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 mould to the OOP architecture.

The player can bounce a ball of a paddle by controlling its position with the mouse. The ball deals damage to the bricks it hits eventually destroying them for varying points. Completing the game within 4 minutes awards varying time bonuses in the form of the score multiplier. The player wins a level when all the bricks are cleared. Completing all three levels to win the game. The player can lose the game if the ball travels bellow the paddle more than 3 times in a single game playthrough—returning to level one again. Depending on how many lives the player completes a level will determine another score multiplier.

Classes and Structures

Name	Derives	Description
Abstract Game	Object	Defines basic methods for an abstract game such as lists to manage objects, animations, and tasks. It's the Games job to handle the creation and freeing of objects and components in a safe way.
Breakout Game	Game	An application of a game that manages the game of breakout. It creates levels, the paddle and ball, and various user interfaces.
Abstract Game Component	Object	A game component defines a component that knows what game it's a part of a reference to random, and a Screen to draw two. Nearly all classes derive from this class to reduce repetitive parameterisation in constructors
Game Object	Game Component	An object with position, dimensions, velocity, and a texture that can be added and removed from a game and derived to extend its functionality. Objects can draw themselves, optionally respond to collisions, and update themselves.

Brick	Game Object	A derived Game objects that defines an object with density and value, that can explode with its provided texture.
Regrowth Brick	Brick	A brick that adds itself back into the game a set time after it has been freed.
Main Menu	Game Component	A component that manages a collection of buttons, toggles, and text to allow the user to choose what to do when the game is run. Such as play, view guides, configure the game, or view the credits.
Cursor	Game Object	A Game object that follows the mouses position so the user can see where the real cursor is (as it is hidden)
Text	Game Object	A Game object that renders a collection of game objects with lettering textures to display a message on the screen. Managing the creation, updating, and removal of such objects.
Button	Text	A text objects that calls the provided action delegate (call-back function) when the mouse clicks within its collision box (AABB)
Toggle	Button	A Button with state. Calling the provided action delegates when toggled and animating the toggle texture.
Paddle	Game Object	A Game object that follows the mouses Y position and takes part in the game's physics simulation.
Ball	Game Object	A Game Object that bounces off bricks, walls, paddles, and worms. Applying damage to bricks and decrementing lives when traveling below the paddle.
Worm	Game Object	A Game objects that moves across the screen horizontally with random wait times participating in the physics simulation. They appear in the third level.
Backdrop Manager	Game Object	scrolls a game object texture over the screen in a conveyor type cycle to give the illusion of an infinite backdrop.
Animation	Game Component	Manages the switching of a collection of textures on a provided game object. Can loop infinitely, a set number of times, and optionally call actions when done animating.

Level	Game Component	Groups a collection of bricks together and their arrangement on the screen and decides when bricks should drop augments.
Second Level	Level	Extends the functionality of a level, adding regrowth bricks along the bottom row of regular bricks.
Third Level	Second Level	Extends the functionality of the second level by spawning 3 worms below the regrowth bricks for extra difficulty
Screen	Object	Bridges the drawing of objects with the forms graphics objects. Managing scaling and the position of the mouse and click events.
Form 1	Form	Creates the Breakout Game, starts the timer, and calls the Main game loop per timer tick. The form passes information such as click events and mouse positions into the Breakout games screen.
Structure Vector 2D	-	A simple structure to manage a vector in 2d space with zeroing and inverting methods.
Task	*Object*	Stores an action delegate to be called by the Game class a set number of milliseconds later. A simple form of asynchronous programming without threading.
Enum Direction	-	Stores a group of constants that act as cardinal directions.
Tile Set	*Object*	Manages the provided image as a group of tiles, returning texture source rectangles to game objects and animations.
Static Time	-	A static class to group a collection of constants referencing time in milliseconds
Augment	Game Object	A game object dropped from bricks by a level. Augments plug in new code when collided with by the paddle and reject such code when a condition is met.
Triple Ball Augment	Augment	Adds 2 extra balls to the game, animates the balls and paddle, and rejects when there are less than 2 balls left above the paddle (also when the game ends or level changes)
Exploding Ball Augment	Augment	Causes random bricks to be deleted from the current level when the ball collides with any brick. Rejecting after its application. The ball and paddle animate while applied.

Functionality

The player will gain a point each time the ball collides with a brick, the value increasing for denser bricks (more hits to destroy). Bricks will award 12 points for the first hit, with subsequent hits awarding 24, then 36.

Although the brief suggested 10 points per brick hit, I decided to use 12 as it is my favourite number.

The first level has minimal functionality, with 6 rows of bricks. The second level introduces regrowth bricks that 'regrow' after they've been destroyed. They award no points and do not deflect the ball when collided with positive vertical velocity (downward ball). The third level expands the second levels functionality by adding annoying worms that traverse the screen horizontally below the rows of bricks. They merely deflect the ball on Collison the same as the regrowth bricks and are not destructible for points.

