# Breakout MVP and Documentation

## Problem and Decomposition

Create an Atari Breakout like game modelling its physics, abstract design, and general gameplay logic. Design the software using Microsoft WinForms making and demonstrate understanding of the object orientated paradigm and event driven programming.

The games components and features easily mold to the OOP architecture.

## Classes and Structures

### Game

An object that stores and manages multiple game objects and handles their updating and rendering to the game screen via double buffering for a smooth display.

### Game Object

any abstract object in the game that has a position, and a texture, and takes part in the physics simulation in some way.

*Game Objects* know their positions and have will have functionality to draw and update their positions to the screen

### Ball

A game object that mirrors its velocity vector on Collison with another game object causing damage to any *bricks* it collides with and varies its direction based on the position it collides with a *paddle*.

### Paddle

A game object that follows the mouse horizontal position within the form applying differing angular velocity to a *ball* should it collide.

### Brick

a game object that is destroyed when collided with, later adding its score value to the games score and exploding on destruction.

## Functionality

### events

The player will gain a point each time the ball collides with a brick, the value increasing for denser bricks (more hits to destroy).

The player wins a level when they have destroyed all the generated bricks, then proceeding to the next level. The player wins the game when they have beat all the provided levels (there will be three in this case).

The player can lose a life if the ball falls below the screen height, losing the entire game if running completely out of lives (three in this case). the player may regain lives from a dropped 'extra life' powerup (**augment**), and lives are regenerated on completion of a game level.

### Form Design

Although built in windows forms. Main use of the provided components will be phased out used to ensure the games aesthetic matches that of a pixel perfect recreation of breakout. This will involve the creation of custom classes such as *Buttons* and *Text Boxes* so it will not be recognisable as a windows form’s app in any way.

The game will make use of a single *Bitmap* to render the game content using *Graphics* objects with double buffering with a single *Timer* to handle the main game loop. I plan to push all the game code out of the *Form1.cs* file and use a separate *Screen* class to manage rendering to completely detach the *Game* class from *Graphics* and *Image* objects.

## Minimum Viable Product

Like mentioned in the **Classes and Structures**section the MVP follows a OOP architecture of 5 classes with the following UML diagram and form design

A picture containing graphical user interface

Description automatically generatedA screenshot of a computer

Description automatically generated

The MVP doesn't have a score, lives, or ability to win or lose. I just created the physics and destruction of bricks. The MVP has main visual functionality of Breakout without terribly extravagant features.