GUÍA DE WITCHES MAZE RUNNERS

Ana Laura Oliva Avilés C111

Dependencias

dotnet 8.0: https://dotnet.microsoft.com/es-es/download/dotnet/8.0
Powershell: https://learn.microsoft.com/es-es/powershell/scripting/install/installing-powershell-on-windows?view=powershell-7.4
Spectre Console: https://spectreconsole.net/quick-start
Visual Studio Code: https://code.visualstudio.com/download

Sobre el juego

Para terminar su entrenamiento mágico y lograr ser miembros oficiales del aquelarre, las brujas y brujos que acaban de recibir la marca deben embarcarse en la aventura del sendero de las brujas (camino al infierno) y salir con vida del mismo. Este sendero no es más que un laberinto lleno de trampas que pondrá a prueba las habilidades de los recién marcados. Antes de comenzar todos son advertidos de que si pierden todos sus puntos de vida tendrán que esperar dos turnos y tendrán que volver al inicio y que solo la mitad podrá salir de ahí, el resto tendrá que intentarlo otro año. Tu misión, es ayudar a tu brujo o bruja a ganar.

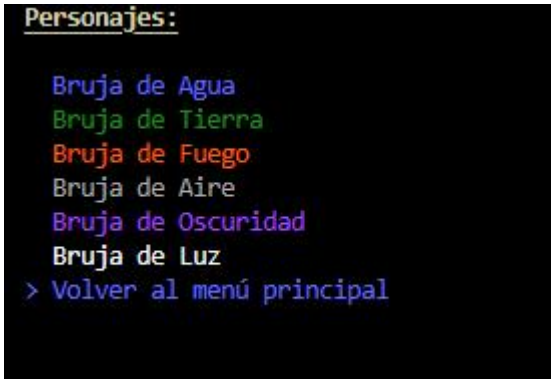
Menú principal

Tendrás tres opciones:

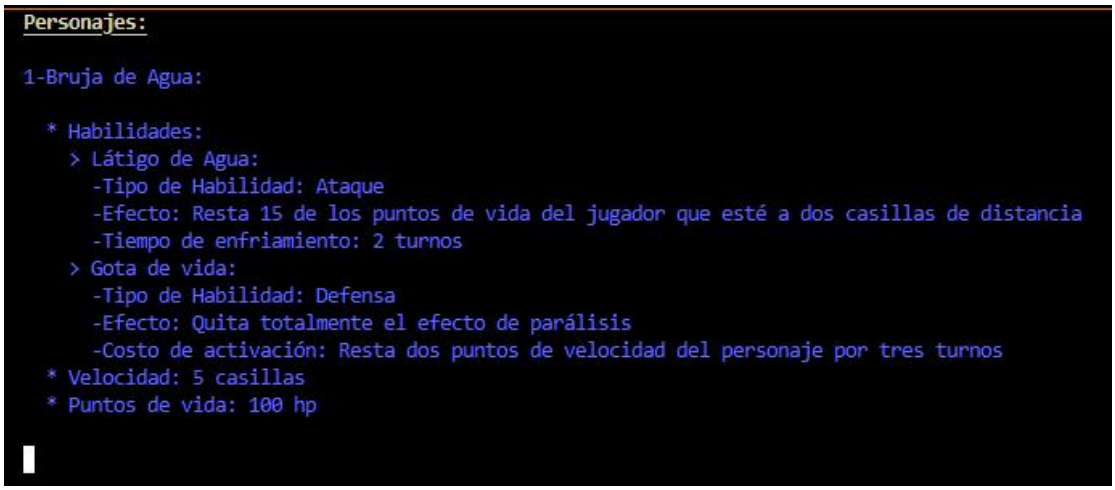
- Nuevo Juego: Al seleccionarse comienza una nueva partida
- Personajes: Muestra el menú de personajes
- Salir: Al seleccionarse saldrás del juego

Nota: Para moverte entre las opciones tendrás que utilizar las flechas del teclado

Personajes (Menú):
Cuando te salga el menú de personajes y selecciones a una bruja te aparecerá la información de la misma



