Trabajo Voluntario

Programación

Juego MegaPoly

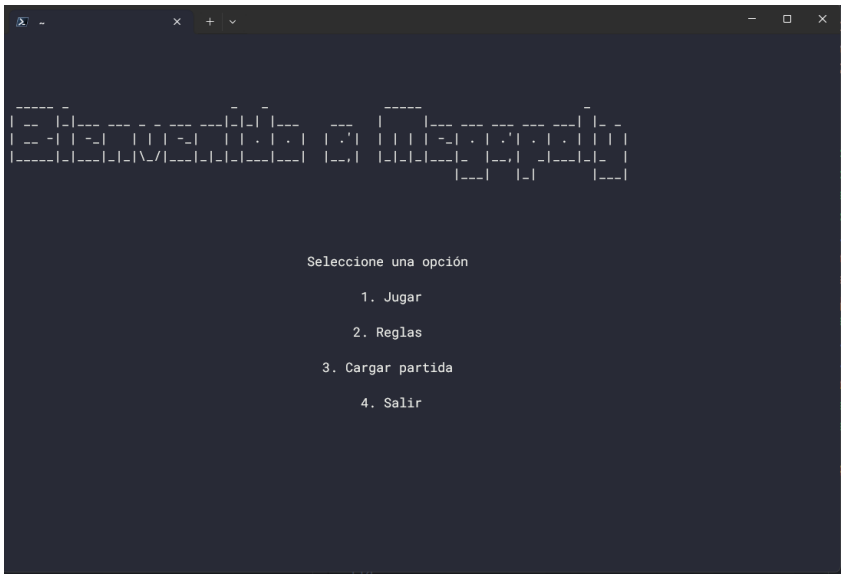
(versión de consola)

Trabajo realizado por David Morgade Gil – 1º DAM – MEDAC – PACIFICO

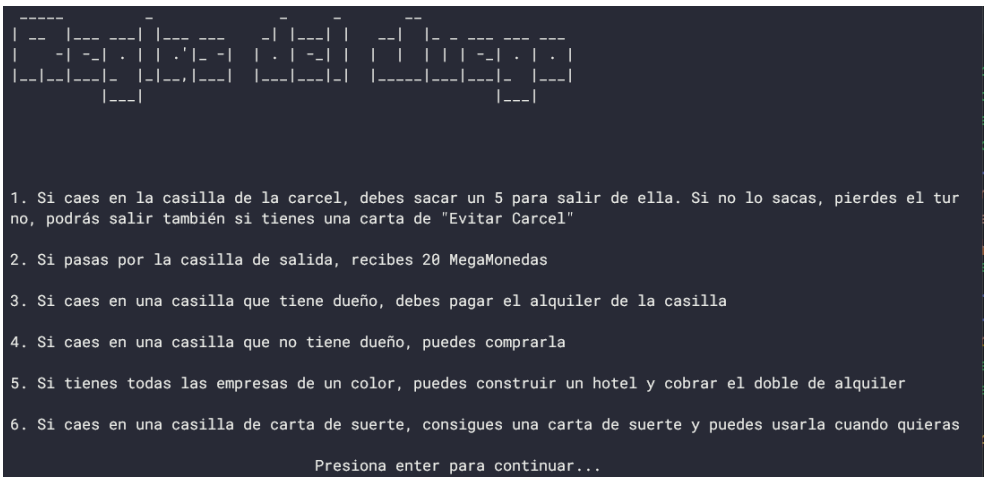
INTRODUCCIÓN

Esta presentación tratará sobre la primera versión de MegaPoly desarrollada por mí, la cual funciona completamente por consola.

Al entrar en el juego tendremos una pantalla de introducción que nos mostrará un mensaje de bienvenida junto a las opciones de Jugar, Reglas, Cargar partida y salir, en este menú podremos navegar introduciendo los números por teclado:



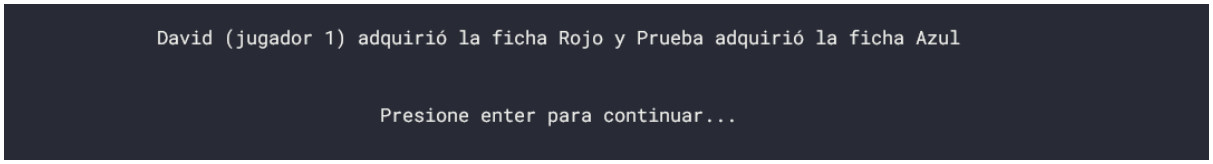
Si pulsamos la opción “Reglas” se mostrarán las reglas del juego MegaPoly:



Si nos vamos a la pantalla "Cargar Partida", nos aparecerán las partidas guardadas con su nombre y la fecha de la partida:

Si vamos a "Salir", la aplicación parará y nos dará un mensaje de despedida:

Finalmente si vamos a "Jugar" nos pedirán el nombre de los jugadores y el color de ficha que queremos seleccionar (azul o roja), una vez seleccionados nos mostrará por pantalla lo seleccionado:



