Design Document

CSCE 361 - Fall 2017

Brick Breaker

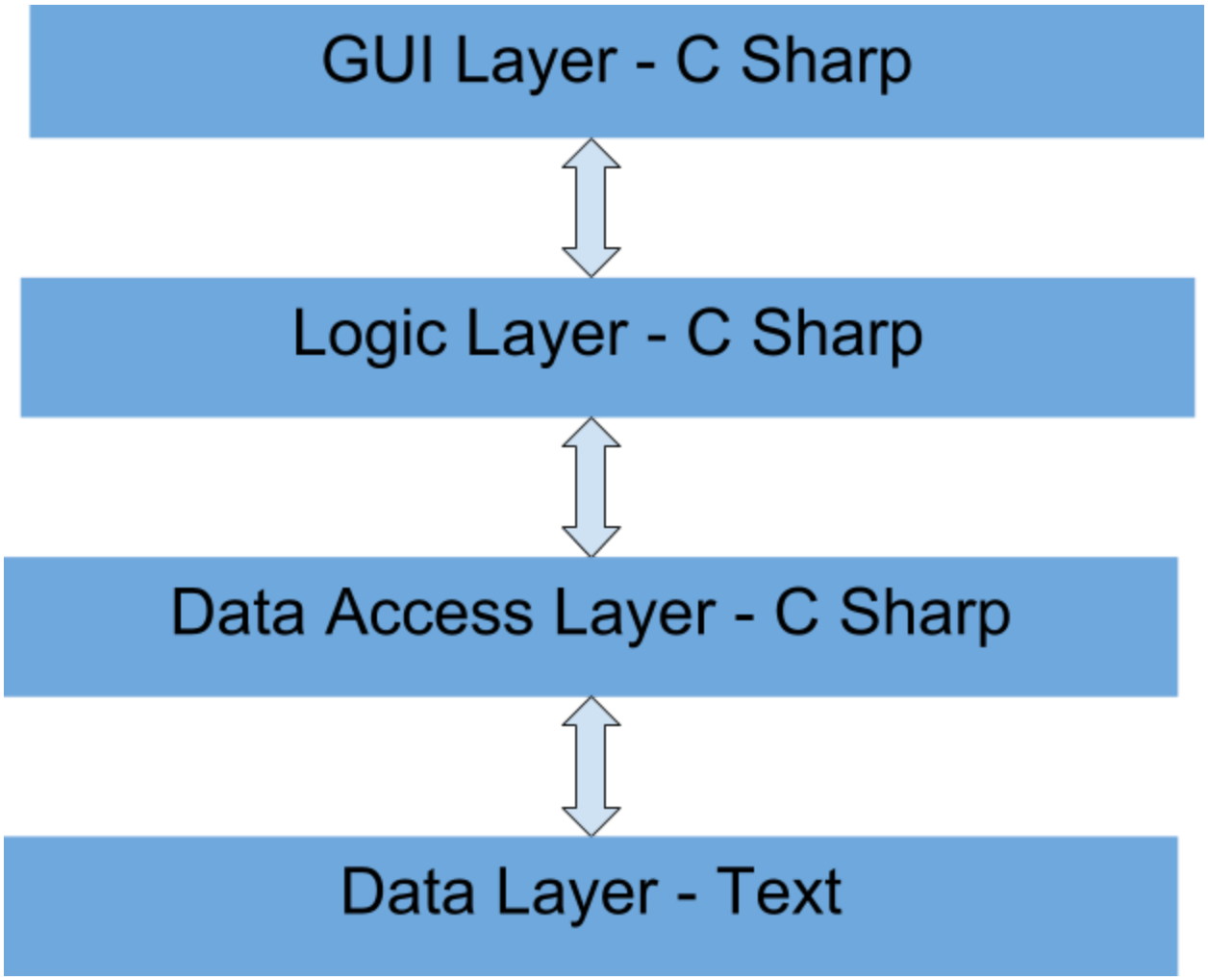
Group 3
(Jasun Hitchcock-Sivak, Skyler Davis, Nick Wieskamp, Jacob Davidson)

1. Introduction

The purpose of this design document is to demonstrate the high level architecture and entity relations, for the Brick Breaker management system. The document will have entity relation diagrams and will show the architecture of the game at a high level. The document will be able to provide a clear difference between on parts of the system as well as the relationships between the tables.

