Documentación PIA POO

Clase Casillas

Esta clase cuenta con los siguientes atributos, constructor y métodos.

Atributos

- int posicionF: indica la fila en la que la casilla está posicionada.
- int posicionC: indica la columna en la casilla esta posición.
- boolean mina: este indicador sirve para saber si la casilla cuenta con una mina.
- int nMinas: es el número de minas totales que habrá en la partida.
- boolean casillaAbierta: este otro indicador sirve para identificar cuales casillas ya fueron abiertas

Constructor

- Casillas: Tiene como parámetros los atributos posicionF y posicionC y tiene la finalidad de inicializar estos dos atributos.

Métodos

- public int getPosicionF: Este método no cuenta con parámetros y tiene la finalidad de retornar la fila en la que está posicionada la casilla.
- public void setPosicionF: Este método cuenta con un parámetro el cual es la variable posicionF y este le asignara un valor una vez que sea llamado
- public int getPosicionF: Este método no cuenta con parámetros y tiene la finalidad de retornar la columna en la que está posicionada la casilla.
- public void setPosicionF: Este método cuenta con un parámetro el cual es la variable posicionC y este le asignara un valor una vez que sea llamado
- public boolean isMina: No recibe parámetros y su propósito es indicar si la casilla en la que esta hay o no una mina, por medio de true o false.
- public void setMina: recibe un único parámetro el cual es la variable mina y sirve para asignar las minas a las diferentes posiciones del tablero.
- public int getnMinasAlrededor: No incluye parámetros y retorna la cantidad de minas alrededor de una casilla sin mina.
- public void setnMinasAlrededor: Recibe un solo parámetro y sirve para establecer cuantas minas irán alrededor de una casilla que no contenga mina.
- public void incrementarMinasAlrededor: No recibe ningún parámetro y sirve para incrementar en uno la cantidad de minas alrededor de la casilla sin mina.
- public boolean isCasillaAbierta: No cuenta no parámetros y nos retorna un true o false dependiendo del estado en el que se encuentre la casilla.
- public void setCasillaAbierta: Recibe como parámetro la variable boolean casillaAbierta y sirve para establecer true o false en la casilla dependiendo lo que haga el usuario.

Clase TableroBuscaMinas

Esta segunda clase cuenta con las siguientes librerías, atributos, constructor, y métodos.

Librerías

- java.util.LinkedList: Se declaró para poder utilizar LinkedLists o listas enlazadas.
- java.util.List: Su función es la de permitirnos utilizar Listas en el código.
- java.util.function.Consumer: Esta interfaz funcional sirve para representar una operación que acepta un solo argumento de entrada y no produce ningún resultado.

Atributos

- Casillas [][] casillasT: Es un arreglo bidimensional de tipo Casillas la cual es la clase que definimos anteriormente variando su número de filas y columnas dependiendo de la dificultad que escoja el usuario.
- int nFilas: Es un atributo de tipo int que almacena la cantidad de filas que tendrá el tablero de juego.
- int nColumnas: Tiene una función similar al atributo anteriormente mencionado solo que aquí se contendrá la cantidad de columnas del tablero.
- int nMinas: También es una variable tipo int como las 2 antes mencionadas, pero con la característica de que almacena la cantidad de minas que habrá en el tablero.
- int numCasillasAbiertas: Este atributo tipo entero guarda la cantidad de casillas que ya han sido abiertas.
- boolean generacionMinas: esta variable sirve para evitar que el tablero que se cree este vacío.
- consumer<List<Casillas>> PartidaPerdida: Este atributo de tipo consumer se utilizará en las ocasiones en las que el usuario abra una casilla con una mina.
- consumer<List<Casillas>> eventoPartidaGanada: Este otro atributo consumer se usará cuando el usuario logre abrir todas las casillas sin minas.
- consumer<Casillas> CasillaAbierta: esta última variable tipo consumer se usará cada vez que el usuario presione alguna casilla sin mina.

Constructor

- Este constructor recibe los parámetros de nFilas, nColumnas y nMinas, las inicializa junto con la variable generacionMinas que por default debe ser igual a false, además de activar el método de inicializacionCasillas que se verá a continuación.

