Project plan

Angry birds C++ project

Scope of the project

The aim of the project is to make a simple angry birds – type of game with multiple levels of gameplay. Players mouse input will be used to control the slingshot that shoots the birds and to activate the special abilities of the birds. The game will have different kinds of birds, with different abilities. The levels are constructed from various destroyable and non-destroyable blocks. The game will feature 2D physics using Box2D-library, and a simple graphical user interface implemented with SFML/QT - libraries. Game levels will be saved into different files and loaded on demand.

Main idea of the game is to shoot different kinds of birds towards enemy fortresses, destroy the fortresses and eliminate the enemies. Destroying all the enemies will grant a victory. The more destruction you are able to, the more points you will get. You also get more points the less birds you use to eliminate all the enemies.

High-level structure of the software

Main classes:

- main – project root. Initializes Physics and Game, run game loop that updates Physics and the Game.
- Game – loads/creates ProgressData, loads LevelData based on progress, creates and updates Level, manages game states (shows different UI).
- Physics – manages Box2d world and bodies.
- Resources - manages game assets loading and access (textures, sounds); save/load game data to files (LevelData, ProgressData).
- LevelData – representation of the level structure: entities properties and positions. Can be loaded from file.
- ProgressData – holds the progress of the player (e.g. current level and score). Can be saved and loaded from file.
- Level – creates and updates all entities on current level, handles scoring logic and win/lose conditions.
- Entities – Abstract class, represent elements that are on the level (brids, Enemies...), used to encapsulate the common parameters.
    - Bird – implement generic bird object each specific bird type will extend this class so they can have special abilities (speed, action)
    - Enemy – enemy object that specify the type of enemy, his position, points
    - Slingshot – object used to represent the slingshot (angle, power).
    - Obstacle – object to specify position, type (breakable?) of an obstacle.
- UiManager – Handle windows, popups, inputs, using SFML library.
    - UIPanel – base abstraction for different kinds of UI panels
        - HUD – UI inside the level (shows score, level number, etc).

        - Popup – base class for a popup.
    - UIElement – base abstraction for different UI elements (e.g. Button).

External libraries

Physics: Box2D

Graphics & UI: SFML

Sound: SFML

Work division

The main game can be divided into the following main workloads:

- Entity: Quang
- UI: Maksim
- Game logic: Teo
- Level: Auvo
- Game physics: Allan

Specific tasks:

1. **Design & Concept**:
    - Sketch levels and define game mechanics.
    - Design upgrade system.
    - Design bird types and their unique abilities.
2. **Setup & Initialization**:
    - Set up the game window and rendering.
    - Makefile compilation
    - Initialize Box2D for physics.
3. **Graphics**:
    - Draw backgrounds, birds, slingshot, obstacles, and enemies.
    - Implement basic animations.
4. **Game Mechanics**:
    - Implement the slingshot mechanic.
    - Implement bird flight and collision.
    - Add unique bird abilities.

- Implement scoring and level progression.

5. **Upgrades & Bird Selection**:
    - Design and implement a bird selection screen.
    - Design and implement an upgrade purchasing and application system.
6. **Levels**:
    - Create multiple level designs.
    - Set difficulty progression.
7. **Audio**:
    - Add background music.
    - Add sound effects for slingshot, bird flights, collisions, etc.
8. **GUI & Menus**:
    - Design main menu, pause menu, and level selection screens.
    - Display scores, bird availability, and other game data.
9. **Testing & Debugging**:
    - Test all mechanics, upgrades, and bird abilities.
    - Fix any bugs or inconsistencies.

Schedule

Week 1 & 2 – game design, basic features: basic UI, setup physics and basic rendering

Week 2 to 4 – core game mechanics implementation i.e., make the game work.

Week 4 & 5 – audio, levels design, extra features implementation

Week 6 – play testing, debugging, refinement

Milestones: 5.11 - design is ready; 26.11 - game is playable

Angry birds C++ class diagram