Sprawozdanie

Programowanie Komputerów

Wykonał:

Bartłomiej Bielak

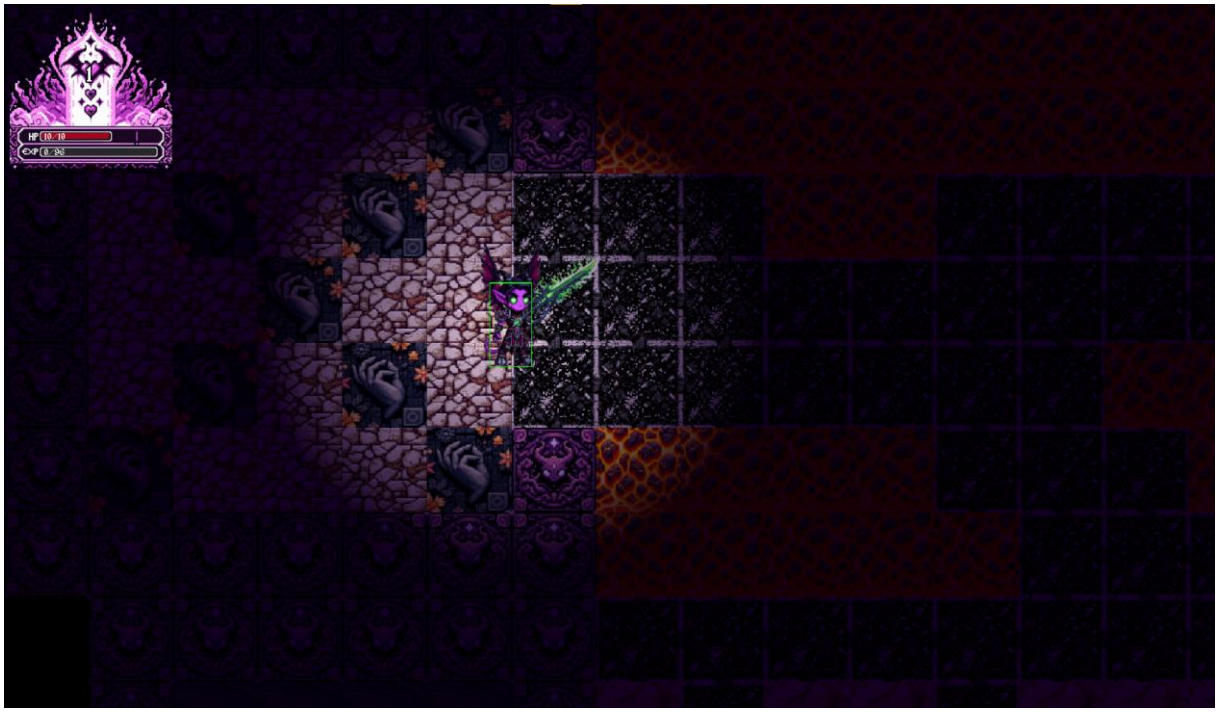
Tytuł projektu:

SFML_RPG_GAME

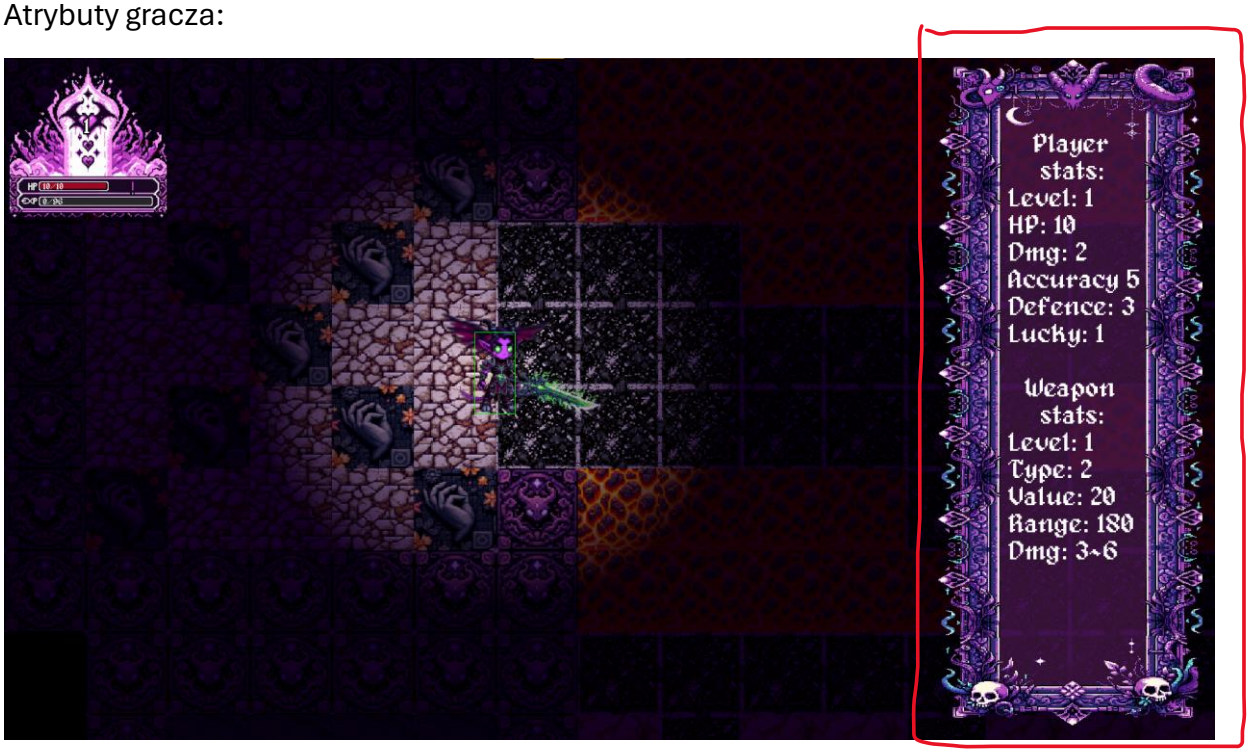
06.06.2024

Menu główne:

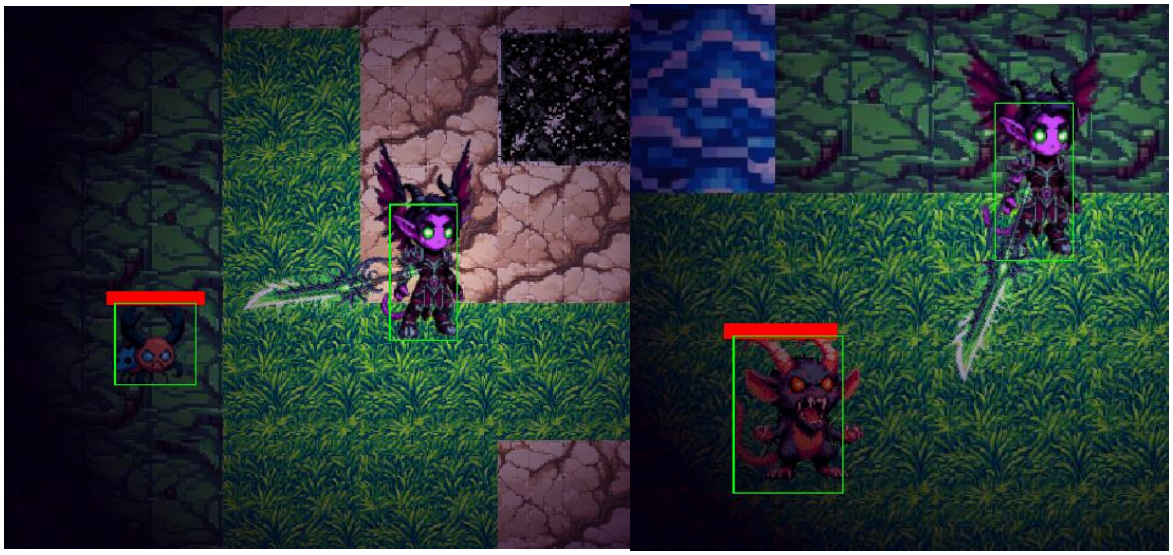
Rozgrywka:



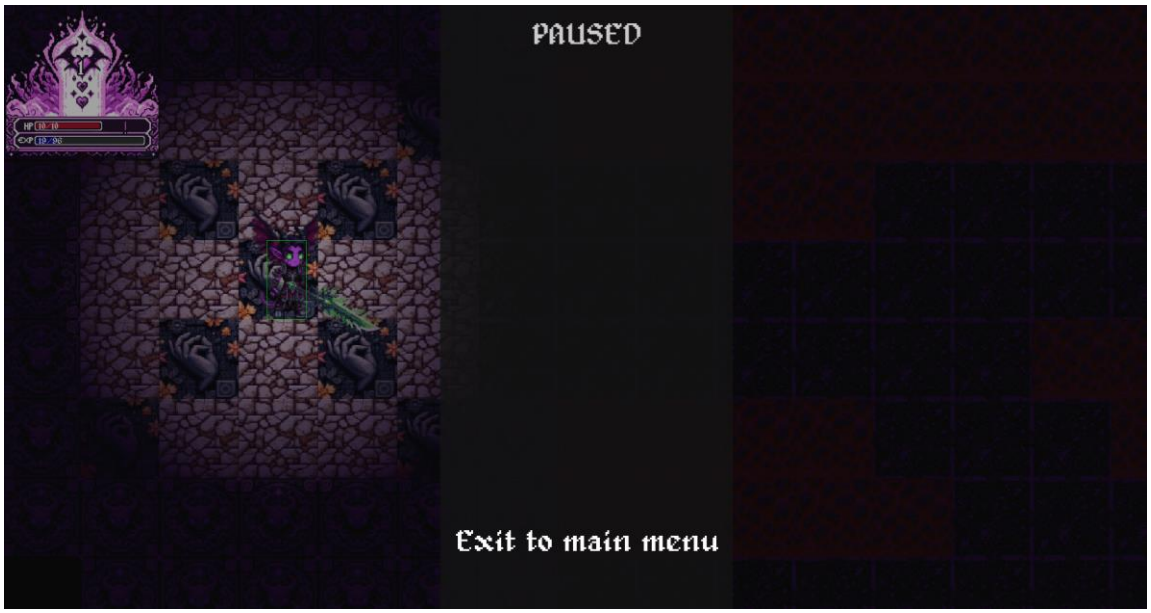
Atrybuty gracza:



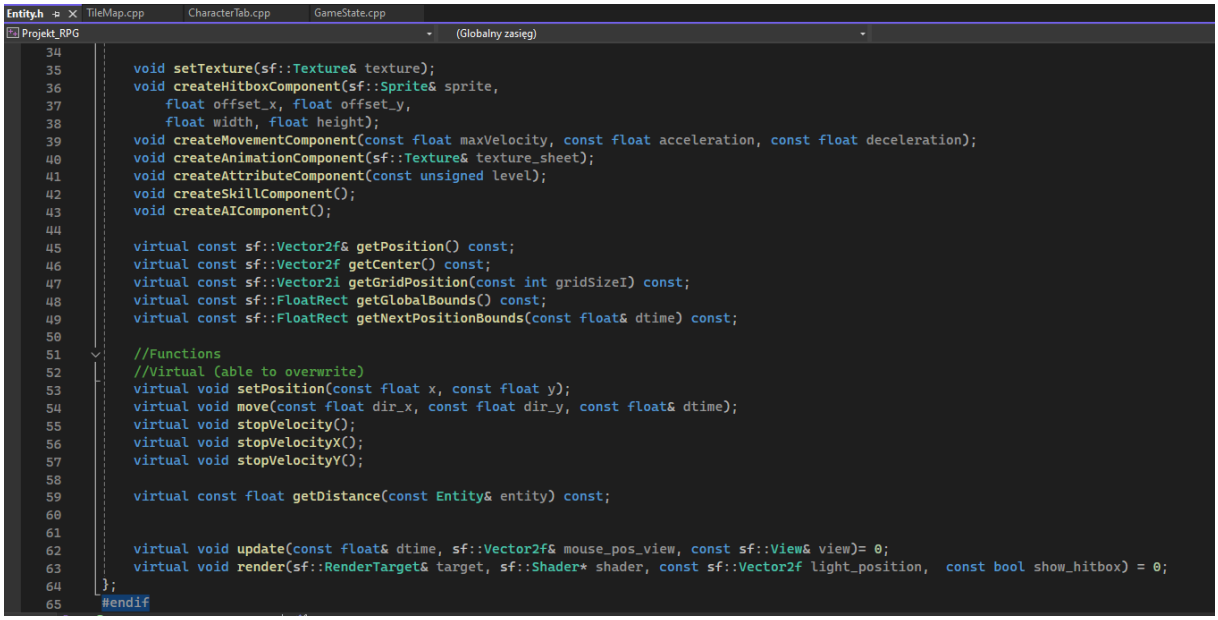
Przeciwnicy:



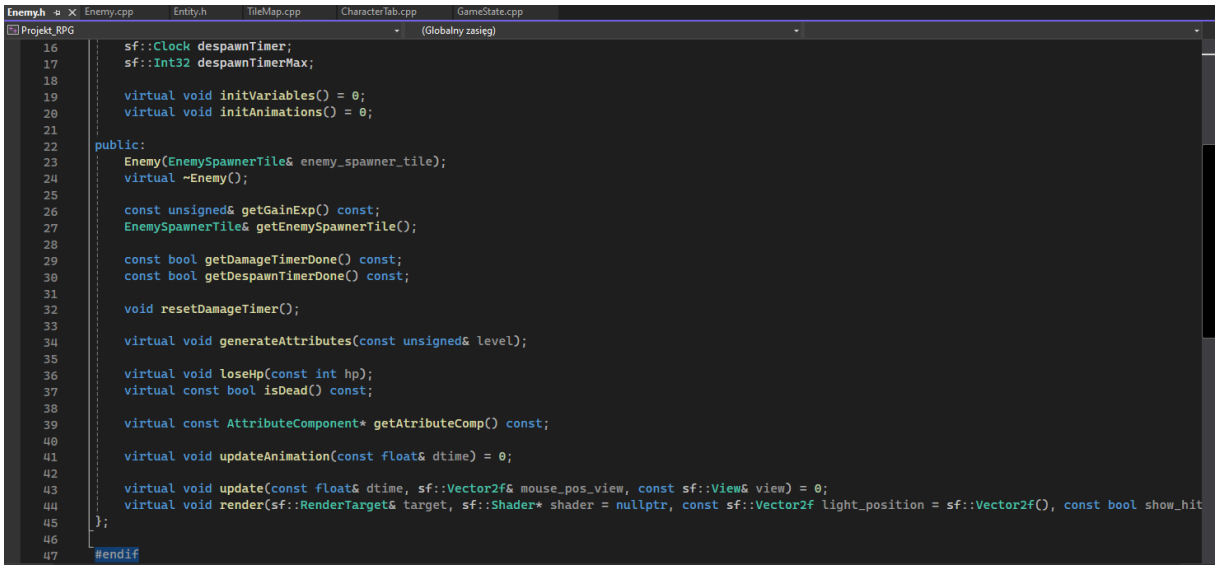
Menu pauzy:



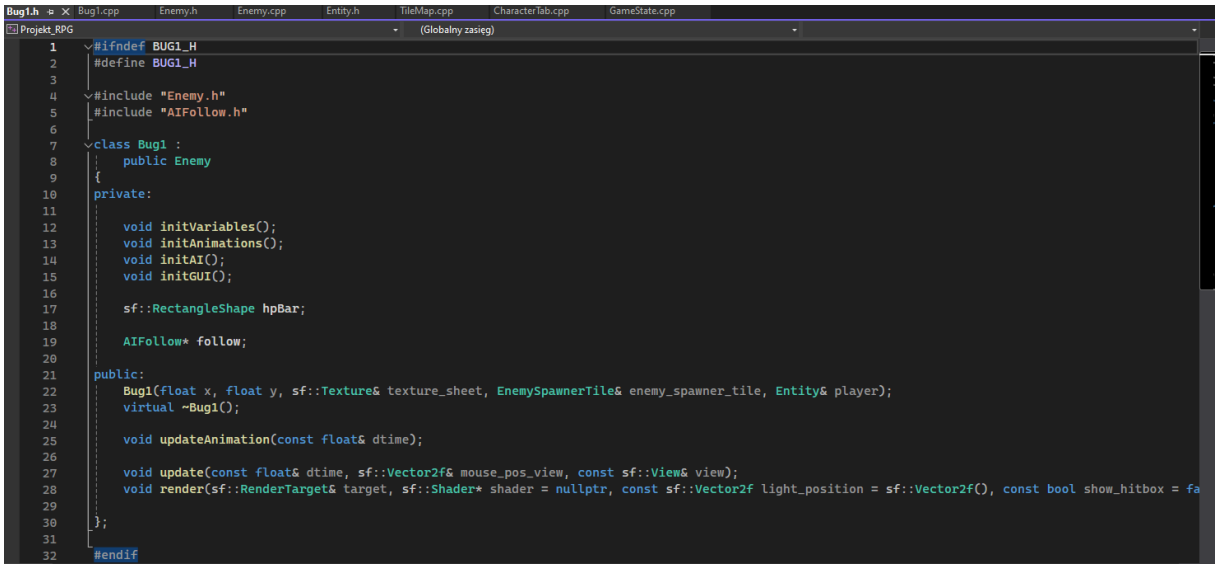
Implementacja klasy wirtualnej:



```
Entity.h | TileMap.cpp | CharacterTab.cpp | GameState.cpp
Projekt_RPG | (Globalny zasięg)
34
35      void setTexture(sf::Texture& texture);
36      void createHitboxComponent(sf::Sprite& sprite,
37          float offset_x, float offset_y,
38          float width, float height);
39      void createMovementComponent(const float maxVelocity, const float acceleration, const float deceleration);
40      void createAnimationComponent(sf::Texture& texture_sheet);
41      void createAttributeComponent(const unsigned level);
42      void createSkillComponent();
43      void createAIComponent();
44
45      virtual const sf::Vector2f& getPosition() const;
46      virtual const sf::Vector2f getCenter() const;
47      virtual const sf::Vector2i getGridPosition(const int gridSizeI) const;
48      virtual const sf::FloatRect getGlobalBounds() const;
49      virtual const sf::FloatRect getNextPositionBounds(const float& dtime) const;
50
51      //Functions
52      //Virtual (able to overwrite)
53      virtual void setPosition(const float x, const float y);
54      virtual void move(const float dir_x, const float dir_y, const float& dtime);
55      virtual void stopVelocity();
56      virtual void stopVelocityX();
57      virtual void stopVelocityY();
58
59      virtual const float getDistance(const Entity& entity) const;
60
61
62      virtual void update(const float& dtime, sf::Vector2f& mouse_pos_view, const sf::View& view)= 0;
63      virtual void render(sf::RenderTarget& target, sf::Shader* shader, const sf::Vector2f light_position,  const bool show_hitbox) = 0;
64  };
65  #endif
```



```
Enemy.h | Enemy.cpp | Entity.h | TileMap.cpp | CharacterTab.cpp | GameState.cpp
Projekt_RPG | (Globalny zasięg)
16      sf::Clock despawnTimer;
17      sf::Int32 despawnTimerMax;
18
19      virtual void initVariables() = 0;
20      virtual void initAnimations() = 0;
21
22  public:
23      Enemy(EnemySpawnerTile& enemy_spawner_tile);
24      virtual ~Enemy();
25
26      const unsigned& getGainExp() const;
27      EnemySpawnerTile& getEnemySpawnerTile();
28
29      const bool getDamageTimerDone() const;
30      const bool getDespawnTimerDone() const;
31
32      void resetDamageTimer();
33
34      virtual void generateAttributes(const unsigned& level);
35
36      virtual void loseHp(const int hp);
37      virtual const bool isDead() const;
38
39      virtual const AttributeComponent* getAtributeComp() const;
40
41      virtual void updateAnimation(const float& dtime) = 0;
42
43      virtual void update(const float& dtime, sf::Vector2f& mouse_pos_view, const sf::View& view) = 0;
44      virtual void render(sf::RenderTarget& target, sf::Shader* shader = nullptr, const sf::Vector2f light_position = sf::Vector2f(), const bool show_hit
45  };
46
47  #endif
```



```
Bug1.h | Bug1.cpp | Enemy.h | Enemy.cpp | Entity.h | TileMap.cpp | CharacterTab.cpp | GameState.cpp
Projekt_RPG | (Globalny zasięg)
1   #ifndef BUG1_H
2   #define BUG1_H
3
4   #include "Enemy.h"
5   #include "AIFollow.h"
6
7   class Bug1 :
8       public Enemy
9   {
10  private:
11
12      void initVariables();
13      void initAnimations();
14      void initAI();
15      void initGUI();
16
17      sf::RectangleShape hpBar;
18
19      AIFollow* follow;
20
21  public:
22      Bug1(float x, float y, sf::Texture& texture_sheet, EnemySpawnerTile& enemy_spawner_tile, Entity& player);
23      virtual ~Bug1();
24
25      void updateAnimation(const float& dtime);
26
27      void update(const float& dtime, sf::Vector2f& mouse_pos_view, const sf::View& view);
28      void render(sf::RenderTarget& target, sf::Shader* shader = nullptr, const sf::Vector2f light_position = sf::Vector2f(), const bool show_hitbox = fa
29
30  };
31
32  #endif
```