Métodos

- public void InicializacionCasillas: En este método no se reciben parámetros y tiene como funcionalidad crear el tablero que estará vacío hasta que se ejecuten los siguientes métodos.
- private void generadorMinas: este método recibe dos parámetros de tipo entero los cuales son la posición de la fila y la posición de la columna, este método tiene como finalidad asignar las minas a varias casillas al azar y la variable generacionMinas pasa a ser true además de que por cada mina agregada se llamara al método setMina de la clase Casillas y se marcara esa casilla como true, es decir, que hay una mina ahí.
- public List<Casillas> obtenerTodasLasCasillas: este método se encarga de crear una lista con todas las casillas del tablero, este método se usará más adelante en otros métodos.
- public void imprimirTablero: Este método no recibe ningún parámetro y su funcionalidad es para hacer pruebas, es decir, este método no es necesario para la ejecución del juego como tal, pero sirve para verificar que se haya hecho bien la creación del buscaminas.
- private void imprimirPistas: Al igual que el método anteriormente mencionado no recibe ningún parámetro y se usa para hacer pruebas en este caso se usa para verificar que se hayan asignado la cantidad de minas que se encuentran alrededor de una casilla.
- private void actualizarNumeroMinasAlrededor: No recibe ningún parámetro y se utiliza para aumentar el número de minas alrededor de una casilla que no contenga una mina.
- private List<Casillas> obtenerCasillasSinMinas: Este método recibe dos parámetros la variable que contiene la fila de la casilla, y otra variable que contiene la columna de la casilla, en este método se tiene un ciclo for de 8 iteraciones en el que cada iteración provoca que las variables creadas en este método aumenten o disminuyan su valor o inclusive en algunos casos solo una de ellas tendrá cambios por ultimo este método cuenta con una sentencia if para verificar que no se ingresen casillas fuera de los límites a la lista creada aquí.
- list<Casillas> obtenerCasillasConMinas: Este método no recibe parámetros y se usa para almacenar en una nueva lista aquellas casillas que tengan una mina colocada.
- public void seleccionarCasilla: Este método recibe dos parámetros siendo estos la fila de la casilla y la columna de la casilla, este método es de los más importantes ya que es el activa los eventos en base a que casilla escoja el jugador todo dependiendo de si escoge una casilla con mina, sin mina o sin minas alrededor estos eventos se explican en la clase formatoBuscaMinas que se encuentra más adelante.
- void MarcarCasillaAbierta: este método también recibe los parámetros de la fila de la casilla y de la columna de la casilla y lo que hace es marcar la casilla seleccionada por el jugador como true en lugar de false que era el valor inicial.
- boolean partidaGanada: Este método no recibe ningún parámetro y retornara true o false dependiendo si se cumple la condición.
- public static void main: Este método se uso para realizar las pruebas de la creación del tablero al igual que de la asignación de minas y pistas, sin embargo, no es necesario para la ejecución del juego.
- public void setPartidaPerdida: Este método recibe un parámetro el cual es la variable tipo consumer llamada partidaPerdida y lo que hace es únicamente inicializarla.

- public void setCasillaAbierta: Este método recibe un parámetro el cual es la variable tipo consumer llamada casillaAbierta y su función es inicializarla.
- public void setEventoPartidaGanada: Este método recibe un parámetro el cual es la variable tipo consumer llamada eventoPartidaGanada y lo que hace es únicamente inicializarla.

Clase FormatoBuscaMinas

Esta última clase incluye las siguientes librerías, atributos, constructor y métodos.

Librerías

- java.awt.event.ActionEvent: se usa para representar los diferentes eventos que pueden ocurrir una vez que el usuario presione alguna casilla.
- java.awt.event.ActionListener: sirve para que haya una reacción a los eventos del tipo actionEvent.
- java.util.list: Su función es la de permitirnos utilizar Listas en el código.
- java.util.function.Consumer: Esta interfaz funcional sirve para representar una operación que acepta un solo argumento de entrada y no produce ningún resultado.
- javax.swing.JButton: sirve para poder usar la clase jbutton del paquete javax.swing la cual forma parte de la biblioteca estándar de java para crear interfaces gráficas.

