"Script and Shell: A Reflection on Game Design for Learning"

By Robert Turner – GAM301

Statement of Intent

Introduction

For my GAM301 project, I decided to create a prototype for a simple programming game. Using coloured "blocks" of code, players must control a tank and lead it to and destroy all targets on a small puzzle board. Players are given a certain amount of code blocks and must organise them into the correct order to solve the puzzle.

Intent

With this project I hope to create a game that can fill an educational role, helping to teach early secondary school children (12-15 years old) the basics of problem solving with programming and computational thinking. I also intend to fill a niche in this genre of game by focusing on sensory-based learning, using sound design, controller vibration and colour-coded blocks of code. I aim to help reinforce learning by engaging different sensory modalities, catering to diverse learning styles, while also making the game accessible to children with dyslexia and other learning disabilities through use of sensory-based learning. Each action in game should be reflected to the player using multiple senses. Each action will have a visual indicator, a sound, and a controller vibration pattern all unique to that particular action and its result (e.g. a tank moving forward successfully and the tank moving into a wall have different sound effects and vibrations of different intensities).

Features

Upon opening the prototype of the game, the player is greeted with this workspace. As seen below (Figure 1), the screen is split into three. On the left is the build zone, a space where the player will build their chain of code blocks to solve the puzzle. The two white buttons in the corner allow the player to reset the workspace and execute their solution. In the centre of the screen is the players toolbox. This is where the blocks available to the player to solve the current problem are shown and can be clicked on to add them to the chain. Almost each block is a different colour, except blocks with similar functions sharing a colour, such as the commands for turning seen in Figure 1. There also exists an indicator next to each block showing how many the player may use for each puzzle. On the right of the screen, we may see the puzzle view. The player controls the white tank and must use the commands given to navigate the terrain and destroy the enemy tanks. The game also features controller support, with the player being able to move the cursor and select blocks with both the left stick and d-pad. Controller vibrations provide tactile feedback for every action, with each action has a unique vibration pattern.

