Proyecto 1 - Tienda RPG

1st Juan Andres Higuerey Hernandez
Escuela de Ingeniería en Computación
Instituto Tecnológico de Costa Rica
Cartago, Costa Rica
hhjuanandres@gmail.com

I. Introducción

Este proyecto consiste en una simulación de una tienda RPG.El objetivo se aplicar conceptos básicos de la programación orientada a objetos. La simulación cuenta con un personaje que tiene cinco stats: Ataque, Vida, Defensa, Mana, y Tasa Critica. Existen tres categorías de items: Arma, Consumible, y Armadura. Los items se encuentran en una tienda donde se pueden comprar y equipar. Al comprarlos se trasladan al inventario donde se pueden vender. Al equipar un item los stats del personaje se modifican segun los stats del item. La aplicación cuenta con una interfaz gráfica donde se visualiza la tienda, los stats del personaje, los items equipados, y el inventario.

II. Diseño

A. Clase Personaje

Atributos:

String nombre: nombre del personaje

stats: estadisticas del personaje, es una instancia de la clase Stats.

statsBase: estadisticas base del personaje, es una instancia de la clase Stats.

int dinero: el dinero del personaje para usar en la tienda.

inventario: inventario del personaje, es una instancia de la clase Inventario.

armaEquipada: arma que tiene equipada el personaje, es una instancia de la clase Arma.

armaduraEquipada: armadura que tiene equipada el personaje, es una instancia de la clase Armadura.

B. Clase Stats

int HP: vida del personaje.

int MP: puntos de mana.

int ATK: ataque.

int DEF: defensa.

int CRIT: tasa critica.

Identify applicable funding agency here. If none, delete this.

C. Clase Inventario

Array tamaño trece de instancias de la clase Item: aqui se guardan los items que compra el personaje. Pueden ser armas, armaduras, o consumibles.

int HP: vida del personaje.

int MP: puntos de mana.

int ATK: ataque.

int DEF: defensa.

int CRIT: tasa critica.

D. Clase Item

int precioVenta: precio del item cuando el personaje lo vende.

int precioCompra: precio del item cuando el personaje lo compra.

String nombre: nombre del item.

Subclase Arma:

int ATK, CRIT, y MP.

Subclase Armadura:

int HP, DEF, y MP.

Subclase Consumible:

int HP y MP.

III. Métodos

A. Clase Personaje

Constructor: recibe el nombre de personaje como argumento.

void equiparArma(Arma): se equipa el arma ingresada como argumento.

void equiparArmadura(Armadura): se equipa la armadura ingresada como argumento.

void equiparConsumible(Consumible): se equipa el consumible ingresado como argumento. Este modifica las estadisticas base del personaje (incremento permanente de stats).

void desequiparArma(): arma equipada se pone en nulo.

void desequiparArmadura(): armadura equipada se pone en nulo.

B. Clase Stats

Constructor: las estadísticas predeterminadas son HP = 1000; MP = 200; ATK = 120; DEF = 250; CRIT = 4;

void setHP(int): cambia el valor de vida al del argumento.

void setMP(int): cambia el valor de MP al del argumento.

void setATK(int): cambia el valor de ataque al del argumento.

void setDEF(int): cambia el valor de defensa al del argumento.

void setCRIT(int): cambia el valor de tasa critica al del argumento.

C. Clase Inventario

void ingresarItem(Item, int): ingresa un item al inventario en el indice especificado.

void borrarItem(int): borra el item en el indice solicitado.

Item getItem(int): retorna el item en el indice solicitado.

D. Clase Item

Constructor: nombre del item como argumento.

void setPrecioVenta(int): cambia el precio de venta por el indicado.

void setPrecioCompra(int): cambia el precio de compra por el indicado.

void setNombre(String): cambia el nombre del item por el indicado.

CONCLUSIONES

Para completar este proyecto fue esencial los conceptos básicos de la programación orientada a objetos y del lenguaje Java. El producto final funciona bien y tiene los requisitos solicitados, pero hay mucho que se puede mejorar. Existen muchas operaciones que se realizan manualmente en el código. Por ejemplo, en el código se crean instancias de items y se asocian individualmente a una etiqueta en el GUI. Después descubrí que las cosas se podrían automatizar utilizando arreglos y una clase dedicada a la tienda. El código también seria mas organizado ya que no habrían tantas instrucciones para crear instancias individuales. La lógica de la simulación también se encuentra en la clase controlador de JavaFX. Esta podría tener su propia clase y así el código seria mas elegante.

REFERENCIAS

Icono de espada: https://banner2.cleanpng.com/20180426/pfe/kisspng-sword-computer-icons-katana-clip-art-5ae1e95a5f4d47.34901609152475477839 04.jpg

Icono de armadura: https://encrypted-tbn0.gstatic.com/images?q=tbn

Icono de consumible: https://encrypted-tbn0.gstatic.com/images?q=tbn

Imagen de fondo: https://i.pinimg.com/originals/48/e9/51/48e951aeca534599fb39d8fdc02519b6.jpg