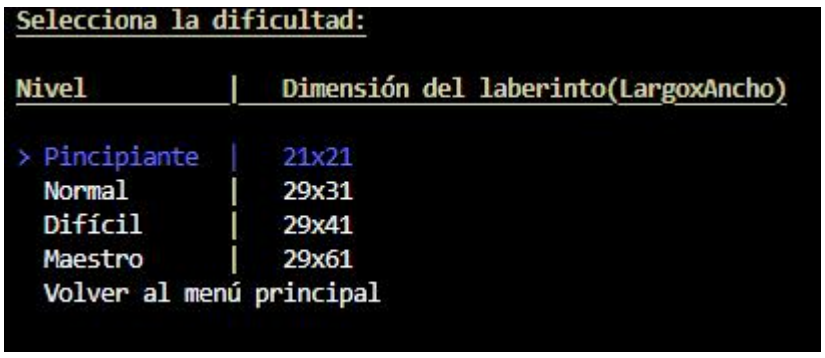
Cuando seleccionas una de las brujas te sale la info del personaje con el color respectivo de su magia.



Para volver al menú de Personajes presiona cualquier tecla y para ir al menú principal a partir del de Personajes selecciona la opción Salir al menú principal

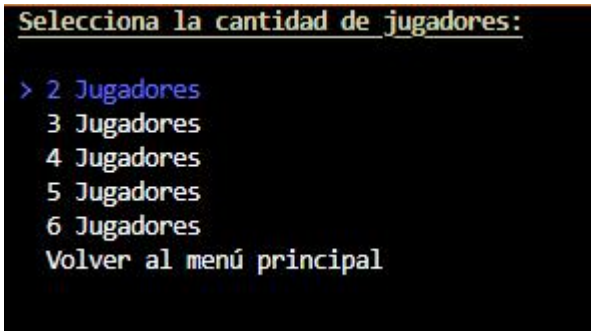
Nuevo Juego:
Seleccionar dificultad:

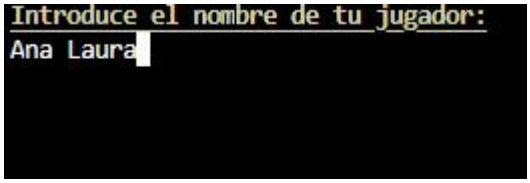
Al seleccionar la opción nuevo juego aparecerá un menú para seleccionar la dificultad del juego: A un lado del nivel aparecerán las dimensiones del laberinto, cabe recalcar que la cantidad de trampas también es proporcional al tamaño del laberinto

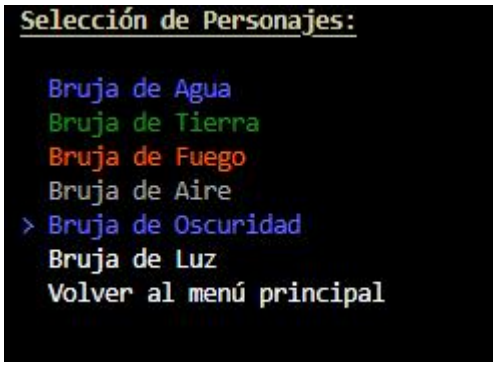


Cantidad de jugadores y selección de personajes:

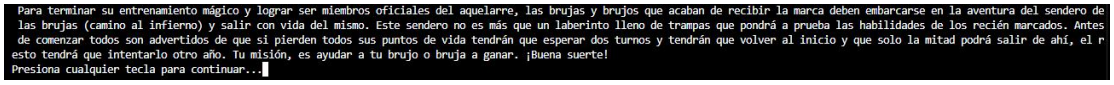
Al seleccionar una dificultad aparecerá un menú con distintas opciones de cantidad de jugadores. Una vez que lo elijas el juego te pedirá tu nombre. Cuando lo pongas presiona enter y saldrá un menú con las brujas disponibles, aclaración, una vez que elijas una bruja cuando el siguiente jugador vaya a elegir la suya verá que la anteriormente seleccionada no aparecerá entre las brujas elegibles.







La partida:
Al iniciar el juego va a aparecer un mensaje con la razón por la cual estás en el laberinto:



Posteriormente aparecerá el laberinto. En la primera ronda, la ficha de la bruja no aparecerá hasta que sea su turno.

