Resumen final Proyecto laberinto 2024/2025

Pablo García Rubio

Ajustándome a los requisitos, te informo de dónde encontrar esta documentación

1. **Archivo JSON: todas las modificaciones que supongan elementos nuevos deben incluirse en el archivo de configuración, es decir, se debe usar el Builder para construir el juego.**

Podrás encontrar todos los ficheros JSON utilizados en el desarrollo de la aplicación en la carpeta /Laberintos. Si quieres cargar otro laberinto distinro, no tienes más que cambiar la referencia al fichero JSON en Program.cs del proyecto Consola.

1. **Control de errores y excepciones: Un error o excepción de una extensión implica descender de categoría**

¿Errores? ¿Qué errores? ;)

1. **Definir las condiciones de fin de juego: (1) Pierde el personaje (2) Gana el personaje (indicar cómo gana el personaje)**

Tal y como he contado en el vídeo, hay un sistema de turnos y el personaje gana si es capaz de quitarle la vida a todos los bichos. Si el personaje pierde todas las vidas, habrá perdido.

1. **Todas las extensiones deben funcionar en la consola (Playground), es decir, con independencia de la interfaz gráfica**

Para eso he creado en Program.cs del proyecto Laberinto.Consola, un menú con la navegabilidad.

1. **Test: Todas las modificaciones deben estar respaldadas por proyectos de test**

Hay un proyecto específico para el entorno de pruebas en el código.

1. **Diagramas: mostrar el diagrama de clases y al menos un diagrama de secuencia (que se corresponda con interacciones entre objetos de alguna de las extensiones que se presenten).**

Se han añadido clases derivadas directamente de ElementoMapa, como Lampara, Cuadro y Pocima, cada una representando objetos específicos dentro del juego que ofrecen interacciones particulares al jugador.

Además, la incorporación del patrón Decorator mediante la clase abstracta Decorator y su implementación concreta Bomba permite añadir dinámicamente comportamientos especiales a los elementos del mapa, como la explosión que afecta al personaje al entrar en ciertas habitaciones.

Otra mejora destacada ha sido la definición clara de relaciones de composición y agregación. Por ejemplo, Armario ahora posee una referencia explícita a su Habitacion madre, indicando dónde se encuentra colocado, mientras que la clase Puerta conecta explícitamente dos instancias de Contenedor (generalmente habitaciones). Además, la clase Pocima está conectada con ComandoTomarPocima, mostrando claramente cómo se usa el patrón Command para ejecutar acciones concretas sobre el personaje, como beber la poción para recuperar vida.

Diagrama

El contenido generado por IA puede ser incorrecto.

He dejado una copia del diagrama en la carpeta raíz.

Por último, si te es más cómodo, puedes clonar el repositorio de esta dirección:

<https://github.com/PabloGR75/laberinto>