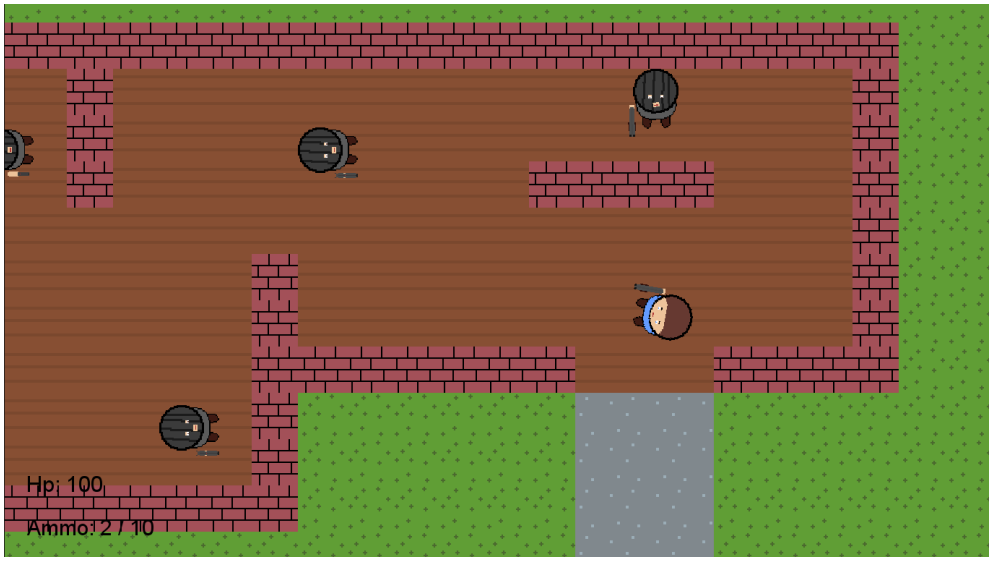
Unpleasant Takeover

Imbrea Daniel

Faculty of Automatic Control and Computer Engineering, Iasi

daniel.imbrea2@student.tuiasi.ro

2020 - 2024



- **Gameplay:** *The goal of the game is to eliminate all the criminals, with the tools available at your disposal. Use the environment to your advantage and time your offensive during their reloads. There are 3 guns you can use: a pistol with moderate firerate, damage and ammo, a shotgun with slow firerate but impressive damage, and an smg with a high firerate and ammunition. Around the map are scattered ammo bags and medkits to help you in your fight, replenishing your ammunition and health. If you end up dying in battle, you will be respawned at the start of the level with full health and ammo, but all enemies will also come back.*

- **Plot:** *A criminal organization known as "Evil inc." has taken over the city of Notingburg. The police was taken by surprise and you are the only officer available in the area. It is your duty to restore order to the city.*

- **Characters:**
  - ***Jim Bob the officer****, a respected police officer of the city of Notingburg.*
  - ***The Evil inc. criminals,*** *the gang that has taken over the city*

- **Mechanics (turns, game points, user interaction, keys):**
  - W/A/S/D to move the character;
  - 1/2/3 to change weapon;
  - R to reload the weapon;
  - E to use medkits and ammo bags;
  - ENTER to go to the next level;
  - Left click to shoot;
  - PLAY button to start the game;
  - TUTORIAL button to open the tutorial tab;
  - EXIT button to close the game;
  - A button that leads you to my website
  - ESC to pause the game while playing

La evaluare se vor avea in vedere urmatoarele:

#	Criteriu	Realizat
1	Abstractizare	10
2	Încapsulare	10
3	Moștenire (ierarhie de grad 3 minim)	8
4	Polimorfism	10
5	Interfețe (clase abstracte)	9
6	Gestionarea erorilor (excepții)	9
7	Salvarea sau încărcarea configurației jocului (Lucrul cu fișiere)	10
8	Număr de niveluri cu dificultate graduală (minim 3)	10

1.Abstractizare: are rolul de a separa metode/campuri private de cele publice, accesibile din orice parte a programului. In acest proiect abstractizarea este folosita in majoritatea claselor. Exemple:

```
class UILabel : public Component
{

private:
    TTF_Font* font;
    SDL_Rect position;
    std::string labelText;
    std::string labelFont;
    SDL_Color textColour;
    SDL_Texture* labelTexture;

public:

    UILabel(int xpos, int ypos, std::string text, int size, SDL_Color& colour) :
        labelText(text), textColour(colour)
    {
        font = TTF_OpenFont("assets/arial.ttf", size);
        position.x = xpos;
        position.y = ypos;

        SetLabelText(labelText);

    }
```

2.Incapsulare: reprezinta utilizarea functiilor publice pentru editarea campurilor private. Aici sunt cateva exemple din proiect:

```
void setPos(int x, int y)
{
    position.x = x;
    position.y = y;
}
```

```
void setFrame(int f)
{
    srcRect.x = srcRect.w * f;
}
```

3.Mostenire: este folosita in mare parte in relatie cu clasa ‘component’. Exemple:

```
class SpriteComponent : public Component
class TileComponent : public Component
class ProjectileComponent : public Component
```

4.Polimorfism: Este folosit in situatii in care o functie trebuie sa functioneze cu diferiti parametri. Exemple:

```
void setPos(int x, int y)
{
    this->position.x = x;
    this->position.y = y;
}

void setPos(int x, int y, int rot)
{
    this->position.x = x;
    this->position.y = y;
    this->rot = rot;
}
```

5.Clase abstracte:

```
class Component
{
public:
    Entity* entity;

    virtual void init() {}
    virtual void update() {}
    virtual void draw() {}

    virtual ~Component() {}
};
```

6.Gestionarea erorilor: Sunt verificate erori in initializarea librariilor folosite cat si la crearea ferestrei si a renderer-ului. Exemple:

```
try
{
    if (window == NULL)
        throw "Error creating window.";
    else
    {
        std::cout << "Window created\n";
    }
}
catch(const std::exception& e)
{
    std::cout << e.what();
}
```

```
try
{
    if (renderer == NULL)
        throw "Error creating renderer.";
    else
    {
        SDL_SetRenderDrawColor(renderer, 0x00, 0x00, 0x00, 0x00);
        std::cout << "Renderer created\n";
    }
}
catch (const std::exception& e)
{
    std::cout << e.what();
}
```

7.Lucrul cu fisiere: este folosit la incarcarea nivelelor si a inamicilor:

```
map->LoadMapChunk("assets/fullmap.txt", 32, 0, 65, 32);
```

```
map->LoadMapChunk("assets/fullmap.txt", 0, 32, 65, 65);
```

```
map->SpawnEnemies();
```

8.Niveluri: Jocul contine 3 nivele cu dificultate graduala reprezentata prin numarul de inamici din fiecare nivel.

Cod GitHub: https://github.com/danelthedev/Proiect-OOP---Unpleasant-Takeover