Overview of the program

I have designed a breakout style game using the SwinGame API in the pascal programming language. The player is a paddle and you use the paddle to hit a ball, to break bricks, to gain score. The aim is to get the highest score and to break all the bricks on screen to progress.

The game ends by either the user closing the window or the player running out of lives from the ball passing the paddle and leaving through the bottom of the screen.

Main Features

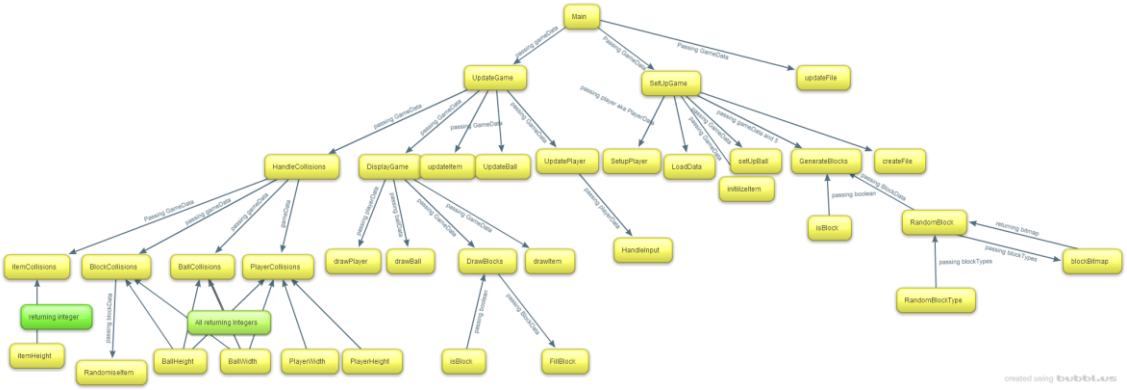
Paddle: The object the player controls in the game. By collecting items you can change the paddle size to be either larger or smaller. The paddle collides with all the ways and cannot get to the top half of the screen. You use the keyboard arrow keys to control the paddle. The paddle is built up of a playerName for entering in highscores, x and y coordinates for position, dx and dy for acceleration, bmp for a bitmap to show the paddle, score for the highscore board and lives for how many chances you have.

Ball: The object the player uses to destroy the blocks. Each time the ball collides with walls and the paddle it will accelerate. The ball can collide with the walls, paddle and blocks. The ball is built up of a bmp as a bitmap to display the ball on the screen, x and y coordinates for position and dx and dy for acceleration.

Blocks: The object that you destroy to gain score. The blocks will only collide with the ball. Blocks have a chance to drop items. blocks are generated at the start of the game. Blocks are built up of a kind which is a block type which includes items, hard and soft, bmp as a bitmap to display the blocks uniquely depending on their kind, x and y as coordinates and an item bool that determines if the block has an item or not.

Items: Items drop out of some blocks . items will collide with the paddle to effect the ball or paddle. They will fall down the screen as time progresses so you can pick them up before they fall off. Item record is built up of a bmp as a bitmap to display the items, x and y coordinates and a kind to determine what the item will do to either the ball or player.

Structure



Description of structure

The objects such as ball, paddle, blocks and items procedures and functions are stored together one structure/record for each and one to contain them all.

Progress of development

I originally scoped the game to be just an Atari breakout clone but as progress on development went by I thought to add items that the blocks could drop to change gameplay. I got stuck with some errors when creating the items code but after debugging it I easily found where the errors were and swiftly fixed the program to run smoothly again. After that I researched about APIs for the High Distinction requirements for pascal but couldn't find much on the internet so I decided to try and convert my project to C# and write the research report on object orientated programming. After trying to understand OOP and converting the code with instances of the objects error free I came across a lot of null reference exceptions which caused the game to not function. After trying to fix the missing instances of all of my objects I gave up and returned to my pascal code to get something that would work and could be shown. In retrospect it would have been easier to write the code from scratch and each object in its own file.

I then added the code to save, create and read to file.

With a week left I will attempt to make an in game highscore board that will appear after the player gets a game over.

Features I would like to add

I mostly completed what I set out to do when planning the game but when the program progressed I thought of some extra ideas to try to add.

I would like to add a pause menu, with features like saving and loading the game but due to a lack of understanding the swingame API and lack of time I decided to scrap the idea.

I wanted to add more items to change the game but due to a lack of ideas and time constraints I decided against it.

Things I like/dislike about my code

I like how it runs without errors.