2. Architecture

2.1 introduction



The high level architectural design of the game will be a layered model. The lowest layer is the data. This layer will contain the flat files that will be read from to get

information such as “High Score”. The data access layer will use C Sharp to read information from the flat files. Next the logic layer, also using C Sharp, will use the data, along with other logic to create the games functionality. Finally, C Sharp will once again be used to create a GUI layer to display the game to the screen.

2.2 Modules

2.2.1 Data layer

The data layer will be responsible for holding data that needs to be remembered for different instances of the application. Namely, the data layer will contain a text file with “High Scores”. Text files will be used because a relatively small amount of information will need to be remembered and there is no need for data to be shared between different installations of the application.

2.2.2 Data Access Layer

This module will be responsible for reading data from the text files and making that data available for the logic layer so that it can be used in the application. This layer will consist of a simple C Sharp code that can read and parse the provided information.

2.2.3 Logic layer

This module will be responsible for performing all of the necessary logic to the c# objects used in brick breaker. The logic layer will provide all the information to the GUI layer including all the users high scores the type of enhancement they have and the lives they have. This layer will need to communicate any changes in the data as a result of the GUI layer to the Data access layer so we are able to make sure all changes will be stored permanently.

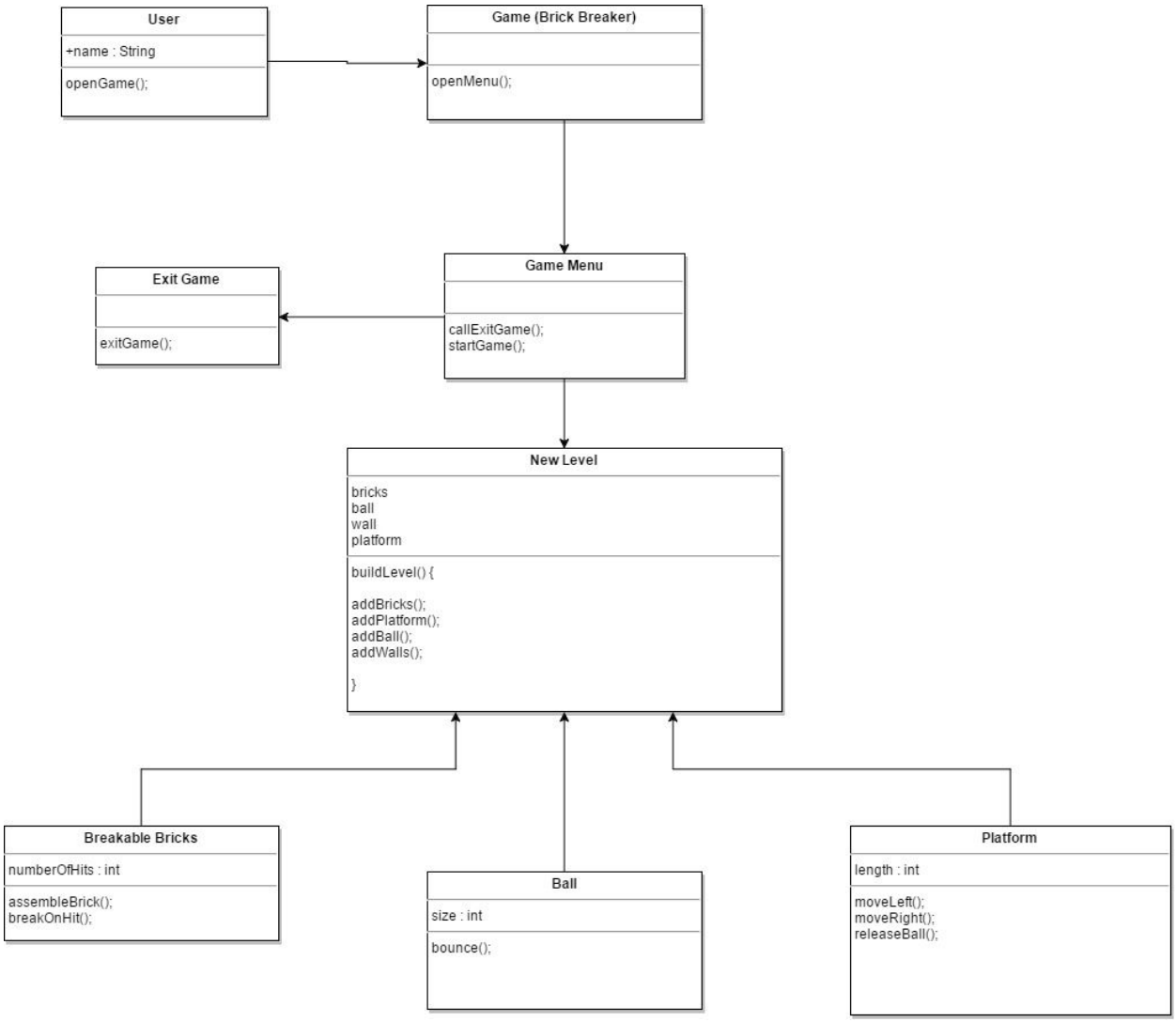
2.2.4 GUI Layer

This module will be responsible for displaying the information and functionality created by the layers below it. This layer will begin with a main menu, where the user will have several options, each of which will trigger actions that are created inside of the logic layer. The most prominent of these options will be to begin a new game, at which point the screen will refresh and display the first level of Brick Breaker. The GUI layer will be created by C Sharp.

3. Class Diagrams

3.1 Data Table Classes

3.1.1 Schema



3.1.2 Schema Information

