de la zonaDeJuego es su única responsabilidad. Encontramos en esta clase los metodos colocarUnidad y colocarEdificio, pero éstos no son responsabilidad de Mapa. Mapa delega esta responsabilidad a zonaDeJuego, y ésta última delega esta responsabilidad a la clase Celda. Las Celdas son las encargadas de posicionar unidades/edificios y saber si están ocupadas o no. En el caso de zonaDejuego es responsable, de acuerdo a sus dimensiones (atributos de la clase), de saber si una celda pertenece o no al espacio de juego.

Clases de Turnos

Modelo de dominio

Para modelar los distintos estados que tienen las unidades y edificios en sus acciones hicimos clases de Turno...Habilitado y Turno....Finalizado (Ejemplo: TurnoMovimientoHabilitado).

Implementación

Utilizamos una interfaz TurnoMovimiento o Turno Recolectar de las cuales heredan sus respectivas clases hijas para Habilitar o Finalizar la acción. La unidad o el edificio al realizar una acción que solo puede hacer una vez por turno (léase recoger oro, moverse, etc) cambia su estado al TurnoFinalizado y hace lo mismo con sus otras acciones que no sean realizables a partir de ahí ese mismo turno. Como puede ser construir y recolectar oro.

Clase Unidad

Modelo de dominio

Esta clase, al igual que Edificio, implementa la interfaz Atacable: cada Unidad y Edificio es considerado un Atacable ya que tiene la habilidad de atacar a otros Atacables. Dentro de esta clase, se encuentran modeladas las siguientes unidades: Aldeano, Espadachín, Arquero y ArmaDeAsedio

Implementación

Nuestra idea fue que la clase Unidad, cómo cada unidad puede atacar a otra unidad o a un objeto de la clase Edificio, implemente una interfaz llamada Atacable: una unidad puede atacar a otra unidad o a un edificio. Para evitar el uso de condicionales para saber a qué tipo de objeto está atacando cada unidad o edificio, se nos ocurrió utilizar el patrón Double Dispatch en la interfaz para así solo decirle a cada unidad/edificio que ataque a un atacable, y cada uno sabrá a qué tipo de objeto atacar (ya que el daño generado a una unidad es distinto al generado en un edificio).