Controles:
Movimiento:
- Arriba: W o Flecha arriba
- Abajo: S o Flecha abajo
- Izquierda: A o Flecha Izquierda
- Derecha: D o Flecha Derecha

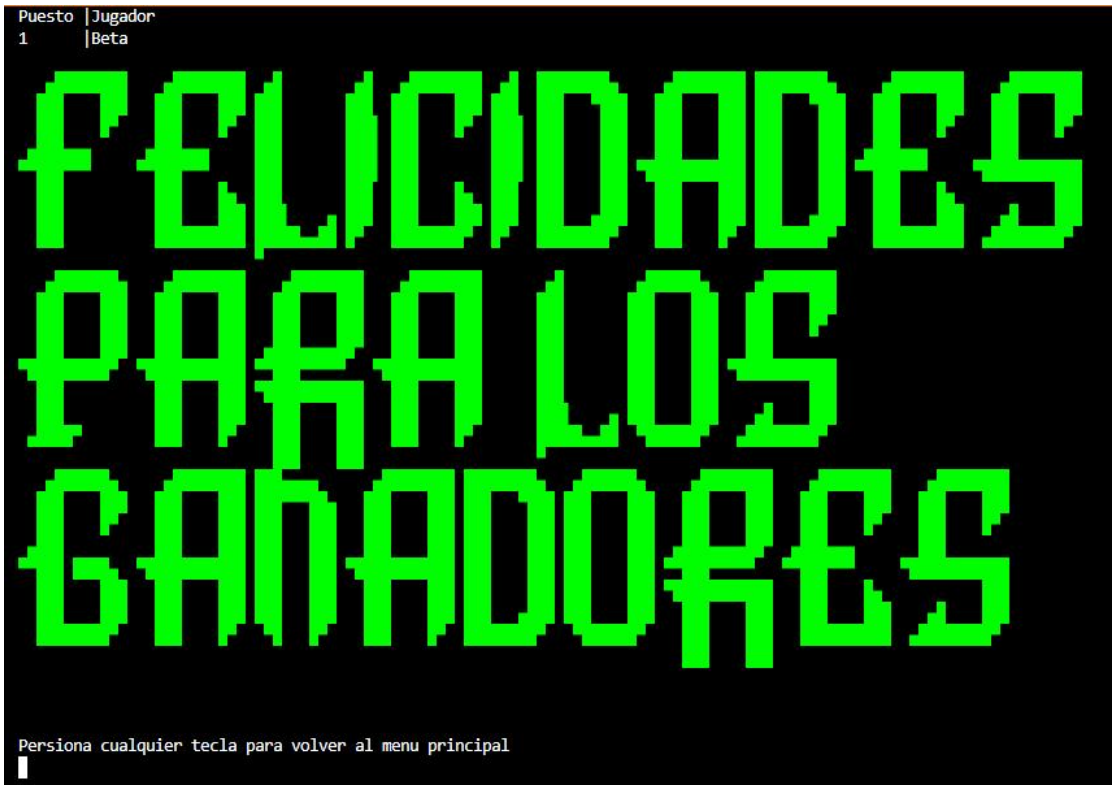
Ataque: L
Defensa: K

Nota: La bruja de oscuridad que se quita 50 hp cuando ataca no puede volver a atacar hasta que no tenga más de 50 hp.

Trampas o tesoros:
Son representados por una estrella y existen 6 efectos distintos:
-Trampa de congelamiento: el jugador no podrá moverse por tres turnos.
-Trampa de teletransportación: el jugador es teletransportado al punto de inicio.
-Trampa de pérdida de velocidad: la velocidad se reduce a un movimiento.
-Trampa de daño: resta 15 puntos de vida al jugador.
-Tesoro de sanación: recuperas 10 puntos de vida (si tienes más de 90 recuperas lo que te queda para que sean 100).
-Tesoro de recuperación: anula el efecto de la pérdida de velocidad tanto de la trampa como del tiempo de enfriamiento de la defensa.

Muerte del jugador:
Cuando el jugador pierde todos sus puntos de vida será teletransportado automáticamente al punto de inicio y tendrá que esperar dos turnos antes de poder volver a jugar. Cabe recalcar que si todos los jugadores están congelados pasarán los turnos hasta que uno se descongele.

