FIUBA - 75.07

Algoritmos y programación III

Trabajo práctico 2: AlgoPoly

2do cuatrimestre, 2017

(trabajo grupal de 4 integrantes)

Alumnos:

Nombre	Padrón	Mail
ALVAREZ WINDEY, Ariel	97893	arieljaw12@gmail.com
OCAMPO, Damián	88055	okalandia@gmail.com
ANCA, Jorge	82399	ancajorgel@yahoo.com.ar
DIZ, Gonzalo	98546	dizmartin.gonzalo@gmail.com

Fecha de entrega final: 30 de Noviembre de 2017

Tutores: Diego Sánchez y Uriel Kelman

Comentarios:

Supuestos

Supuesto sobre avance dinámico y retroceso dinámico:

Dado que un jugador no puede retroceder dinámicamente desde retroceso dinámico hasta avance dinámico pero sí puede avanzar dinámicamente y llegar a retroceso dinámico, se supondrá que llegado el caso que esto ocurra el casillero Retroceso Dinámico afectará normalmente al jugador dependiendo de los últimos dados obtenidos.

Esto era necesario contemplarlo ya que sino podría haber entrado en un bucle infinito y habría que suponer en ese caso que llegando a uno de estos casilleros viniendo del contrario no debía afectar al jugador.

Supuesto sobre intercambios:

Se consideró que para los intercambios sólo pueden cambiarse terrenos con terrenos o compañías con compañías, no se admiten cambios mezclados. Es decir, por ejemplo, cambiar Buenos Aires Sur por Subtes.

Modelo de dominio

Las entidades principales que se crearon fueron:

- Jugador
- Casillero
- Tablero
- ArmadorDeTablero
-

De la clase Casillero heredan los distintos tipos de casilleros que tiene el juego. Estos son los Terrenos (simples y dobles), las Compañías, los casilleros denominados Especiales, como lo son el Quini6, Policía, Cárcel, etc. Todos tienen redefinido el método afectarJugador() que, como su nombre lo indica, afecta al jugador de manera distinta dependiendo qué casillero es y en qué condiciones esté (por ejemplo, un terreno con propietario no afecta a un jugador de la misma manera que si no tuviese propietario). Luego se crearon entidades que trabajan junto con un tipo de casilleros puntualmente. Estas son:

- AlquilerDeTerrenoDoble
- AlquilerDeTerrenoSimple
- Hotel
- Casa

- BonificadorDeCompania
- PorcentajeDeImpuesto

Los primeros dos, son entidades encargadas de devolver el costo del alquiler correspondiente a un terreno, según la cantidad de casas u hoteles que posea el mismo. Las entidades Hotel y Casa, modelan en principio la existencia de las distintas edificaciones posibles en los terrenos, pero no tienen ningún comportamiento o responsabilidad particular.

En cuanto al bonificador de compañía se encarga de devolver el factor de bonificación que multiplica a la última tirada de dados del jugador dependiendo si el propietario de la compañía tiene o no la compañía pareja correspondiente.

El porcentaje del impuesto, es la entidad que trabaja junto con el casillero ImpuestoDeLujo para el cálculo y la extracción del impuesto al jugador que cae en él.

Por último se crearon dos interfaces de restricciones:

- RestriccionDeConstrucción
- RestricciónDeIntercambio

La primera valida las condiciones en que se quiere construir una casa o un hotel en determinado terreno (por ejemplo, que sea el propietario quien quiere construir, que no se haya alcanzado el límite de edificaciones permitidas, etc.).

La segunda valida las condiciones en que se quiere realizar un intercambio de terrenos o de compañías entre jugadores. Por ejemplo, que el intercambio sea entre dos jugadores distintos o que las propiedades en cuestión corresponden a cada jugador que participa del intercambio.

Se utilizó el patrón Modelo, controlador, vista. En donde el controlador es el encargado de “mediar” entre el modelo y la vista, interpretando los eventos que se generan durante el juego.

Diagramas de clases

Ver Anexo1 (Diagramas.asta)

Diagramas de secuencia

Ver Anexo1 (diagramas.asta)

Diagrama de paquetes

Ver Anexo1 (diagramas.asta)

Diagramas de estado

Ver Anexo1 (diagramas.asta)

Detalles de implementación

En un principio, para la primera entrega, se había utilizado el patrón de diseño Singleton, para el Tablero y para los Casilleros, ya que era de utilidad poder acceder a las instancias desde cualquier clase y se consideró lógico que haya una sola instancia de estas entidades.

Luego se empezó a tener conflictos con este patrón ya que, por ejemplo, en el momento de la asociación a Terrenos Pareja o Compañías Pareja, no se podía hacer a través de un constructor ya que la creación de uno dependía de la creación de su pareja entonces no se podía terminar de crear ninguna de las dos ya que se entraba en un bucle infinito. Además se analizó la poca extensibilidad que impone este patrón para el caso de los casilleros, porque en un futuro si se querría tener un tablero con dos casilleros iguales no se podría dado que solo puede haber una instancia de cada uno.

