**C++ For Games Designers Post-Mortem**

# Project Description

This project is a 3D third-person platformer with a retro feel to it. The idea behind this game is to dodge on coming cannonballs and collect purple gems to get to the finishing flag. In this game, C++ was used in the following: character movement, gem pickups, cannonball spawner, cannonball damager and audio implementation.

# C++ and Blueprint Integration

I have attempted to use mainly C++ in this project to understand its benefits and drawbacks. I have used C++ to hard code important mechanics that are not needed to be accessed often for change, e.g. character blueprints. I have also used C++ to make parent classes that blueprints can inherit from and give more features e.g. actors that collide with the player (cannonballs and gems).

I had consulted parts of [‘Unreal Engine 5 Game Development with C++ Scripting’](https://learning.oreilly.com/library/view/unreal-engine-5/9781804613931/) edited by E. W. Roberts for initial guidance with the game. I had also consulted tutorials about creating [moving platforms in C++](https://www.youtube.com/watch?v=U-BZQuVjugg) and making a [pixel post processing material in blueprint](https://www.youtube.com/watch?v=-zuuWCAbyfM&t=315s).

From this project, I believe the benefits of C++ are creating parent classes. It allows flexibility within the child blueprint (from the C++ class) and add more features to the blueprint without compromising the existing code in any way. On the other hand, blueprints lend themselves to easy implementation in projects and are best for prototyping concepts of game mechanics. It also allows for designers to easily edit the code for fast prototyping and testing as opposed to C++ wherein you must edit the class itself and compile the code before knowing what must be changed.

Quality of the C++ code

I had organised the code into public and private folders and gave sensible names for them e.g. *‘MovingPlatform’*, *‘PlayerCharacter’* etc…

A screenshot of a computer

AI-generated content may be incorrect.

Due to this being my first time writing C++ code I feel as if I didn’t comment my code correctly in places. Furthermore, I made minimal considerations for performance or memory management. This means the quality of my code is below standard.

Quality of Game Feel and Mechanics

For the game feel, I wanted to achieve a retro look. To do this, I implemented a pixel post processing material into the world to help with this. A concern of mine was making sure the pixels were not hindering the readability of the game. I had messed around with the material multiple times to get the desired effect that I wanted. Another addition was music in which I sourced out [this instrumental](https://pixabay.com/music/video-games-retro-game-arcade-236133/), which was made by [*‘moodmode’*](https://pixabay.com/users/moodmode-33139253/?utm_source=link-attribution&utm_medium=referral&utm_campaign=music&utm_content=236133%22%3eVlad%20Krotov%3c/a%3e%20from%20%3ca%20href=%22https://pixabay.com//?utm_source=link-attribution&utm_medium=referral&utm_campaign=music&utm_content=236133), and coded it into the game using C++. I wanted the music to start right before the platforming sections to emphasis that section of the level instead of starting immediately. In future, I will add sound effects for character actions and particle effects to gems to give more life. These simple additions would make the game feel more alive than what it currently is.

I had decided early on that the character functions, pickups and damagers were going to be done in C++. This was mainly done due to these functions not being needed to be frequently accessed (e.g. player character base functions) and to make use of parent C++ classes and child blueprints from C++ classes (e.g. damagers and pickups). It allows the important code to be unchangeable easily and allow designers to change the child actor variables needed, e.g. damager amount change.

A diagram of a diagram

AI-generated content may be incorrect.

Another reason was to stretch myself with my coding skills as I have never done C++ coding within unreal engine. However, I had searched for [a tutorial](https://www.youtube.com/watch?v=U-BZQuVjugg) about creating moving platforms and decided to give it a try and now it is also another C++ class used in game!

The moving platform C++ class is also very designer friendly as changing and editing the path of the platform can all be done within the editor in scene. Other examples of functions and variables that designers can change within the editor include the audio track for the background music, player health, gem count and movement variables within the player blueprint. Some sections of the code are not exposed to blueprints, e.g. player move and looking control values, collision boxes on pickups and damagers etc... This is done because there is no need to change these variables or functions in any way.