If the user specifies the 'single level mode' option. Only the first level needs to be completed for the game to be won.

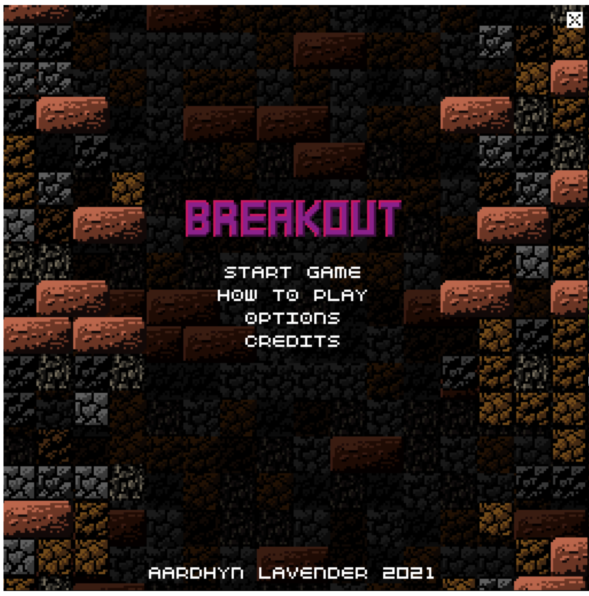
Although the player can lose a life if the ball falls below the screen height, if the Triple ball augment is active, the player will only loose a life when the last ball goes off the screen.

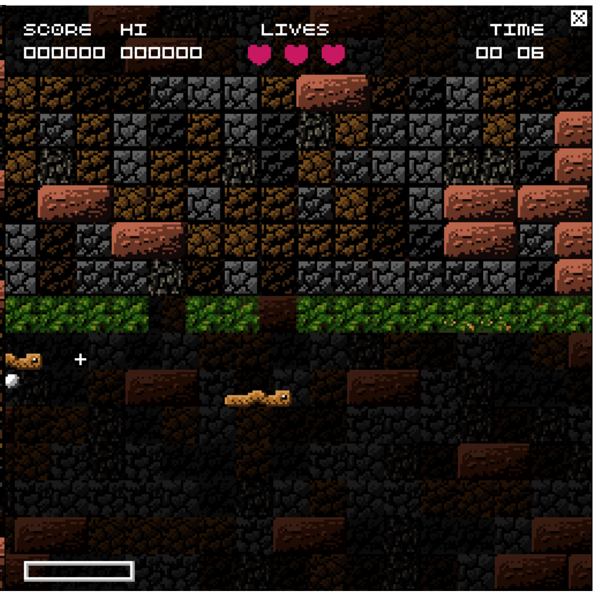
Form Design

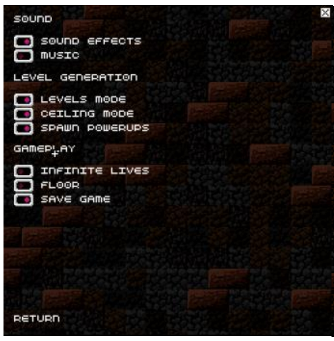
I used only two graphics objects, and image, a Stopwatch, and a timer from the WinForms library. All the text, objects, toggles, and buttons are custom classes. Input such as mouse position and left click state are passed in from the Form1 class.

To further the disembodiment of game to WinForms, I removed the application border, adding in custom window dragging and a close button. The single image being rendered to the screen has reduced resolution with nearest-neighbour interpolation to make the pixels crisp without artifacts or blur. This ensures pixel snapping for all rendered game objects for a true pixel perfect game. Unfortunately, windows forms seems to struggle to render the pixel data smoothly causing some minor horizontal synchronisation issues with moving objects like the backdrops and balls.

I decided against using multiple windows (multiple forms) for the games HUD or menu system. Most games run in a single window, so I followed that model with Breakout.



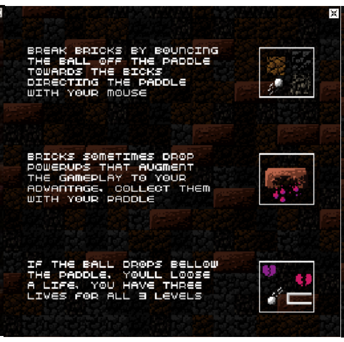



SOUND
- SOUND EFFECTS
- MUSIC

LEVEL GENERATION
- LEVELS MODE
- CEILING MODE
- SPAWN POWERUPS

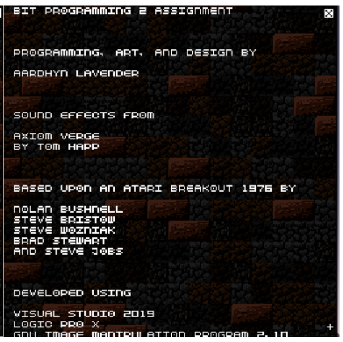
GAMEPLAY
- INFINITE LIVES
- FLOOR
- SAVE GAME

