Proyecto 3

Alejandro Hernández Cano
Martínez Santana Brayan
Trad Mateos Kethrim Guadalupe

22 de octubre de 2019

Repositorio: `https://github.com/BrayanCaro/BuscaminasVala`

1. Introducción

Buscaminas (Hecho en Vala)
Presentamos un proyecto completo con pruebas, documentación de un buen juego buscaminas que busca mejorar el que ya teníamos hecho al inicio de la carrera, haciendo un código un poco más compacto, conciso y eficiente si usamos una parte gráfica para no estar imprimiendo muchas veces en la terminal el tablero.

2. Definición del problema

Diseñar el videojuego Buscaminas donde también podamos guardar la partida.
En comparación con el proyecto anterior la idea es la misma, aunque intentaremos que el programa pueda funcionar en una pequeña ventana.

3. Análisis del problema

El juego tiene que ser capaz de:

1. Guardar la partida
2. Tener en una tabla los mejores puntajes
3. Dar la oportunidad de que el usuario decida el tamaño del tablero

4. Mejor alternativa

Ahora nuestro proyecto usaremos una interfaz gráfica para que el usuario solo tenga que seleccionar los botones sin necesidad de meter coordenadas

5. Pseudocódigo

En esta sección describiremos los algoritmos utilizados para hacer posible este proyecto. Hay que tomar en cuenta que todos estos algoritmos están encapsulados dentro de una clase `Tablero`, por lo tanto hacemos la suposición de que al inicio de cada algoritmo, el tablero se encuentra en un estado válido, además, dentro de estos, denotaremos por n y m la cantidad de filas y columnas en el tablero, respectivamente y por *casillas* la matriz que representa las casillas del tablero.

Algorithm 1 Presionar casilla

```
1: procedure PresionarCasilla(x, y)
Require: CasillaValida(x, y)
2:     casillas[x, y].marcarPresionado( )
3:     Extender(x, y)
4: procedure Extender(x, y)
5:     if casillas[x, y].bombasVecinas > 0 then                ▷ Caso base
6:         return
7:     coords ← [(x, y + 1), (x − 1, y), (x + 1, y), (x, y − 1)]   ▷ Vecinos arriba/abajo y de los lados
8:     for all (i, j) ∈ coords do
9:         if CasillaValida(i, j) then
10:            c ← casillas[i, j]
11:            if not c.minado and not c.abanderado and not c.presionado then
12:                c.marcarPresionado( )
13:                Extender(i, j)
14: procedure CasillaValida(x, y)
15:    if 0 ≤ x < n and 0 ≤ y < m then
16:        return true
17:    return false
```

6. Pruebas y mejoras

Planeamos que el programa pueda ser más amigable con el usuario usando una ventana y que se pueda interactuar con el tablero seleccionando botones

7. Pensamiento a futuro

El programa podría ponerse en las tiendas de alguna plataforma de software como en 'Software de Ubuntu' porque Vala esta diseñado para la creación de aplicaciones (usando las bibliotecas de GNOME).

Bibliografía

1. Tutorial de Vala
   https://wiki.gnome.org/Projects/Vala/Tutorial/es
2. Introducción a Vala
   https://www.youtube.com/watch?v=hkUnly_Viys