Ganadores:
El juego termina cuando la mitad de los jugadores gana y aparecerá un cartel con el podio y un mensaje de felicitación



Una vez terminada la partida volverás al menú principal, si quieres seguir jugando puedes volver a presionar nuevo juego o salir para salir del juego.

Sobre el código:

Dividido en tres partes principales:

- Game.Model: Parte lógica del juego. Contiene las clases:GameModel, Cells, Gameboard, Maze, TrapsAndTreasures, Players, la clase abstracta Witches y las clases que heredan de la misma.
- Game.Visuals: Contiene la parte visual del juego. Contiene las clases: GameVisuals, MenuVisuals y MazeVisuals. Estas tres clases son las que contienen código de la biblioteca Spectre Console
- Game.Controller: Es el intermediario entre el usuario, Game.Visuals y Game.Model. Contiene a la clase GameController.

Carpeta Witches:
Está formado por la clase abstracta Witches, la clase Player y las clases que heredan de Witches (fichas).

La clase Witches contiene los métodos abstractos Attack que devuelve un entero con el daño del ataque; Defense que representa la actividad de la habilidad defensiva; IsInmune que indica si el jugador es inmune al daño por ataque; IsDefrost que indica si el jugador es inmune a las trampas de congelamiento; RecoverSpeed() recobrar los puntos originales de velocidad del personaje; DefenseUsed() se activa una vez que el jugador activa la defensa. Estos métodos se reescriben en las clases que heredan de Witches con sus datos específicos. También están los métodos LossOfSpeed que reduce la velocidad a 1; restoreSpeedTreasure que reduce el tiempo de enfriamiento de la defensa haciendo posible que se restauren los puntos de velocidad; DefenseCoolingTimeOver que devuelve un bool indicando si el tiempo de espera de la activación de la defensa acabó y ReduceCoolingTime que reduce en 1 el tiempo de espera tanto de la defensa como del ataque si es posible.

Las clases que heredan de Witches al reescribir Attack verifican si es tiempo de enfriamiento es 0 y dependiendo de eso devuelven el daño o 0 menos la clase Darkness que no tiene tiempo de enfriamiento y el coste por atacar es otro y al reescribir Defense se comprueba que la velocidad sea mayor estricto que 3 y que el tiempo de enfriamiento sea 0.

La clase Player representa al jugador. Contiene los métodos Effects que indica si el jugador está bajo los efectos de la trampa de congelamiento; ReduceEffects que reduce en 1 de ser posible los efectos de las trampas, llama al método ReduceCoolingTime de la clase Witches y finalmente restaura los puntos de velocidad cuando el efecto de la trampa de pérdida de velocidad y el tiempo de espera de la defensa llegan a 0; Attack que recibe los puntos de daño del método de igual nombre de la clase Witches, si el daño es 100 y el HP es mayor estricto que 50 se restan 50 puntos de vida de lo contrario no se realiza el ataque esto es debido al personaje de la Bruja de Oscuridad con el objetivo de que no se mate a sí misma si tiene menos de 50 puntos de vida, si es posible atacar añade el nombre de la habilidad a la lista de mensajes y le resta el daño a cada uno de los jugadores a menos de que sean inmunes en cuyo caso activa el método DefenseUsed de la clase Witches; Defense que activa el método Defense de la clase Witches y en caso de que dicho método haya devuelto un entero distinto de 0 el HP se actualiza a ese entero; UpdatePlayerPosition que actualiza la posición actual del jugador; GetPlayerPosition que devuelve la posición actual del jugador, PlayerDeath que comprueba si el jugador murió, agrega dos turnos de efecto de congelamiento y reinicia los puntos de vida si en efecto murió; ChangeHP recibe un entero positivo o negativo que afecta el HP del jugador; y TrapsEffects que actualiza los efectos de las trampas.

Carpeta Gameboard:

Está formada por las clases TrapsAndTreasures, Cells, Maze, Gameboard

La clase TrapsAndTreasures es la representación de las trampas o tesoros del laberinto tiene un único método llamado FallIntoTrap el cual tiene un switch que representa las trampas y los efectos. Códigos de las trampas: 0 Teletransportación te teletransporta a la casilla de inicio, 1 Congelamiento a menos que tu defensa sea descongelarte y esté activada quedas congelado por tres turnos, 2 pérdida de velocidad solo podrás moverte una vez por turno, 3 daño te quita 15 puntos de vida, 4 recuperación de HP recuperas 10 puntos de vida, 5 recuperar velocidad, anula los efectos de la trampa de pérdida de velocidad y reduce a 0 el tiempo de enfriamiento de la defensa.