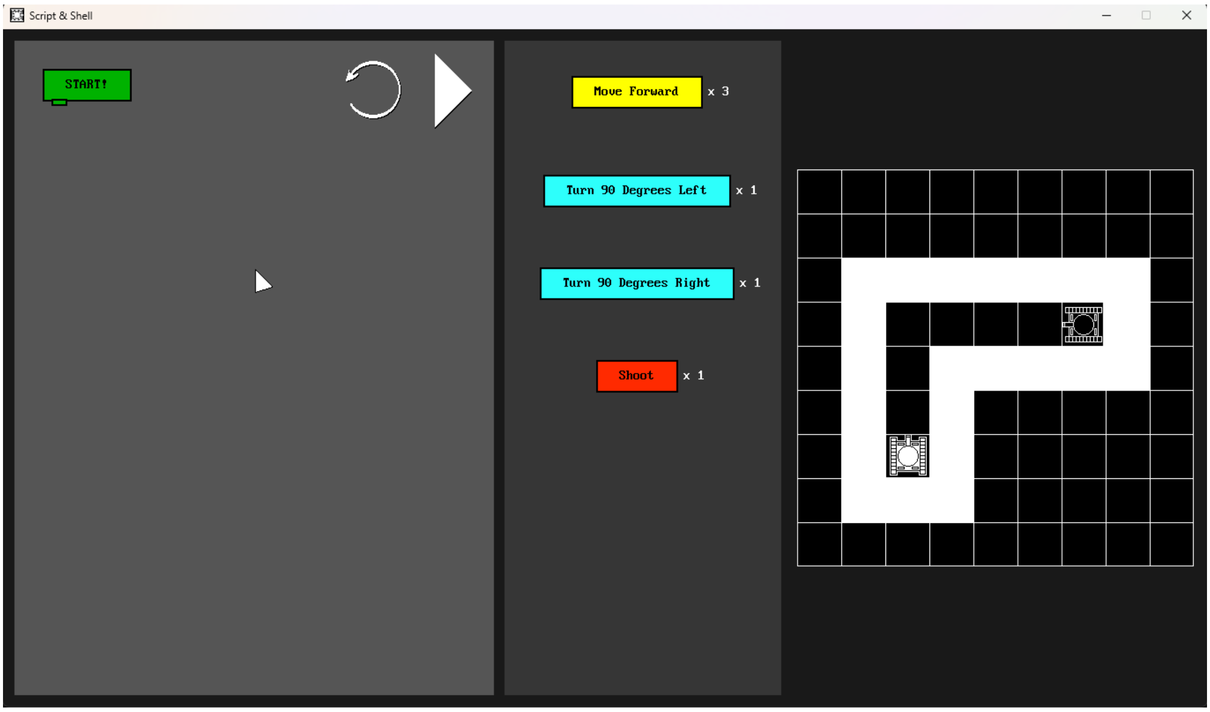


Figure 1: The first puzzle in the prototype of the game, shown in the workspace provided to the player.

Upon building their code and pressing the play button, the game will