DOCUMENTAZIONE TECNICA MALARA DIEGO

Utilizzando il linguaggio di programmazione C#, i principi della programmazione ad oggetti (OOP) ed il software Visual Studio sono riuscito a realizzare un gioco in windows forms ispirato ad undertale.

Il sistema è progettato con un forte utilizzo della programmazione orientata agli oggetti. Sono state definite classi astratte come Personaggio e Mostro che fungono da base per i personaggi giocabili e per i nemici, rispettivamente.

Ho utilizzato anche classi singole come Items, che contiene tutti gli oggetti curativi e ne gestisce l’utilizzo, SchermataStart, che coordina le scelte prima dell’inizio dell’effettivo gioco, e GestoreGioco, che è il fulcro delle interazioni tra il gioco e l’utente, regolando i turni ed aggiornando la form.

STRUTTURA DELLE CLASSI

abstract class Mostro {

- hp: int

- immagine: PictureBox

- rnd: Random

- nome: string

+ Mostro(immagine: string)

+ GetHp(): int

+ setHp(danno: int): void

+ getSprite(): PictureBox

+ Attacca(): int

+ Schiva(): void

+ Azione(danno: int, ref p: Personaggio): void

}

' Classi Derivate da Mostro

class Asgore {

+ Asgore(immagine: string)

+ Attacca(): int

+ Cura(): void

+ NTB(): int

+ Difendi(dannoSubito: int): void

+ ColpoSicuro(): int

+ Schiva(): void

+ Azione(d: int, ref p: Personaggio): void

+ ToString(): string

}

class Froggit {

- t: Timer

+ Froggit(immagine: string)

+ Attacca(): int

+ Salta(): void

+ Gira(): void

+ Schiva(): void

+ Azione(danno: int, ref p: Personaggio): void

+ ToString(): string

}

class Papirus {

+ Papirus(immagine: string)

+ Attacca(): int

+ Cura(): void

+ Difendi(dannoSubito: int): void

+ Schiva(): void

+ Azione(d: int, ref p: Personaggio): void

+ ToString(): string

}

abstract class Personaggio {

- hp: int

- nome: string

- inventario: List<Items>

- sprite: PictureBox

- maxHp: int

+ Personaggio(immagine: string)

+ Attacca(): int

+ UsaItem(selezionato: Items): void

+ setHp(danno: int): void

+ getHp(): int

+ getSprite(): PictureBox

+ items(): List<Items>

+ Azione(indice: int, ref m: Mostro): void

}

' Classi Derivate da Personaggio

class Clover {

- colpi: int

+ Clover(i: string)

+ Attacca(): int

+ GetColpi(): int

+ Spara(): int

+ Ricarica(): void

+ Azione(indice: int, ref m: Mostro): void

+ ToString(): string

}

class Frisk {

+ Frisk(i: string)

+ Attacca(): int

+ SuperSlash(): int

+ Azione(indice: int, ref m: Mostro): void

+ ToString(): string

}

class Kunigami {

+ Kunigami(i: string)

+ Attacca(): int

+ BodyBlock(): int

+ PowerShot(): int

+ Azione(indice: int, ref m: Mostro): void

+ ToString(): string

}

Altre classi del sistema

class GestoreGioco {

- a: Mostro

- b: Personaggio

- sch: SchermataStart

- turnogioc: bool

+ GestoreGioco()

+ Gioco(p: Panel): void

+ RS(): Panel

+ RD(): List<string>

+ Attacca(i: int): void

+ HpGiocatore(): int

+ HpMostro(): int

+ RitornaInventario(): List<Items>

+ UsaItem(i: Items): void

+ getMosse(): List<string>

}

class Items {

- nome: string

- cura: int

+ Items(n: string, c: int)

+ Nome: string {get; set;}

+ Cura: int {get; set;}

+ ToString(): string

}

class SchermataStart {

- p: Panel

- \_comboBox: ComboBox

- \_comboBox2: ComboBox

- \_button1: Button

- lbTitolo: Label

- lbTitolo2: Label

- lbTitolo3: Label

- c: List<string>

+ SchermataStart()

+ getSchermata(): Panel

+ getDati(): List<string>

}

class Form1 {

- g: GestoreGioco

- timer: Timer

+ Form1()

+ Form1\_Load(sender: object, e: EventArgs): void

+ c(sender: object, e: EventArgs): void

+ ProvaPerGioco(): void

+ AggiornaInv(): void

+ AggiornaSkill(): void

+ button1\_Click(sender: object, e: EventArgs): void

+ button2\_Click(sender: object, e: EventArgs): void

}

GIOCO

Immagine che contiene testo, schermata, giallo, Carattere

Il contenuto generato dall'IA potrebbe non essere corretto.

Immagine che contiene testo, cartone animato, schermata

Il contenuto generato dall'IA potrebbe non essere corretto.

Immagine che contiene testo, cartone animato, schermata, software

Il contenuto generato dall'IA potrebbe non essere corretto.

RAPPORTI TRA LE CLASSI

Immagine che contiene diagramma, linea, testo

Il contenuto generato dall'IA potrebbe non essere corretto.

Immagine che contiene testo, schermata, Carattere, linea

Il contenuto generato dall'IA potrebbe non essere corretto.