**Project Report: Pygame 2D Action Game**

**Ignat Tyo, Agybaev Aidyn, Alibek** [**Zhaileubai**](https://moodle.astanait.edu.kz/user/view.php?id=8111&course=5712)

**IT-2206**

**1. Introduction**

This project is a 2D action game developed using Python and the Pygame library. The game challenges the player to navigate a level, avoid obstacles, collect items, and defeat enemies. Key mechanics include character movement with directional animations, shooting projectiles toward the mouse cursor, collecting powerups and coins that affect gameplay, and dynamically adjusting difficulty based on player selection.

**2. Main Mechanics**

**2.1 Player Movement and Animation**

* **Movement:**  
  The player controls Capy using the keyboard (W, A, S, D). The character’s movement is collision–aware; the intended movement is maintained so that enemy AI can react even if the actual position is constrained by walls or obstacles.
* **Animation:**  
  A walking animation is implemented for the player. The character has separate animation frames for two directions (left, right). For each direction, multiple PNG frames are loaded and scaled. An animation timer cycles through frames every 110 milliseconds when the player is in motion.
* **Shooting:**  
  Capy can shoot projectiles toward the mouse cursor. The shooting mechanic supports both single and multiple projectile bursts—with a slight angular spread for added challenge. The shooting animation is synchronized with directional input and incorporates audio cues.

**2.2 Enemy Behavior and Animation**

* **Spawning and Movement:**  
  Enemies spawn outside the visible level boundaries and move continuously toward the player’s current (or intended) position. They ignore collisions with walls and obstacles, ensuring persistent pursuit regardless of the environment.
* **Animation:**  
  Enemies have animation system focused on horizontal movement. Two sets of four frames are loaded and scaled. An animation timer updates the enemy sprite every 200 milliseconds based on its movement direction.

**2.3 Items, Coins, and Powerups**

* **Items:**  
  Items appear at regular intervals on the level. They include healing item, temporary speed boost and freezer. When collected, these items affect the player immediately (e.g., restoring health, temporarily increasing movement speed or decreasing enemies moevement speed) and trigger a visual effect—a pulsating ring around the player. Freezer also activates the blue overlay.

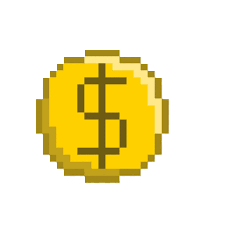
Изображение выглядит как пиксель, снимок экрана

Автоматически созданное описание Изображение выглядит как прямоугольный, шаблон, снимок экрана, Красочность

Автоматически созданное описаниеИзображение выглядит как пиксель, Красочность

Контент, сгенерированный ИИ, может содержать ошибки.

* **Coins:**  
  Coins spawn on the field periodically and add score when collected. They also trigger a short visual effect.



* **Powerups:**  
  When the player reaches a score threshold, the game pauses and presents two randomly selected powerup cards. Each card offers a permanent upgrade and is presented with a custom image, description, and a “Select” button for user interaction. Types of powerups: 1) max hp increment; 2) moevement speed increment; 3) bulet speed increment; 4) temporary shield; 5) new weapon – shotgun; 6) new weapon – piercing gun; 7) +1 bullet per shoot. Every powerup has different chance to appear.

**2.4 Difficulty Levels and Level Design**

* **Difficulty Modes:**  
  The game supports two difficulty modes—easy and hard. In hard mode, enemies spawn more frequently, move faster, and items spawn less often. The game selects a different level file (e.g., level2.txt for hard mode) and adjusts timers and parameters accordingly.
* **Level Design:**  
  Levels are defined using text files that represent a grid layout. Specific characters denote walls, floors, obstacles, and the player’s spawn point. The level loader parses these files and renders tiles accordingly.

**2.5 Narrative and Menu**

* **Dialog Scene (Narrative Introduction):**  
  Before gameplay begins, a dialog scene is presented. In this scene:
  + A **dialog box** appears occupying the bottom quarter of the screen.
  + A **character sprite** is displayed above the dialog box at the right-hand side.
  + The dialog text is animated, appearing letter-by-letter to enhance storytelling.
  + Once all messages have been shown, the entire dialog scene slides downward off the screen, providing a seamless transition into the game.
  + The background used in this scene is a frozen snapshot of the level with the camera centered on Capy’s spawn point, ensuring continuity.
* **User Interface Enhancements (Menus):**  
  The menus have been designed for a more polished and consistent look across the game:
  + **Main Menu:**
    - Utilizes a background image that sets the visual tone.
    - Displays a “GAME NAME” image at the top of the screen.
    - All menu options are contained within one semi-transparent black panel that spans a substantial portion of the screen, ensuring clear visibility of options.
  + **Pause Menu:**
    - Features the same background image and integrated panel style as the main menu, providing consistency even when the game is paused.
  + **Options Menu:**
    - Adopts a similar layout, with the volume adjustment controls displayed inside a centered, semi-transparent panel over the background image.

**3. Main Scripts Overview**

**3.1 main.py**

* **Responsibilities:**  
  Acts as the entry point for the game. It initializes Pygame, creates the display, manages game states (main menu, game, pause, and game over), and adjusts difficulty parameters. It also controls the main game loop, spawns enemies, items, and coins, and handles collisions and scoring. Main loop includes a pre-game dialog scene that smoothly transitions into gameplay. This is accomplished by capturing a snapshot of the level (with the camera centered on Capy’s spawn) and overlaying the animated dialog.
* **Difficulty Handling:**  
  The script checks the difficulty selected in the main menu and loads either level1.txt or level2.txt accordingly, adjusting enemy spawn intervals, enemy speed, and item spawn intervals.

**3.2 game/player.py**

* **Responsibilities:**  
  Contains the Player class, which handles movement, collision detection, shooting, and playing animations. The class loads and scales different animation frames for each movement direction. It also implements methods for temporary and permanent powerups and displays visual collection effects.

**3.3 game/enemy.py**

* **Responsibilities:**  
  Contains the Enemy class. Enemies spawn outside the level and move toward the player using a simple left/right animation cycle. They ignore collisions with walls and obstacles and have methods to take damage and update their animations.

**3.4 game/level.py & game/camera.py**

* **Level Module:**  
  Loads a text-based level layout and creates game tiles accordingly (walls, floors, obstacles, etc.). It also sets the player’s spawn point.
* **Camera Module:**  
  Implements a camera that follows the player while clamping the view to the level’s boundaries.

**3.5 game/ui.py, game/item.py, game/coin.py, game/powerup.py**

* **UI Module:**  
  Displays real-time information such as the player’s health, score, movement speed, bullet speed, and the game goal.
* **Item and Coin Modules:**  
  Manage the spawning and collection of items (which provide temporary effects) and coins (which add to the score).
* **Powerup Module:**  
  Implements a powerup card selection system that pauses the game when a score threshold is reached, letting the player choose a permanent upgrade.

**3.6 game/sounds.py**

* **Responsibilities:**  
  Centralizes sound management, including background music and sound effects for various game actions (e.g., collecting items, collisions, and enemy kills).

**3.7 game/menu.py & game/dialog.py**

* **Responsibilities:**  
  Handles the different menu designs and the dialog scene before the main gameplay loop.

**4. Game Screenshots**



Изображение выглядит как шляпа, текст, мультфильм, Человеческое лицо

Контент, сгенерированный ИИ, может содержать ошибки.

Изображение выглядит как текст, снимок экрана, Мультимедийное программное обеспечение, программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.