begin to step through the code, highlighting each block as it executed, while also displaying it beneath the puzzle view (Figure 2).

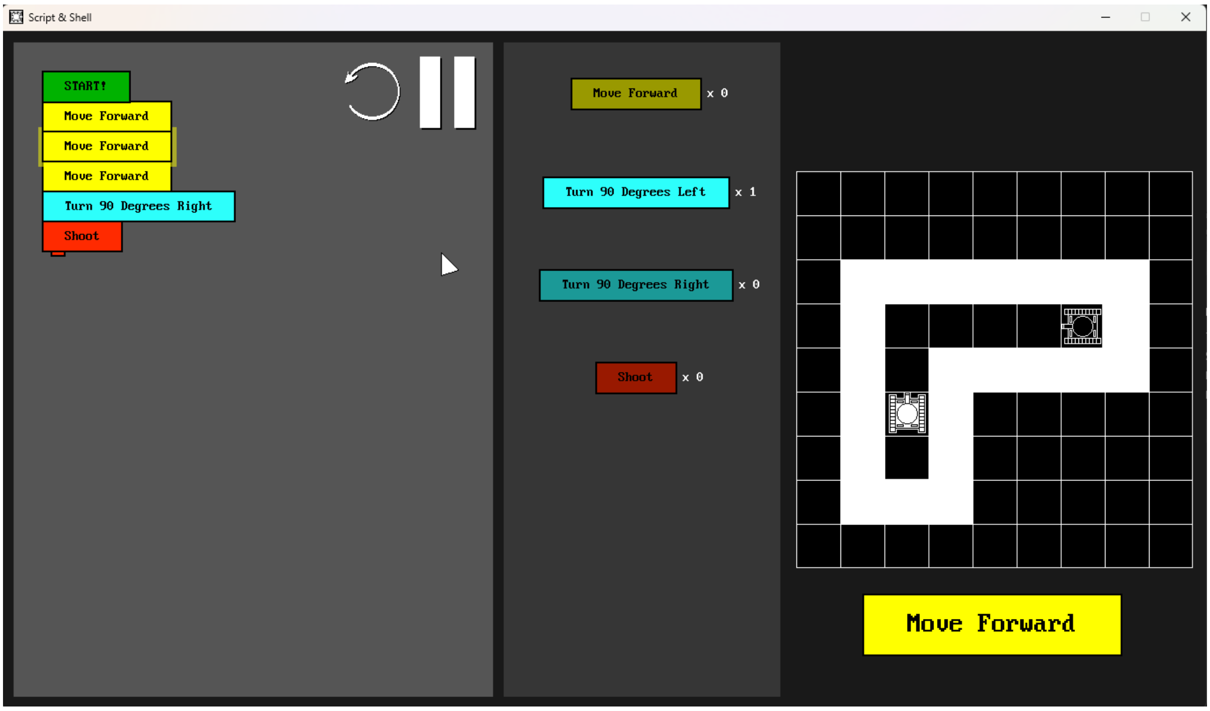
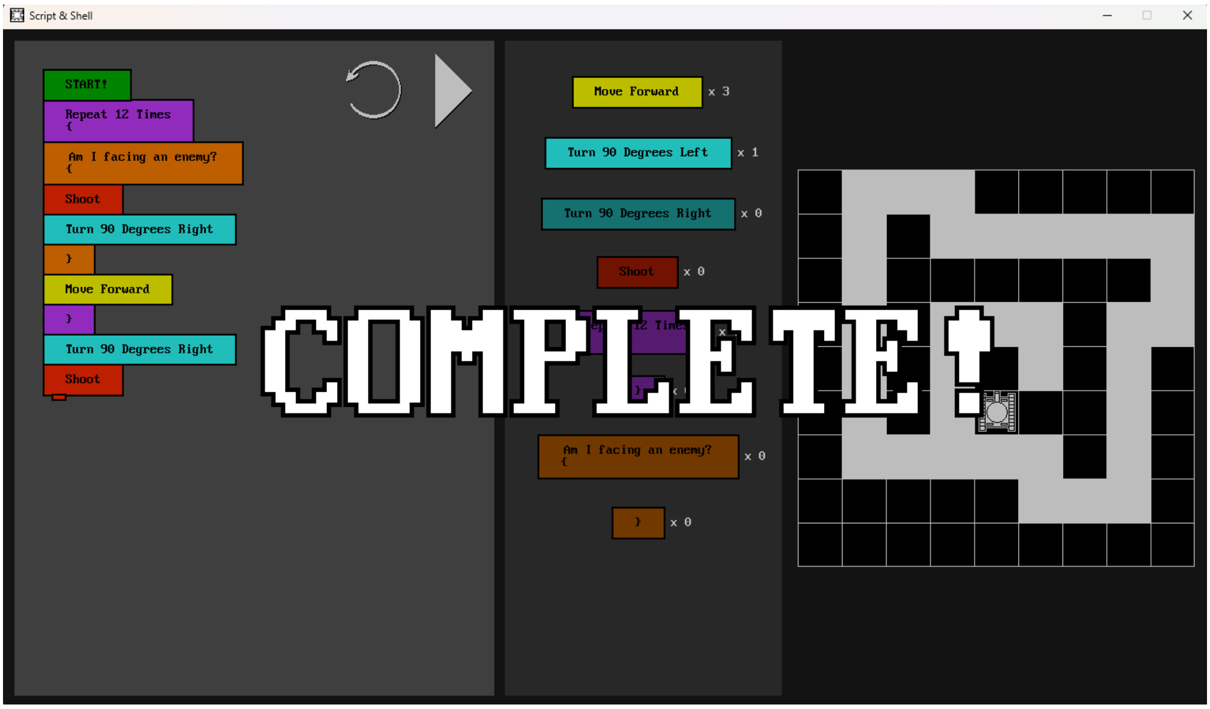