I like how clean it looks with all the objects like ball, paddle have procedures and functions stored together.

I like how I have a lot of comments telling what each function/procedure does and where each object starts and ends.

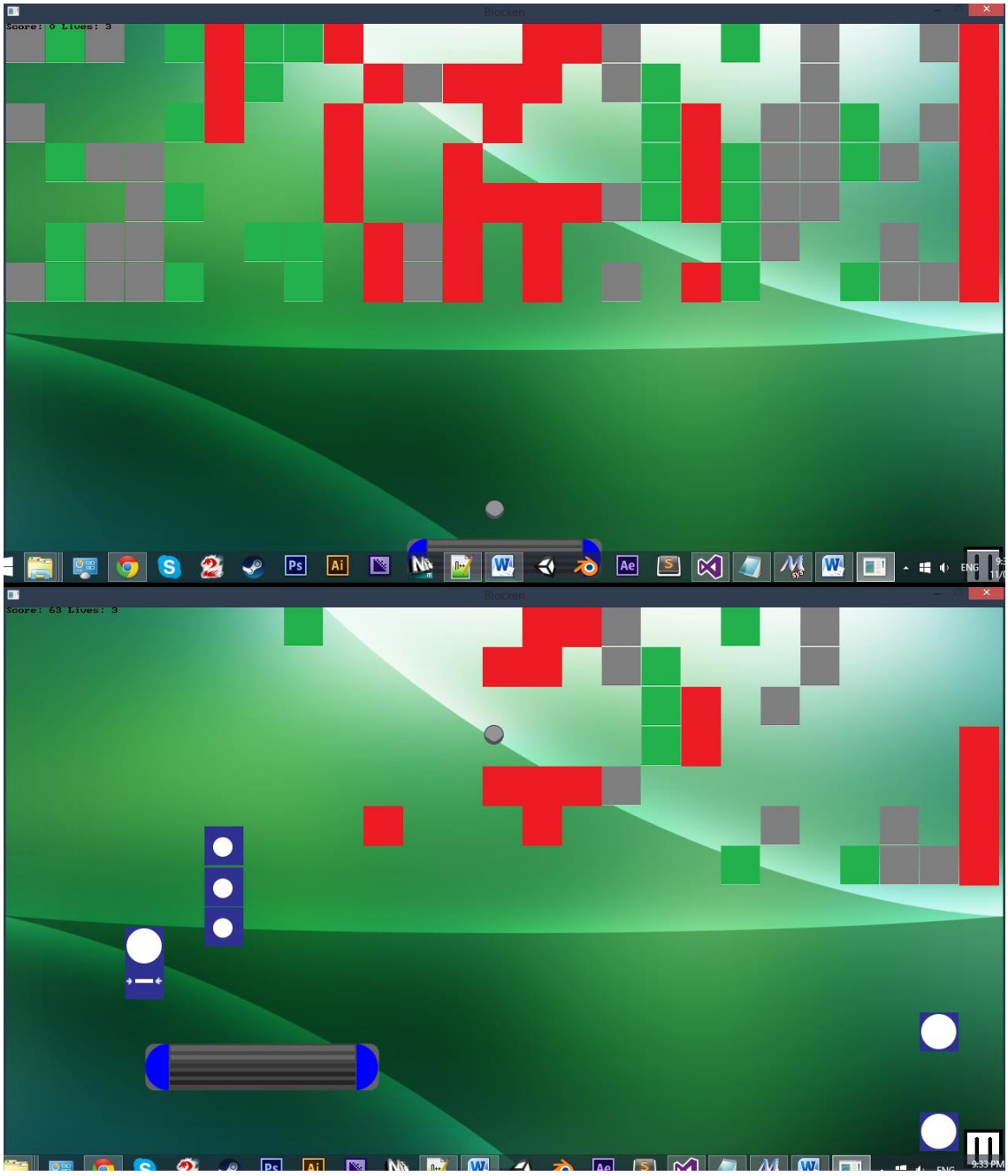
I feel proud that I got the item code to work after getting runtime errors.

I like when the ball collides the left side of the paddle it will go left, the middle it will go up and on the right it will go to the right.

I like my game so that when items are collected, the items effect the paddle and ball.

I didn't like my code for some of the collisions such as when the ball hits the bottom of the paddle, the ball will actually go up.

Screenshots


Blocken
Score: 0 Lives: 3
Blocken
Score: 63 Lives: 3