Atributos

- int nFilas: Esta variable representa la cantidad de filas que tendrá el tablero.
- int nColumnas: Esta variable sirve para trabajar con la cantidad de columnas del tablero.
- int nMinas: esta variable indica el número de minas que estarán ocultas en el tablero.
- JButton[][] botones: Es un arreglo bidimensional que variara su cantidad de espacios en base a la dificultad seleccionada por el usuario.
- TableroBuscaMinas tableroBuscaMinas: Esta variable nos permitirá trabajar con el tablero, por ejemplo, con esta variable podemos crear el tablero o asignarle las minas a las diferentes casillas.

Constructor

- Este constructor no recibe ningún tipo de parámetro, pero se usa para ejecutar los métodos de initComponents y juegoNuevo las cuales se explicarán más adelante.

Métodos

- void descargarControles: sirve para evitar una sobrecarga de controles al momento de iniciar una nueva partida.
- private void juegoNuevo: no recibe ningún parámetro y tiene la funcionalidad de ejecutar los métodos descargaControles, cargaControles, crearTableroBuscaminas y por ultimo ejecuta la función repaint.
- private void crearTableroBuscaminas: en este método se establecerá lo que ocurrirá cuando se activen los diferentes eventos mencionados en clases anteriores, cuando se active el evento partidaPerdida se inhabilitaran automáticamente todas las casillas, en el evento eventoPartidaGanada se inhabilitaran todas las casillas una vez que se complete la partida y las casillas con minas pasaran a mostrar “:)”, por último el evento casillaAbierta

inhabilitara aquellas casillas seleccionadas por el usuario y en algunos casos varias casillas que se encuentren alrededor.

- private void cargaControles: este método no recibe ningún parámetro y tiene la finalidad de cargar los controles como su nombre lo dice y además sirve para ajustar el tamaño de cada casilla.
- private void btnClick: recibe un parámetro de tipo ActionEvent y tiene la finalidad de mostrar una pequeña interfaz que muestra la coordenada de la casilla que el usuario esta apunto de abrir, pero como quería hacer mas fluido el juego en el sentido de que sea más cómodo para el usuario resolver el buscaminas deje comentada la línea que muestra la interfaz anteriormente mencionada.
- private void DificultadDificilActionPerformed: recibe un parámetro de tipo java.awt.event.ActionEvent y tiene su función es la de asignar la cantidad de filas, columnas y minas, luego de esto se ejecuta el juego con las especificaciones dadas.
- private void DificultadMediaActionPerformed: recibe un parámetro de tipo. java.awt.event.ActionEvent, este método es muy similar al anteriormente mencionado solo que aquí la cantidad de filas, columnas, y minas es menor.
- private void DificultadFacilActionPerformed: recibe un parámetro de tipo java.awt.event.ActionEvent, este método es muy similar a los dos anteriormente mencionados con la diferencia de que aquí la cantidad de filas, columnas, y minas es todavía menor a las de la dificultad media.

Para terminar esta última clase se declararon los siguientes atributos:

- private javax.swing.JMenuItem DificultadDificil;
- private javax.swing.JMenuItem DificultadFacil;
- private javax.swing.JMenuItem DificultadMedia;
- private javax.swing.JMenu jMenu1;
- private javax.swing.JMenuBar jMenuBar1;

Todos estos atributos son los diferentes componentes del menú de la interfaz del buscaminas que se llevó a cabo.

Clase CasillasTest

En esta clase se incluyeron las siguientes librerías y unit tests.

Librerías

- org.junit.Test: Esta librería nos permite usar la anotación @Test, que se coloca sobre un método para hacer énfasis en que es un unit test.
- static org.junit.Assert.*: Esta librería sirve para importar todos los métodos estáticos pertenecientes a la clase Assert.

Unit Tests

- public void testIsMina: Se usó para verificar que al marcar una casilla como mina con setMina(true), el método isMina realmente devuelva true.
- public void testSetMina: Este test sirve para verificar que realmente se asignó una mina.
- public void testIsCasillaAbierta: Su propósito es el de verificar que, si se marca una casilla como abierta, el método isCasillaAbierta devuelva true.
- public void testSetCasillaAbierta: Valida que el atributo casillaAbierta al momento de abrir la casilla pase a ser true.