The data in the classes will be organized in the following tables and columns:

User: The user class will hold the data for the single user to open and close the game application. It will also hold the name of the user if we decide to implement that feature.

Game: Table contains all the data that the user is able to interact with to process through the game application. The main entity that the user will interact with.

Game Menu: This will contain the data of the startup menu. At its simplest form, it will only contain start game or exit game options. Later with implementation, it may also contain the options to see a list of high scores.

Exit Game: Data processed through this table will close out the application.

New Level: For every new level the table will hold data for the bricks, the ball, and the player/platform. It will also contain the data for the organization and arrangement of the bricks for each new level provided.

Bricks: holds the data for how the bricks are put together on each level. It will also provide data whether a brick is hit with the ball and breaks. For example, we may incorporate bricks that take multiple hits to fully break.

Ball: This table will keep track of when the ball bounces off of the platform/walls/bricks, and it will ensure that it bounces when it is supposed to. An example of when it may not would be when the fireball powerup is active and the bricks do not affect the ball.

Platform: This table will represent the movement of the player as they move from left to right and the release of the ball. Each of these functions will be mapped to keyboard keys to allow the player to have control.

3.2 class information

All classes will be implemented in C# throughout the data access layer and used throughout the game application. For example the logic layer will pass information to the GUI layer so that it will be displayed to the user such as the ball hitting a brick and the brick breaking or the ball bouncing off the platform.

3.3 GUI Layer

The top layer will be the main menu that will just be one page. It will give the user the option to start a game.

The next layer will have the game and the pause menu. The game will keep track of what level the user is on. It will show when the ball bounces off the platform or wall or bricks. It will also show when the bricks break, along with having a transition in between levels.