Game Development

Where you build an Arkanoid game.

Now that you are familiar with the game and with the graphics library, start by thinking about the high level design of what you will build.

It does not need to be a perfect or complete design, but something that enables you to start working. Remember that the design is never done and is never perfect. Nevertheless, your design should be according to OOP and SOLID principles.

Guidelines

The game development is divided into 6 milestones.

1. Each milestone has a set of features.
2. Don't add features from advanced milestones.
3. If you have ideas for cool features, add them after you finish the last milestone.
4. Each milestone has an estimated time allocated for development and test.
   You should make every effort to finish in the allotted time.
5. Some features are marked as *Extra*.
   These features are for advanced students and should only be developed if you finish the milestone before the allocated time.

Git Setup:

Work shall be done starting with a new branch named: *`arkanoid`*

- Start by switching to projects branch, execute the command: `git checkout projects`
- Create a new branch named *`arkanoid`* and switch to it: `git checkout -b arkanoid`
- Each milestone of the game must be in its own branch.

Design

But design is overrated.
Remember: 6 hours of debugging can save you 30 minutes of design.

Work on each milestone should always start with a design effort. Invest about up to 60 minutes in this Document your design with a:

1. High level design document
2. Detailed level design document
3. You can use any drawing/diagramming tool of your choice. Hand drawing is OK if its clear and neat.
4. Commit and push your design files before starting to code.

Milestone 1

Where you build a rudimentary game.

Inside a new directory named: *prj-01-arkanoid* create a new project for the game. Make sure you have subdirectories for *hpp*, *cpp* files. Feel free to make more subdirectories as you see suitable.

The first version shall have the following features:

1. Player starts with 3 lives.
2. Player controls the game paddle movements with arrow keys and space bar to shoot the ball.
3. Game has one level. The level has 3 rows of 5 bricks each. Each row is of different color.
4. The bricks will disappear if hit by the ball and player will get 40 points.
5. Keep a score updated and display it during the game.
6. If the ball falls off the bottom of the screen, one life is lost.
7. When all lives are lost, game ends and exits.

Branch: `dev-arka-1`
Files: use `inc`, `src` and `tests` directories as usual.
Time estimated: 5-8 hours

Milestone 2

Where you enhance the UI flow of the game. Add the following features:

1. Add an opening screen with a graphic and help text.
   a. A *play* button on this screen will take the player to the first level.
   b. A *quit* button on this screen will end the program.
2. While playing in a level, pressing the *ESC* key will return the player to the opening screen.
3. When the player is disqualified – all lives are lost – exit the level and return the player to the opening screen.
4. If player clear all the bricks from the level, move to the Top 10 high scores screen.
   a. Top 10 screen displays the top 10 player names and their scores.
   b. Two players with same score should be listed according to the time taken to finish.
   c. If the player score qualifies him to be in the top 10
      i. ask for player name and add it to the list
   d. The list should be saved in a file named ***top10.dat***
   e. The file should be read on game start or on entering the top 10 screen.
      i. It should be automatically created if not present.
   f. Top 10 screen will exit when user hits any key. Move back to opening screen.
   g. Add the filename ***top10.dat*** to .gitignore

Branch: `dev-arka-2`
Time estimated: 5-7 hours