Se decidió finalmente sacar el patrón de diseño en las entidades de Casillero y agregar una clase ArmadorDeTablero, que se encarga de crear la instancia del Tablero (única, para esta clase si se dejó el patrón antes mecionado), crear los casilleros, configurar los casilleros que tienen relación con otros y finalmente agregarlos al tablero.

Otro punto conflictivo que se tuvo en cuanto al diseño fue la toma de decisiones por parte del jugador. Se entendió finalmente que esto estaría contemplado en la vista, por lo que en cuanto al diseño del modelo solo era necesario crear los mensajes que ejecuten los comportamientos que se querían probar. Por ejemplo, cuando un jugador caía en un terreno, se tuvo dificultad en avanzar ya que no se sabía cómo ofrecerle al jugador si quiere

o no comprarlo. Pero esto estará manejado por los controladores y las vistas. Entonces cuando un jugador decide comprar el terreno se le enviará el mensaje comprar(unTerreno).

Por último, para el desplazamiento del jugador, se utilizó el patrón de diseño State, en donde se creó la clase EstadoJugador. Entonces el encargado del desplazamiento del jugador es el estado del mismo. Por ejemplo, para el estado Libre se desplazará de una manera (avanzará lo que indiquen los dados), mientras que para un estado Encarcelado se desplaza de otra manera (en este caso no se puede desplazar y lanza una excepción en caso de intentarse desplazar).

Excepciones

Las excepciones creadas son:

- AlquilerInexistenteError: Esta excepción la maneja las clases de alquileres y es lanzada cuando se le pasan argumentos inválidos para la obtencion de alquileres, por ejemplo: si se pide el alquiler para dos casas y dos hoteles. Ya que como máximo un terreno doble solo puede tener un hotel y cuando tiene un hotel no tiene casas.

- CapitalInsuficienteError: Esta excepción se lanza cuando se intenta de extraerle un dinero mayor al disponible a un jugador.

- ElTableroNoPoseeEseCasilleroError: Esta excepción la lanza la clase Tablero cuando se le pide un casillero que no posee.

- LimiteDeEdificacionesExcedidoError: Esta excepción es controlada por la interfaz de RestriccionDeConstruccion y es lanzada cuando se intenta construir más edificaciones de las permitidas en un terreno. Por ejemplo, construir una tercer casa en un terreno doble que ya posee dos casas.

- NoEstasEncarceladoException: Esta excepción la lanza el estado Libre cuando intenta pagar la fianza a la cárcel. Ya que un jugador que está libre no puede pagar la fianza.

- NoPuedeCederUnaCompaniaNoPropiaError: Esta excepción la maneja la interfaz de RestriccionDeIntercambio y es lanzada cuando un jugador quiere hacer un intercambio con una compañía de la que no es dueño.

- NoPuedeCederUnTerrenoASiMismoError: Esta excepción la maneja la interfaz de RestriccionDeIntercambio y se lanza cuando un jugador intenta realizar un intercambio y como parámetro de jugador de intercambio se recibe a el mismo.

- NoPuedeCederUnTerrenoNoPropioError: Es el mismo caso que NoPuedeCederUnaCompaniaNoPropiaError pero para el caso de los terrenos.

- NoPuedePagarFianzaException: Esta excepción la maneja el estado EncarceladoTurno1 ya que un jugador que pasó un solo turno en la cárcel no puede pagar la fianza.

- NoSePuedeConstruirUnHotelEnUnTerrenoSinConstruirPrimeroTodasLasCasasPosiblesEnElTerrenoParejaError: Esta excepción es controlada por la interfaz de RestriccionDeConstruccion y se lanza cuando no se construyó el límite de casas permitido en el terreno pareja correspondiente.

- NoSePuedeConstruirUnHotelEnUnTerrenoSinConstruirPrimeroTodasLasCasasPosiblesError: Esta excepción es controlada por la interfaz de RestriccionDeConstruccion y se lanza cuando no se construyó el límite de casas permitido en el terreno en el que se quiere construir el hotel.

- NoSePuedeDesplazarJugadorEncarceladoException: Esta excepción es lanzada cuando un jugador encarcelado intenta desplazarse.

- PorcentajeInvalidoError: Esta excepción se lanza cuando se intenta crear una instancia de PorcentajeDeImpuesto con un porcentaje inválido.

- SeNecesitanAmbosTerrenosParaEdificarEnUnTerrenoDobleError: Esta excepción es controlada por la interfaz de RestriccionDeConstruccion y se lanza cuando no se dispone del terreno pareja y se está intentando construir.

- SoloElPropietarioPuedeEdificarEnElTerrenoError: Esta excepción es controlada por la interfaz de RestriccionDeConstruccion y se lanza cuando se intenta construir en un terreno no propio.