Memoria del Proyecto

Jorge García Martínez

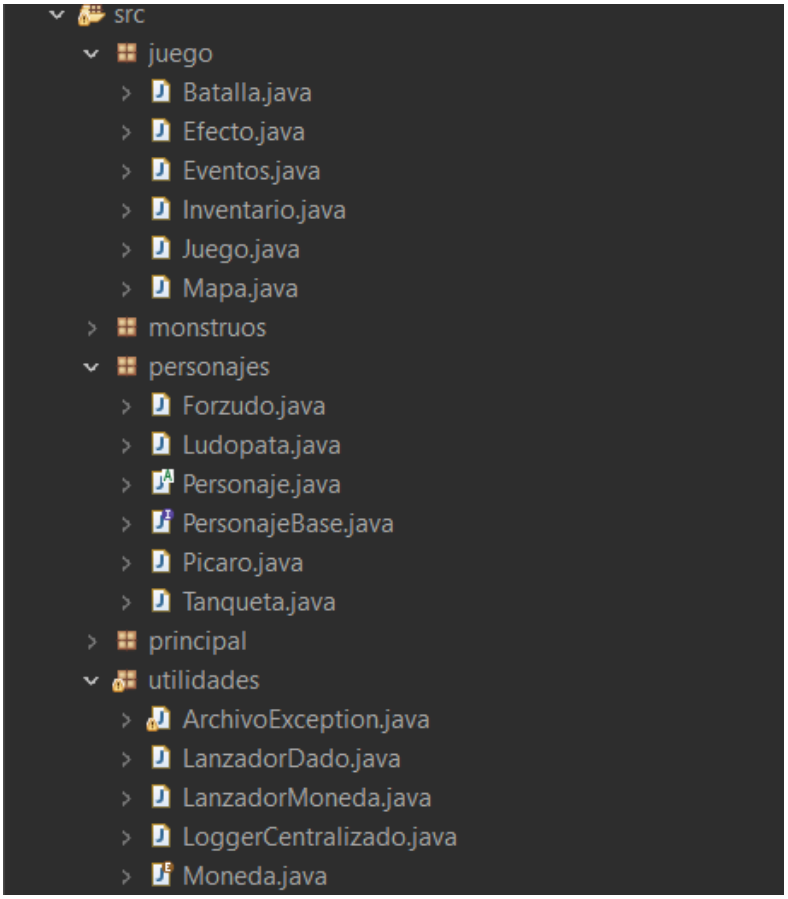
1. Introducción

Este proyecto se basa en la implementación de un juego tipo *Battle Royale* por terminal, inspirado en las bases de una guía de desarrollo. El objetivo principal fue crear un sistema funcional y escalable que permita gestionar personajes, combates, eventos y un sistema de turnos, terminando con unjuego donde el último jugador en pie es el ganador.

La estrategia de desarrollo se centró en construir una versión mínima funcional y expandir progresivamente, agregando elementos y mejoras tanto propias como sugeridas por la guía. El resultado final consta de cinco paquetes que organizan las distintas funcionalidades del juego.

2. Metodología de Desarrollo

Para abordar el proyecto, seguí el siguiente enfoque:



1.Diseñé y desarrollé un modelo básico con las funcionalidades básicas.

2.Extendí el código, agregando características y mejorando la lógica.

3.Registré los cambios y tareas pendientes en un archivo de texto, lo que facilitó la organización.

4.Realicé pruebas para identificar errores, mejorar el balance del juego y optimizar el código.

La estructura inicial incluía una clase padre `Personaje`, que implementa una interfaz y define atributos y métodos comunes. Posteriormente, se integró un sistema de turnos gestionado por la clase `Juego`, así como clases adicionales para los combates, eventos y otros elementos.

3. Estructura del Proyecto

El código se organizó en cinco paquetes:

3.1 Paquete `juego`

Contiene las clases principales de la lógica del juego:

- **`Juego`**: Gestiona el lobby, los turnos y las acciones disponibles para cada personaje.
- **`Batalla`**: Controla los enfrentamientos entre personajes, incluyendo las opciones de ataque, habilidades especiales, huida y uso de objetos.
- **`Eventos`**: Implementa la exploración, permitiendo a los jugadores obtener herramientas y objetos para el inventario y enfrentarse a monstruos.
- **`Inventario`**: Gestiona el almacenamiento de objetos para cada personaje.
- **`Mapa`**: Introduce una funcionalidad adicional para variar los eventos según la ubicación del jugador.
- **`Efecto`**: Aplica los efectos de los objetos sobre los atributos de los personajes.

3.2 Paquete `monstruos`

Implementa enemigos que aparecen durante los eventos de exploración. Hereda la estructura de la clase padre `Personaje`.

3.3 Paquete `personajes`

Incluye la interfaz `PersonajeBase`, la clase abstracta `Personaje` y las subclases específicas de cada tipo de personaje, que definen habilidades especiales y atributos personalizados.

3.4 Paquete `principal`

Contiene la clase `Inicio`, que ejecuta la lógica principal del juego.