La clase Cells representa las celdas del laberinto. Tiene los métodos PutTrapsOrTreasure inicializa una trampa en la celda y vuelve verdadero el bool que indica que hay una trampa en la casilla; FallIntoTrap quita la trampa y llama al método FallIntoTrap de la clase TrapsAndTreasures; MakeAvailable vuelve disponible la celda.

La clase Maze es la encargada de crear el laberinto. Inicializa todas las celdas del array bidimencional de celdas como no disponibles, después obtiene dos números impares aleatorios, uno para las filas y otro para las columnas y vuelve la celda que coincide con esa posición disponible. Posteriormente se crea una pila y se almacena la posición inicial en ella y entramos en un ciclo que termina cuando la pila se queda sin elementos. El ciclo consiste en obtener una de las celdas vecinas al azar, comprueba si esa celda no es (0,0) ya que eso significa que no hay celdas vecinas, si lo es, saca el primer elemento de la pila, si no, hace disponible la celda en las coordenadas de la celda vecina y esa posición es almacenada en la pila, este proceso se repite continuamente hasta que se termina el ciclo.

Gameboard es la clase que representa el tablero de juego, verifica si una celda no es pared con el método VerifyAvailability, FallIntoTrap devuelve el valor de la propiedad Traps de una celda específica del laberinto, Effects activa el método de mismo nombre de la clase Cells y por último PutInitialPosition que devuelve la fila inicial del laberinto.

Carpeta Visuals:
Componentes visuales del juego, únicas clases del laberinto que contienen código de Spectre Console.

MenuVisuals es la clase que representa los menús desde el menú principal, el de selección de la dificultad, el de selección de personajes y jugadores.

MazeVisuals es la clase que representa el visual de la partida, imprime el laberinto y el prólogo.

Carpeta MVC
Contiene el GameModel, el GameVisuals y el GameController, clases principales que controlan al resto de las clases.

La clase GameModel es la encargada de inicializar a los personajes en el laberinto, revisa si alguien ha muerto después de recibir un ataque, un método que devuelve si alguien ha alcanzado la condición de victoria, elimina un jugador, su método MakeAValidMode verifica si el movimiento es válido y si lo es actualiza la posición

del personaje, revisa si cayó en una trampa y por último revisa si murió, además de otros métodos que agregan un jugador, obtienen el turno actual, la velocidad o la lista de personajes y pasan al siguiente turno.

La clase GameVisuals es la clase visual principal actúa como mediador entre el GameController y las otras dos clases visuales. Además contiene el método de confirmación de si se desea volver al menú principal y el de imprimir los ganadores.

La clase GameController obtiene la información del usuario, hace cambios en el GameModel y los muestra por medio del GameVisuals. Lo principal de esta clase es el método newGame, que contiene el ciclo principal del juego. Este consiste en inicializar los personajes en el laberinto cuando es el primer turno y aún no se han realizado movimientos por medio de un if, mandar al GameVisuals a imprimir el laberinto y los mensajes, ver si el personaje se puede mover (caso positivo se quita un movimiento, caso negativo los movimientos se hacen 0, caso nulo se sale al menú principal), verifica si el jugador alcanzó la condición de victoria (caso positivo se quita ese personaje de la lista de personajes, se añade su nombre a la lista de ganadores y vuelve al inicio del bucle, verifica si se activó la defensa y por último se verifica si se cayó en una trampa de pérdida de velocidad o si se le acabaron los movimientos (se reducen los efectos de las trampas y tiempo de enfriamiento de la defensa y el ataque, se cambia de jugador y se actualizan los movimientos restantes. El ciclo acaba cuando la cantidad de nombres en la lista de jugadores es igual a la parte entera de la mitad de jugadores. El otro método importante es el MoveOrSkill, que recibe la información de lo que desea hacer el usuario: moverse, atacar o activar la defensa y por medio de un switch cambia determinados aspectos de las clases del Game.Model.