UNIVERSIDAD NACIONAL DE GENERAL SARMIENTO

Trabajo Práctico 1

Integrantes:

Ezequiel Tomas Ramallo

Legajo: 36827376/2017

E-mail: ezeramallo@gmail.com

Alejandro Daniel Pacheco

Legajo: 26478545/2017

E-mail: ale1202@gmaIl.com

Profesores:

Patricia Bagnes

Javier Marenco

Asignatura: Programación III – Primer Cuatrimestre 2020

Comisión: COM-1

Fecha de entrega: 14/04/2020

Introducción

El presente informe tiene como objetivo exponer, de forma breve, el desarrollo de la aplicación “juegos aritméticos”, el cual consisten en rellenar una matriz con números de formar que la sumas de estos, fila por fila y columna por columna, coincidan con valores asignados previamente a esas filas y columnas.

Al iniciar la app ,el usuario tendrá que ingresar el tamaño del tablero, el cual deberá respetar los valores dentro de un intervalo acotando de números naturales comprendido entre 4 y 8, tanto para filas como columnas (4<=Filas<=8 /\ 4<=Columnas<=8 }. Luego de la elección de la dimensión del tablero, se crearan una cantidad de botones que se corresponden a la multiplicación de la cantidad de filas por columnas elegidos (Cantidad_de_Botones = Filas*Columnas) y se iniciara un contador de segundo. Estos botones, cumplirán la función de recibir los valores a ingresar en el tablero mediante el uso de un panel, con valores prefijados del 1 al 9, que se abrirá al hacer click sobre el botón que ocupe lo posición en el que el usuario quiera hacer su jugada. En caso de que las sumas de la columnas y filas de los valores ingresados al tablero de coincidan con los valores prefijado en la inicialización del juego, la interfaz mostrar un mensaje indicando al usuario que gano.

Implementación

Para llevar a cabo implementación de esta aplicación decidimos utilizar el patrón MVC (modelo-vista-controlador), el cual nos permite separar el código de negocio del código de la interfaz. Para llevar a cabo la implementación de este patrón, decidimos dividir el proyecto en 4 package: modelo, controlador, vista y MVCPrincipal.

- **Paquete modelo**
  - **clase “LogicaJuego”**: clase encargada de la inteligencia de la aplicación. En ella utilizamos 3 variables globales, dos de ellas son arreglos de enteros utilizados para almacenar valores de filas y columnas prefijados mediante el uso de una función de esta clase; mientra que la otra, sera una instancia a la clase “Matriz” (ubicada en el mismo paquete), la cual se encarga de reflejar, de una forma lógica, al tablero de la parte grafico . Las funciones mas implementadas en “LogicaJuego” serán las siguiente :
    - verificarDimensionDelJuego: verificar el rango acotado a la hora de la creación del tablero mediante el uso de excepciones
    - generarValorReferecia: generar los valores predeterminados de las filas y columnas utilizados al inicio del juego mediante el uso de una matriz con valores aleatorios.
    - Ganador: comprueba la finalización del juego mediante la comparación entre los valores prefijados de filas/columnas al inicio del juego y los valores de filas/columnas generador con los números ingresados.
  - **clase “Matriz”**: clase auxiliar,utilizada para poder realizar la funciones requeridas en la clase LogicaJuego. En ella utilizaremos 3 variables, un arreglo bidimensional y 2 variables enteras que almacenaran la cantidad de filas y columnas de dicho arreglo. La funciones mas importante que encontrar en esta clase serán:

- IngresarValor: ingresa un entero al arreglo en una fila y columna indicado por parámetro
- Suma_Columna / Suma_Fila: suma de los valores de una fila o columna indicada .
- GenerarMatrizRandom: genera una matriz con valores aleatorios.