Figure 2: The first puzzle of the game in the process of being solved, with the blocks of code on the left being stepped through the puzzle on the right.

When the puzzle is solved, a congratulatory message will show (Figure 3), and the game will fade to the next stage. This short prototype contains 3 different stages aiming to showcase loops and conditionals, important parts of programming at any level. If the players solution does not complete the stage, the player will be able to replay their code to find any issues and reset the board state when they are ready.



```
START!
Repeat 12 Times
{
Am I facing an enemy?
{
Shoot
Turn 90 Degrees Right
}
Move Forward
}
Turn 90 Degrees Right
Shoot
```

Figure 3: A completed level showing a congratulatory message.

Implementation

The features of the game have been designed to meet my intent with this project. The first of my goals, to create a game for teaching early secondary school children (12-15 years old) the basics of problem solving with programming. The simplified block-based coding system abstracts complex programming concepts, introducing loops and conditionals in a clear, problem-solving context. The compartmentalized UI ensures clarity and ease of use for early learners. My other goal with this project was to cater to diverse learning style through sensory based learning. The use of sound, colour, and vibration engages multiple senses, reinforcing diverse learning styles.

Connection to Domain

Introduction to the Domain

My chosen domain “Games for Learning” entails the process of using game mechanics and interactivity to make educational experiences more engaging and effective, also referred to as the gamification of learning. What exactly is meant by gamification varies widely, but one of its defining qualities is that it involves the use of game elements, such as incentive systems, to motivate players to engage in a task they otherwise would not find attractive (ERIC 2024). This domain of the industry focuses on using games to teach skills and increase knowledge retention. This industry is growing fast, predicted to reach a global value of $30.7bn by 2025 (Instinct 2023).

How "Script and Shell" Aligns with Games for Learning

“Script and shell” aligns with games for learning as it uses the pretence of a video game and the included game mechanics to teach skills to the user. Its goals are also shared with similar projects. The goal of aiming to teach children programming and computational thinking is shared with the previously mentioned online ‘Scratch’ tool. As stated on their about page, “Scratch promotes computational thinking and problem-solving skills” (Scratch 2024). “Script and shell” is also a short game focusing on teaching programming concepts using blocks, like many of the activities on the “Hour of code” website. (Hour of Code 2024).

The Role of Multisensory Feedback in Learning Innovation

Unlike other games in the domain, “Script and Shell” focuses on using multisensory feedback to reinforce learning. The use of sound, colour and touch have been proven methods to help retain knowledge. "Multisensory technologies… engage visual, auditory, and tactile modalities simultaneously, significantly enhancing the learning experience by reinforcing understanding and memory" (Frontiers in Psychology 2020). I hope that the implementation of these methods in my project will help it stand out from similar projects and help it achieve its goals of teaching programming concepts and computational thinking.

Reflection on Practice

Development Process

While reading the brief for this assignment, one of the four domains we could cover stood out at me – Games for learning. It is a subject I am familiar with - in primary school I founded and ran a club based around the online “Scratch” tool, one of the main inspirations for this project. I aimed to design a simple puzzle game where players solve problems using block-based coding, enhanced by sensory feedback. Considering scope and my skills, I focused on creating three levels with straightforward mechanics, such as navigating a grid and firing at targets. (Figure 4).