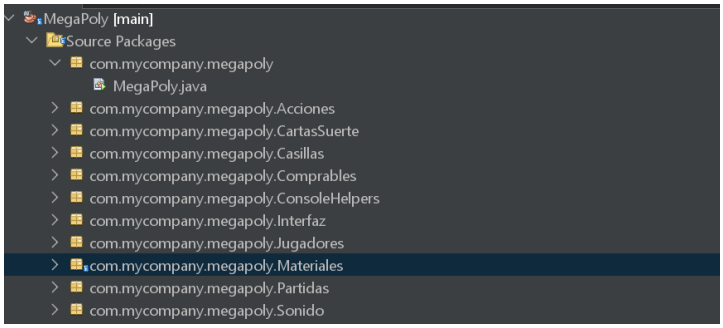
Una vez presionemos continuar comenzaremos el juego, se nos mostrará el tablero por pantalla y podremos tirar el dado, ver las cartas que tenemos, guardar la partida o salir, cada vez que caigamos en una casilla recibiremos un feedback:

Para ver el resto de menús ver el video del funcionamiento de la aplicación.

CONTENIDO

Estructura principal del proyecto

Para tener el código bien organizado y estructurado, he decidido organizar el proyecto en diferentes paquetes de clases relacionadas, evitando así crear clases muy grandes y con muchos métodos, haciendo un código más legible y limpio:



La aplicación se ejecuta en el método main que se encuentra en la clase principal (MegaPoly.java), en este lo único que haremos será instanciar fichas, jugadores, tablero y pasarlos por parámetros a los constructores de los menús, que pedirán los datos y los mostrarán por pantalla, quedando la clase main bastante limpia:

```java
public class MegaPoly {

  public static void main(String[] args) {
    Ficha fichaUno = new Ficha();
    Ficha fichaDos = new Ficha();

    Jugador jugador1 = new Jugador(fichaUno);
    Jugador jugador2 = new Jugador(fichaDos);

    Tablero tablero = new Tablero(5, jugador1, jugador2);

    MenuInicio menuInicio = new MenuInicio(
      jugador1,
      jugador2,
      fichaUno,
      fichaDos,
      tablero
    );

    Tablero tableroActualizado = new Tablero(5, jugador1, jugador2);

    MenuJuego menuJuego = new MenuJuego(jugador1, jugador2, tableroActualizado);
  }
}
```

Clases principales a destacar

A continuación, listaré las que para mi son las clases principales dentro de este programa, no podré dar un vistazo a todas ni a las profundidades de cada clase ya que he creado bastantes clases para poder añadir todas las funcionalidades del juego en base a mi conocimiento actual en Java.

La clase jugador es la clase que se encarga de almacenar los datos de los jugadores en sus parámetros, consta de distintos métodos para mostrar el dinero, usar las cartas de suerte (he decidido que en mi MegaPoly las cartas se puedan almacenar en una ArrayList para poder usarlas posteriormente, todas con efectos positivos), además de los diferentes getters y setters de los parámetros de la clase, esta clase será serializable para poder guardar y cargar la partida:

```java
/*
 * Clase que se encarga de crear los jugadores
 * @see Ficha
 * @see CartaSuerte
 * @see Comprable
 *
 */
public class Jugador implements java.io.Serializable {

  private static final long serialVersionUID = 1L;

  private boolean carcel = false;

  private static final int w = 100;

  private String nombre;

  private boolean turno;

  private int megaMonedas;

  // lista de cartas de suerte de cada jugador
  private List<CartaSuerte> cartas = new ArrayList<CartaSuerte>();

  // lista de propiedades de cada jugador
  private List<Comprable> comprables = new ArrayList<Comprable>();

  // ficha de cada jugador
  private Ficha ficha;

  /*
   * Constructor de la clase
   * @param nombre Nombre del jugador
   * @param turno Turno del jugador
   * @param ficha Ficha del jugador
   */
  public Jugador(String nombre, boolean turno, Ficha ficha) {
    this.nombre = nombre;
    this.turno = turno;
    this.megaMonedas = 100;
    this.ficha = ficha;
    this.cartas = new ArrayList<CartaSuerte>();
    this.comprables = new ArrayList<Comprable>();
  }

  /*
   * Constructor de la clase
   * @param ficha Ficha del jugador
   * @see Ficha
   */
  public Jugador(Ficha ficha) {
    this.ficha = ficha;
    this.megaMonedas = 100;
  }
```

La clase ficha será la que se encargue de representar la ficha y sus movimientos, además de su posición, esta la usaremos accediendo desde la propia clase jugador, por ello la instanciamos en el propio jugador como parametro, también será serializable para poder guardar y cargar partida:

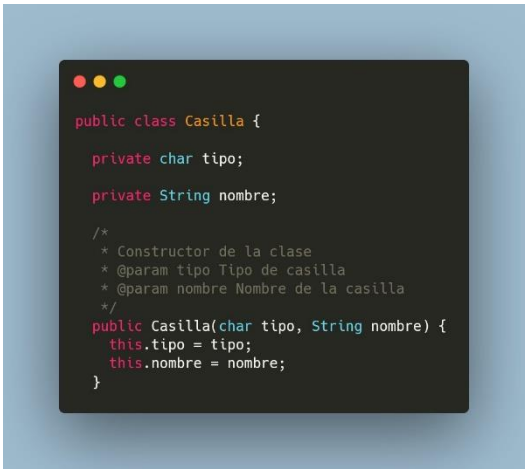
```java
/*
 * Clase que se encarga de crear las fichas
 * @see Jugador
 */
public class Ficha implements java.io.Serializable {

  private static final long serialVersionUID = 1L;

  private int posicion;

  private String colorFicha;

  private Jugador jugador;

  /*
   * Constructor de la clase
   * @param posicion Posición de la ficha
   */
  public Ficha() {
    this.posicion = 24;
    this.jugador = null;
  }
```

Siempre iniciaremos la ficha en la posición 24 (que es la posición de inicio en el array de casillas, además de los getters y setters, hago uso de un método avanzar para poder mover la ficha por el array.

Después tendremos el tablero, que estará formado por un array de casillas (clase creada por mi), el cual se encargará de mostrar el tablero cada vez que sea necesario en el menú, también contendrá las fichas de los jugadores:

```java
public class Tablero {

  private Casilla[] casillas;

  private Ficha fichaRoja;

  private Ficha fichaAzul;

  private int tamanioLado;

  /*
   * Constructor de la clase
   * @param tamanioLado Tamaño del tablero
   * @param fichaRoja Ficha del jugador rojo
   * @param fichaAzul Ficha del jugador azul
   */
  public Tablero(int tamanioLado, Jugador jugadorRojo, Jugador jugadorAzul) {
    this.tamanioLado = tamanioLado;
    this.casillas = new Casilla[tamanioLado * tamanioLado];
    this.fichaRoja = jugadorRojo.getFicha();
    this.fichaAzul = jugadorAzul.getFicha();
  }
```

La clase Casilla será la clase padre de todos los tipos de casillas creados (carcel, parking, propiedad, salida, suerte), estas compartirán todas un tipo y un nombre que heredarán, además de sus getters y setters:



```java
public class Casilla {

  private char tipo;

  private String nombre;

  /*
   * Constructor de la clase
   * @param tipo Tipo de casilla
   * @param nombre Nombre de la casilla
   */
  public Casilla(char tipo, String nombre) {
    this.tipo = tipo;
    this.nombre = nombre;
  }
```

Como ejemplo podemos ver la casilla recompensa, que heredará de casilla y será la que guarde la recompensa de 20 MegaMonedas para el jugador que pase por esta casilla:

```java
public class CasillaSalida extends Casilla {

  private int recompensa;

  /*
   * Constructor de la clase
   * @param tipo Tipo de casilla
   * @param nombre Nombre de la casilla
   * @param recompensa Recompensa que se va a dar al jugador
   */
  public CasillaSalida(char tipo, String nombre) {
    super(tipo, nombre);
    this.recompensa = 20;
  }

  public int getRecompensa() {
    return this.recompensa;
  }
}
```

Me hubiera gustado explicar en detalle la gran mayoría de clases de este proyecto, como las encargadas de los menús, de guardar la partida, de asignar propiedades o de manejar el sonido del juego, pero no puedo extenderme más allá de lo que índica la guía del documento para la entrega del trabajo.

CONCLUSIÓN

Como he comentado en la introducción este trabajo, tengo pendiente de desarrollar la aplicación con interfaz, lo he dejado a la espera a dar la clase correspondiente sobre el desarrollo de interfaces gráficas en Java.

Como extras a los requisitos que se piden para este proyecto he añadido la música para el menú inicial, además de ello cada vez que se cae en una casilla se reproducirá un sonido (sacados de la versión de Monopoly de la NES), que cambiará dependiendo de la casilla donde se caiga.

Finalmente, la parte de ficheros también la he realizado ya que ya tenía conocimientos previos en lo que respecta al manejo de ficheros en Java, por lo que la parte de guardar y cargar partida están implementadas, aunque por un bug que no consigo localizar, no consigo representar alguna de la información en el tablero, pienso en arreglar esto cuando haga la aplicación en versión con interfaz gráfica ya que el hacerlo por consola y que quede visualmente bonito me costó bastante de desarrollar.

Finalmente he introducido varias cartas suerte, en concreto 4 tipos diferentes, para salir de la cárcel, volver a la salida, mandar oponente a la cárcel, o que el oponente te done dinero, estas se guardarán en un ArrayList y podrán ser utilizadas durante el juego, me gustaría implementar alguna carta más para la version final.

El juego no se basa en calles, se basa en empresas, como por ejemplo netflix, google o facebook, me gustaría que para la versión final se puedan crear “sedes” una vez se tengan todas las empresas del mismo tipo, haciendo que si el rival cae en una casilla con una sede, tenga que pagar un mayor alquiler.

BIBLIOGRAFÍA

Recursos utilizados durante el desarrollo

StackOverFlow: Para la resolución de dudas concretas con el código.

DiscoDuroDeRoer: Profesor de programación que tiene videos y blogs bastante interesantes que me han ayudado para cosas como por ejemplo la implementación del sistema de ficheros.

Oracle: Consulta de la documentación de las diferentes clases de Java.