RETURN


BREAK BRICKS BY BOUNCING THE BALL OFF THE PADDLE TOWARDS THE BICKS DIRECTING THE PADDLE WITH YOUR MOUSE

BRICKS SOMETIMES DROP POWERUPS THAT AUGMENT THE GAMEPLAY TO YOUR ADVANTAGE. COLLECT THEM WITH YOUR PADDLE

IF THE BALL DROPS BELLOW THE PADDLE, YOULL LOOSE A LIFE. YOU HAVE THREE LIVES FOR ALL 3 LEVELS


BIT PROGRAMMING 2 ASSIGNMENT

PROGRAMMING, ART, AND DESIGN BY

AARDHYN LAVENDER

SOUND EFFECTS FROM

AXIOM VERGE
BY TOM HAPP

BASED UPON AN ATARI BREAKOUT 1976 BY

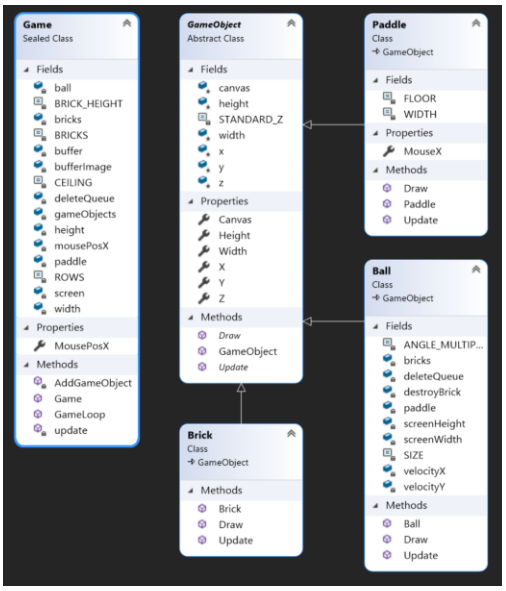
NOLAN BUSHNELL
STEVE BRISTOW
STEVE WOZNIAK
BRAD STEWART
AND STEVE JOBS

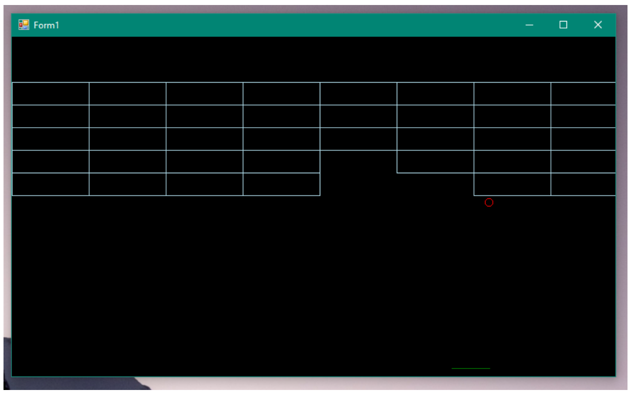
DEVELOPED USING

VISUAL STUDIO 2019
LOGIC PRO X
GNU IMAGE MANIPULATION PROGRAM 2.10

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

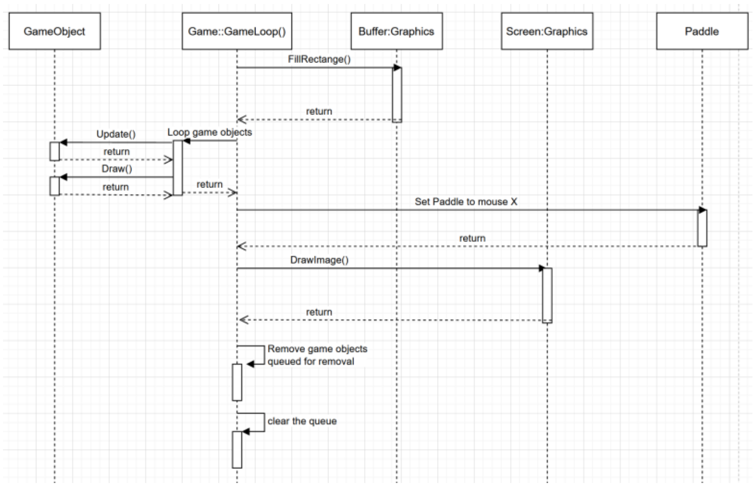




The MVP doesn't have a score, lives, or ability to win or lose. I just created the physics and destruction of bricks. The MVP has main visual functionality of Breakout without terribly extravagant features.

MVP Sequence Diagram

Here is a sequence diagram of the main game loop in Breakout MVP.



Final Product Class Diagrams

Found in Breakout Project—*ClassDiagram1.cd*—as it is simply too large to display legibly in this document