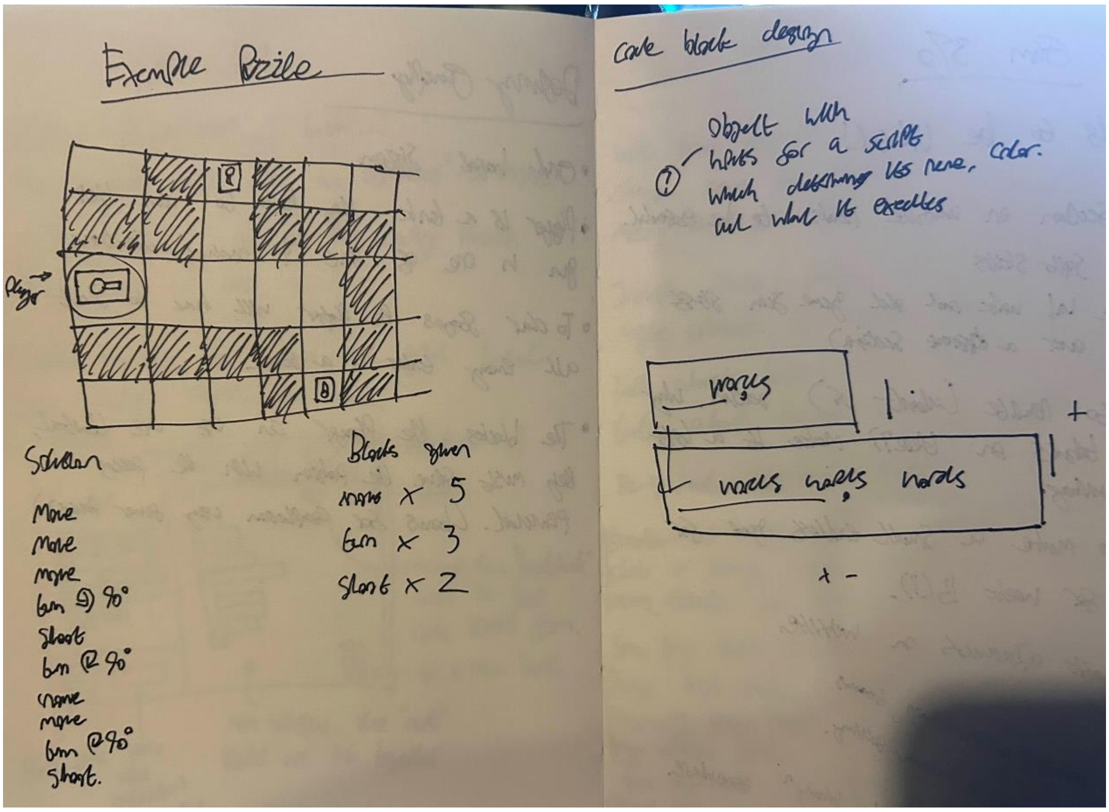
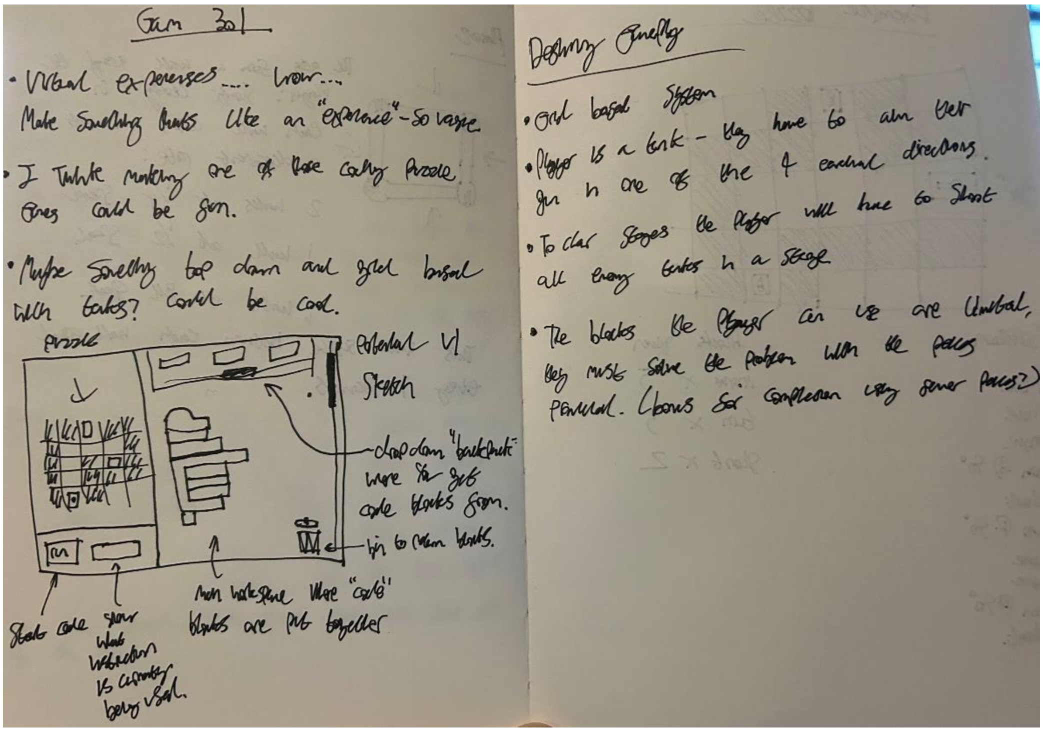


Figure 4: Some notes on an example of the puzzle system used in the game, with the layout, blocks given and solution on display. The right page contains planning out for the programming of the blocks, and how they would arrange themselves on the screen.