3.5 Paquete `utilidades`

Agrupa herramientas y funcionalidades auxiliares:

- **`LanzadorDado`**: Simula el lanzamiento de dados, como un d20 utilizado en la clase `Eventos`.

- `LanzadorMoneda`: Simula un lanzamiento de moneda para determinar el inicio de las batallas.
- `LoggerCentralizado`: Registra todas las acciones del juego en un archivo de texto.
- `ExcepcionPersonalizada`: Maneja errores específicos del juego.
- `Moneda`: Define un *enum* para las opciones de cara y cruz.

4. Principales Implementaciones

4.1 Clase `Personaje`

Define los atributos comunes (vida, ataque, velocidad, defensa, etc.), si está controlado o no por el jugador y los métodos generales, como:

- **Atacar:** Calcula el daño infligido a otro personaje considerando atributos como ataque y defensa.
- **Habilidad especial**: método abstracto que definirán las clases hijas.
- **Descansar:** Restaura atributos como la vida.
- **Inventario:** Implementado con un HashMap para gestionar objetos mediante pares clave-valor y usarlos.
- **Efectos:** Gestiona los buffs y debuffs del jugador.
- **Herramienta:** Gestiona las herramientas del jugador, con sus efectos.

4.2 Clase `Juego`

Gestiona:

- La creación de personajes mediante un sistema interactivo por terminal.
- Los turnos cíclicos, permiten a los jugadores atacar, descansar, explorar o usar objetos.
- La gestión de personajes y eliminación de aquellos con vida igual a cero.
- El inicio y el final del juego, dando el ganador.
- Otros elementos de la lógica de la partida.

4.3 Clase `Batalla`

Controla los enfrentamientos, ofreciendo opciones como:

- Atacar.
- Usar una habilidad especial.
- Huir del combate..
- Utilizar herramientas del inventario.
- Implementa el lanzamiento de moneda para decidir quién comienza.

4.4 Clase `Eventos`

Introdujo exploración con:

- Gestión de las herramientas y objetos que modifican atributos y su obtención.
- Enfrentamientos contra monstruos controlados por la máquina.
- Uso del dado d20 para determinar los eventos.
- Uso el mapa para añadir más variedad.
- Añade una función que permite el retraso de una acción para añadir suspense.

4.5 Sistema de Registro (Logs)

Implementé un registro de las acciones del juego, donde cada acción se guarda en un archivo de texto. Esto facilita el análisis de las partidas y permite guardarlas.

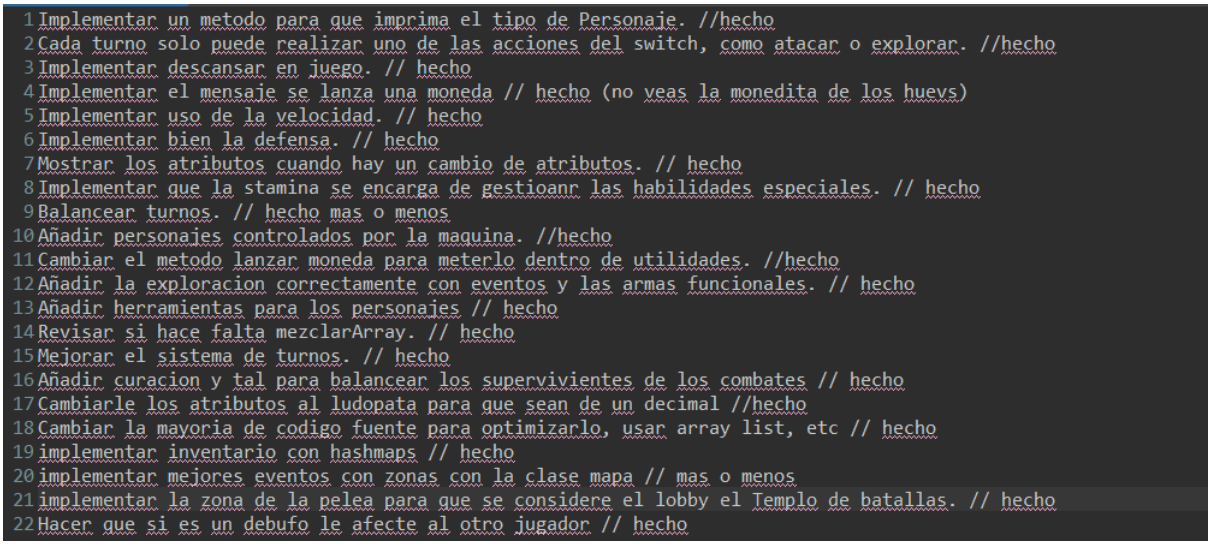
5. Dificultades y Soluciones

- **Problemas de balance**: Reajuste valores de atributos y habilidades para intentar que el juego fuese justo para todos los personajes.
- **Gestión de errores**: Implementé excepciones personalizadas para manejar errores comunes y asegurar la estabilidad del juego.
- **Código extenso y complejo**: Dividí el proyecto en paquetes y utilicé un archivo de log para registrar y organizar los cambios realizados y uso herramientas nuevas destacando el uso de hashMaps.