# Overall Quality of Work Produced

Overall, this project has taught me a lot about C++. I had no knowledge about C++ before this project however did have experience in C# coding and blueprints. The theory is very similar, but the implementation is different.

A key challenge that I had to over come was creating the spawning mechanic for the cannonballs, more specifically the timer for the cannonball’s spawner – the cannon. As the cannon was to fire continually, it was decided to use a timer to handle this. However, even after multiple times after tweaking the code, the cannonballs still wouldn’t spawn. This eventually led to unreal engine itself crashing and being unable to open. After looking at the code again a solution was put in place. Originally the code was within the ‘Construct’ section of code – *‘AcanonballSpawner::ACanonballSpawner()’*.

A computer screen shot of a computer screen

AI-generated content may be incorrect.

This is incorrect because construct code executes when unreal engine opens, not when the game starts. Thus, the cannon was trying to spawn cannonballs while unreal engine was in editor mode causing crashes. To fix this issue, I simply moved the timer code into the begin play section of code – ‘*AcanonballSpawner::BeginPlay()’*.

A computer screen shot of a computer screen

AI-generated content may be incorrect.

This was the only significant key challenge that occurred during coding. Another minor issue was my overall lack of C++ skills causing me to down scale the project greatly to make it very simple.

In future game developments, I will use C++ to create parent class for blueprints to inherit from as this methodology works well for me and allows more interesting mechanics to be done.

# Playthrough Video Link

<https://youtu.be/wWQhALP8g80>

# Functionality

|  |  |  |
| --- | --- | --- |
| **C++ Game Feature** | **Implemented?**  **Yes/No/Partial/ Forum Only** | **Forum Link/s to topic – to specific posts where possible, and example C++ Class name where used.** |
| Enhanced Inputs | Yes | <https://digitalacademy.staffs.ac.uk/forum/index.php?/topic/77544-abbas-sarah-a018024m/>  (Character C++ Class) |
| Overlap and/or Hit events | Yes | <https://digitalacademy.staffs.ac.uk/forum/index.php?/topic/77544-abbas-sarah-a018024m/>  (Gem pick ups, Damagers, Audio, Level end C++ Class) |
| Line Trace / Hit Scan |  |  |
| Actor Spawning | Yes | <https://digitalacademy.staffs.ac.uk/forum/index.php?/topic/77544-abbas-sarah-a018024m/>  (Damagers and Spawners C++ Class) |
| Actor Components (built-in) | Yes | <https://digitalacademy.staffs.ac.uk/forum/index.php?/topic/77544-abbas-sarah-a018024m/> |
| Actor Components (custom) |  |  |
| Game Modes |  |  |
| Character Movement | Yes | <https://digitalacademy.staffs.ac.uk/forum/index.php?/topic/77544-abbas-sarah-a018024m/>  (Character C++ Class – also seen with multiple videos) |
| Structs |  |  |
| Enums |  |  |
| Arrays | Yes | <https://digitalacademy.staffs.ac.uk/forum/index.php?/topic/77544-abbas-sarah-a018024m/>  (Moving platforms C++ Class) |
| Maps |  |  |
| Interfaces |  |  |
| Event Delegates |  |  |
| UI with C++ data |  |  |
| Math function usage | Yes | <https://digitalacademy.staffs.ac.uk/forum/index.php?/topic/77544-abbas-sarah-a018024m/>  (Character health and Spawner C++ Class) |
| Data Assets / Tables |  |  |
| File I/O |  |  |
| Audio | Yes | <https://digitalacademy.staffs.ac.uk/forum/index.php?/topic/77544-abbas-sarah-a018024m/>  (Audio C++ Class) |
| Materials |  |  |
| Moving Platforms |  | <https://digitalacademy.staffs.ac.uk/forum/index.php?/topic/77544-abbas-sarah-a018024m/>  (Moving platforms C++ Class) |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| C6 Quality of Video Showcase |  | X |  |  |  |  |