The main challenge for the player would be navigating the environment to reach their target – I decided this would be a good idea to help teach concepts such as conditionals and loops. I decided to use Gamemaker Studio 2 (YoYo Games 2024) for this project. It’s a tool I am familiar with, and its simple 2D capabilities would make it perfect for this prototype which should prioritise functional mechanics over graphics. It was also a great pick as I am well-versed on implementing sound effects and controller vibration in the engine. I decided I would use the SFXR tool (DrPetter 2007) to generate my sound effects, as it’s a very easy tool to use, and would provide me with some solid sound effects to help give proper feedback to the player. I am also well experienced with this tool, and use it in almost all my projects, as despite its age it is very effective. Around this time, I began designing UI for the program, seen below in Figure 5.



Game 301

- Virtual experiences.... hmm...
Make something thats like an "experience" - so vague.
- I think making one of those coding puzzle games could be fun.
- Maybe something top down and grid based with tanks? Could be cool.

Puzzle
Potential UI Sketch
drop down "backpack" where you get code blocks from.
bin to remove blocks.
main workspace where "code" blocks are put together
Start code
show what instruction is currently being used.

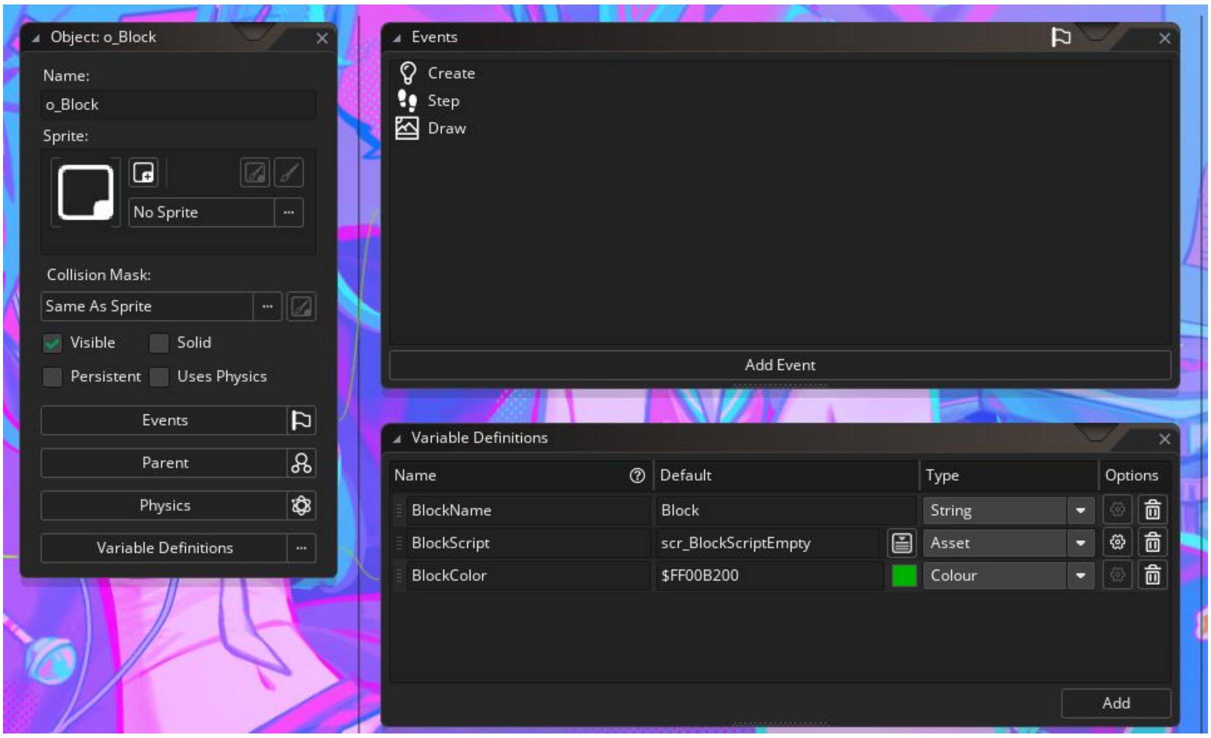
Defining Gameplay

- Grid based system
- Player is a tank - they have to aim their gun in one of the 4 cardinal directions.
- To clear stages the player will have to shoot all enemy tanks in a stage
- The blocks the player can use are limited, they must solve the problem with the pieces provided. (bonus for completion using fewer pieces?)