6. Conclusiones

Este proyecto me permitió afianzar conocimientos en Programación Orientada a Objetos, incluyendo herencia, polimorfismo y manejo de excepciones. Además, aprendí a organizar un proyecto en paquetes, a investigar por mi cuenta herramientas y sus usos como `HashMap` para implementar estructuras eficientes y a gestionar la complejidad de distintos métodos.

El enfoque del proyecto fue clave para mantener la motivación y permitir una mejora continua. Sin embargo, también implicó rehacer partes del código en varias ocasiones, subrayando la importancia de un diseño claro desde el principio.



```
Implementar un metodo para que imprima el tipo de Personaje. //hecho
Cada turno solo puede realizar uno de las acciones del switch, como atacar o explorar. //hecho
Implementar descansar en juego. // hecho
Implementar el mensaje se lanza una moneda // hecho (no veas la monedita de los huevs)
Implementar uso de la velocidad. // hecho
Implementar bien la defensa. // hecho
Mostrar los atributos cuando hay un cambio de atributos. // hecho
Implementar que la stamina se encarga de gestioanr las habilidades especiales. // hecho
Balancear turnos. // hecho mas o menos
Añadir personajes controlados por la maquina. //hecho
Cambiar el metodo lanzar moneda para meterlo dentro de utilidades. //hecho
Añadir la exploracion correctamente con eventos y las armas funcionales. // hecho
Añadir herramientas para los personajes // hecho
Revisar si hace falta mezclarArray. // hecho
Mejorar el sistema de turnos. // hecho
Añadir curacion y tal para balancear los supervivientes de los combates // hecho
Cambiarle los atributos al ludopata para que sean de un decimal //hecho
Cambiar la mayoria de codigo fuente para optimizarlo, usar array list, etc // hecho
implementar inventario con hashmaps // hecho
implementar mejores eventos con zonas con la clase mapa // mas o menos
implementar la zona de la pelea para que se considere el lobby el Templo de batallas. // hecho
Hacer que si es un debufo le afecte al otro jugador // hecho
```

Log que realice desde el principio del proyecto para ir cambiando y añadiendo cosas.

7. Trabajo Futuro

Aunque el proyecto es funcional y he añadido muchas cosas que tenía en mente, hay varias mejoras que me habría gustado implementar:

- **Interfaz gráfica**: Diseñar una interfaz que permita controlar el juego mediante botones y que incluya sprites y fondos para mejorar la experiencia del usuario.
- **Inteligencia artificial avanzada**: Mejorar la lógica de los personajes controlados por la máquina para que tomen decisiones más inteligentes.
- **Expansión del mapa**: Introducir zonas con efectos específicos en la lógica del juego.
- **Logs más completos**: Optimizar el registro de acciones y ampliar su utilidad para analizar partidas.
- **Excepciones**: Mejorar el uso de excepciones y explotar sus posibilidades

Este proyecto fue una experiencia enriquecedora que me permitió integrar conceptos teóricos y prácticos en un contexto desafiante y creativo.

Resumen del juego/guía:

El funcionamiento del juego es sencillo. Se comienza seleccionando el número de jugadores, su nombre y su clase tanto controlables como los que maneja la máquina. Después se mezclan los personajes y se establece un orden en el que comenzará el turno de cada uno de forma cíclica hasta que solo quede un jugador.
Los jugadores comienzan la partida en el Templo de Batalla y podrán en su turno atacar a otro jugador, explorar más allá del Templo o descansar en la fogata para recuperar atributos esenciales como vida o estamina, la cual se gasta al realizar habilidades especiales.

Dentro de la batalla de los personajes se comenzará lanzando una moneda y el jugador deberá elegir cara o cruz para determinar quién comienza el combate, el ganador comienza el combate y tiene varias opciones.
Si elige atacar al enemigo le quitará un número de puntos de vida que dependerá de su ataque y la defensa del rival, tiene una probabilidad de atacar una vez más que escala con la velocidad y una probabilidad de fallar el ataque, si elige usar la habilidad especial sus atributos cambiarán en función de su tipo de personaje y se gastará su estamina.
Si decide huir tendrá una probabilidad de escapar del combate por lo que el combate terminará. Al escapar el personaje recibe una pequeña curación.
Finalmente puede elegir usar un objeto de su inventario que tendrá distintos efectos dependiendo del objeto. Cada una de estas acciones finalizará el turno salvo la de usar un objeto del inventario.
Por otro lado si en el Templo de Batalla el jugador decide explorar en vez de pelear, el jugador llegará a una zona aleatoria y lanzará un dado, dependiendo del resultado del dado obtendrá un cofre con objetos y herramientas o enemigos peligrosos que tendrá que vencer para obtener recompensas.
Al obtener herramientas aumentarán los atributos del personaje que los porte dependiendo de la herramienta, si el jugador ya tiene una herramienta podrá reemplazarla. Por último, si elige descansar recuperará ciertos atributos. Los turnos se repetirán hasta que solo quede un vencedor.
Por último recuerda que en el Templo de Batalla todos los atributos que se mejoren se mantendrán a no ser que otro jugador te lance un debufo o mueras.