- **Paquete controlador**
  - **clase "Controlador"**: esta clase implementa la interfaz ActionListener, y se encarga de funcionar como una especie de intermediario entre el código de interfaz y el código negocio; es decir, mediante la reacción a un evento producido en el interfaz grafica, esta clase de encargar de actualizar y/o procesar el código de negocio para la posterior actualización de la parte grafica.. En ella tendremos dos variables que harán referencia tanto a la clase encargada del código de inteligencia como a la clase del código encargado de la parte grafica. Las funcionalidades que cumple serán:
    - ActualizarValor: actualizar el valor el tablero logico ubicado en el package modelo.
    - Ganador: notificar a la interfaz grafica si el juego ha finalizado(si hay ganador).
    - ActionPerformed: método que reacciona ante un evento(proviene de la interfaz ActionListener). Este se encargara de settear los valores de inicio de juego en el código de negocio (tamaño de tablero, valores predefinidos de las filas/columnas, ect) y actualizar la parte grafica en consecuencia a esto.

- **Paquete vista**
  - **clase "JbuttonConIndice"**: esta clase amplia a JButton de manera que el componente se pueda relacionar una coordenada de dos dimensiones (X,Y).
  - **clase "BTNNumElegido"**: es un clase que amplia un Jdialog y se desarrollo utilizando el wWndowBuilder. En ella tendremos un panel de 9 botones con una representación numérica del 1 al 9 que funcionaran como selectores de los posibles valores de ingreso de la interfaz grafica
  - **clase "Botones_Eleccion"**: clase que implementa ActionListener, y se encarga de trasladar el valor elegido en la ventana modal a la ventana principal del juego; ademas, se encarga de actualizar y controlar la finalización del juego en el código de inteligencia mediante el uso de la clase "Controlador"
  - **clase "GUIJuego"**: en esta clase se encuentra implementada la ventana grafica principal juego, la cual contendrá los componentes(arreglo de JButton que representen el tablero interactivo, JTxtField para ingresar el tamaño de la matriz, arreglos de JLabel para colocar los valores prefijados de filas y columnas, ect) utilizados para interactuar con el código de negocio de juego. Las funcionalidades mas importantes de esta clase son:
    - IniciarButtonsEvento / Inicia_Labels_Col_Fil: ambos métodos inicializan los arreglo sus componentes en sus respectivos paneles dentro del tablero
    - GUI: este método se encarga de distribuir y acoplar los paneles de componentes en el frame, así como también de plasmar el tablero de juego.
    - CreaGUI: método encargado de inicial la aplicación

- **Paquete MVCPrincipal:**
  - clase “Principal”: esta clase contendrá el main utilizado para aunar las partes del modelo, vista y controlador, a fin de lanzar la aplicación.

Problemas durante el desarrollo y decisiones tomadas

- Al principio del proyecto intentamos implementar el patrón de diseño MVC utilizando la extensión WindowBuilder del eclipse, cosa que nos resulto imposible, ya que no podíamos settear al controlador, antes de la inicialización del la aplicación, con referencias del codigo de vista y modelo. Debido a este inconveniente decidimos quitar el main predeterminado de la “aplication window” de la clase GUIJuego e implementarlo en una clase aparte.

- Tambien, tuvimos un problema conceptual, que refiere al situar del del actionPerformed de la clase “Botones_Eleccion”. Este problema consistía en la elección de package, si era correcto dejarlo en la paquete de vista, ya que dentro del método se convocaba a una ventana modal, o bien, trasladarlo al paquete de control, debido a que contiene código que requiere la interacción con el código de negocio. Finalmente, optamos por dejarlo en el paquete vista y utilizar funciones implementadas en la clase Controlador como medio de interacción con el código de inteligencia.

- Por ultimo, nos encontramos con problemas a la hora de implementar un tablero redimencionable usando la extención WindowBuilder. Debido a eso, tuvimos que prescindir de la utilización de esta extensión y codificarlo a mano, por ende, la parte desing de la “aplication window” no reflejara ningún arreglo de componentes.