Figure 5: My initial notes on the project while brainstorming. It includes my ideas on a potential UI design (heavily inspired by the 'Scratch' UI and some thoughts on how the puzzles would work on the right page.

After much thought I decided to simplify the UI even further into the 3 segments structure used in the prototype. After this I added a basic block object into the game. The block object was designed with editable properties, such as function and colour (Figure 6), enabling modularity. This approach allowed for efficient implementation of loops and conditionals as gameplay elements. After programming the puzzle elements and making sure they all interacted correctly, I added a ‘Play’ button to execute the player’s solution. This would then step through the code block by block, ensuring each instruction corresponds the desired effect on the puzzle. With these features in place, I designed three stages and began testing.

During testing, it became apparent that players struggled to follow which block was being executed when running their solution. To address this, I added a feature where the active block is highlighted as the game executes the code, offering immediate visual feedback. I also added a display beneath the puzzle view to show the current block, allowing players to directly associate their logic with the following actions. These changes made the game clearer and easier to follow, helping players see how their code would affect the puzzle. By improving the connection between the code blocks and their outcomes, the game better achieves its goal of teaching logical thinking in an engaging way.



Name: o_Block

Sprite: No Sprite

Collision Mask: Same As Sprite

☑ Visible ☐ Solid

☐ Persistent ☐ Uses Physics

Events:

- Create
- Step
- Draw

Name	Default	Type	Options
BlockName	Block	String	
BlockScript	scr_BlockScriptEmpty	Asset	
BlockColor	$FF00B200 🟩	Colour	

Figure 6: A screenshot from the Gamemaker 2 editor showing the block object - The block name, colour and script that it executes can each be edited and assigned per instance.

Future Applications

My work in creating "Script and Shell" in Gamemaker Studio (YoYo Games 2024) and implementing sensory-based learning mechanisms for this assignment has helped me develop my skills in creating educational experiences, considering accessibility and the best ways to gamify learning. These acquired skills broaden my horizons, allowing me to potentially move into the quickly growing "Games in education" sector of the industry. This is a promising industry to work towards as it is getting bigger every year, predicted to reach a global value of $30.7bn by 2025 (Instinct 2023). In the future, I'd like to expand this game to incorporate more advanced programming concepts, such as functions and variables. I've furthered my skills in developing modular systems and best practices, something very applicable to jobs in industry. The project also fits in nicely with my other portfolio pieces, showing my adaptability and ability to produce a diverse range of game genres. Overall, this project strengthens my professional portfolio by showcasing a unique blend of technical and design skills, helping me get closer to my goal of working in the games industry.

Bibliography

1. SCRATCH. 2024. *About Scratch* [online]. Available at: https://scratch.mit.edu/about [Accessed 3 December 2024].

2. HOUR OF CODE. 2024. *Activities and Tutorials* [online]. Available at: https://hourofcode.com/ [Accessed 3 December 2024].

3. ERIC. 2024. *The argument for game-based learning* [online]. Available at: https://files.eric.ed.gov/fulltext/ED610946.pdf [Accessed 5 December 2024].

4. FRONTIERS IN PSYCHOLOGY. 2020. *Multisensory interactive technologies for primary education: From science to technology* [online]. Available at: https://www.frontiersin.org/articles/10.3389/fpsyg.2020.00786/full [Accessed 7 December 2024].

5. INSTINCT. 2023. *The rise of eLearning gamification in higher education* [online]. Available at: https://www.instinct.co.uk/blog/the-rise-of-elearning-gamification-in-higher-education/ [Accessed 5 December 2024].

6. YOYO GAMES. 2024. *GameMaker Studio 2* [online]. Available at: https://gamemaker.io/en [Accessed 12 December 2024].

7. DRPETTER. 2024. *SFXR* [online]. Available at: https://www.drpetter.se/project_sfxr.html [Accessed 12 December 2024].

Word Count: 1614