Employable Skills

Skills and Abilities

Hard Skills

- Know development methodologies.
- Know how to use specific software like:
    - Unity Engine
    - Unreal Engine
    - Godot Engine
    - GameMaker
    - Blender
    - Maya
    - Adobe Photoshop
    - Microsoft Office
    - Microsoft Excel
    - Microsoft Powerpoint
    - Version Control - Git, GitHub, GitLab, Bitbucket
- Keeping up with relevant software in spare time.
- Professional written skills - spelling, grammer, punctuation.
- Programming / scripting knowledge for the relevant game engine.
- Understanding design and user interface principles.
- Portfolio and showreel.

Soft Skills

- Communication to a large range of industries.
- Being able to talk technical and layman terms.
- Know industry-specific technical terms for:
    - Designers
    - Artists
    - Programmers
    - Producers
    - Marketing staff
    - Investors
    - The public people during testing phases
    - Lawyers
- Knowledge of history of games.
- Knowledge of current games market.
- Taking constructive criticism and giving helpful feedback.
- Being punctual and professional when meeting outside parties.
- Being innovative and creative.

Strong Areas

My strong areas are mainly in the soft skills section. I like to add educational aspects to my games and thus take popular games, remake them in Unity so that they are endless, have replayability, and educational content.

Serious Games are a major genre in the video game industry, but the few serious games I have played tend to put the education first and then make a game around it. I try putting the fun first with replayability, and through beating their own personal high score, the education slowly starts to seep in. As this is my personal foundation to build games, the ideas I come up with other people think are weird but innovative.

Throughout my development process, I constantly try to imagine how the game would feel to the player and how easy or hard it is to pick up. I confirm my bias by talking to my friends who do or do not play games. They then report back to me how the game feels in its current state, and I adjust the game accordingly so any person would be able to pick up and play it (after reading through the short “how to play”). This also helps me understand design and user interface principles. Since the majority of my friends are not in the games development module, they are quite interested in how I made it and how I am going to make this new mechanic I talk about - therefore I quite enthusiastically explain to them how the game was made in a way they would understand.

My friends know they can be as harsh as possible as it goes towards making the game as best as possible, and I try to implement the feedback given constructively. Ideas and mechanics can be dropped or picked up on a whim, but if it doesn’t work, will it still be worth it to try and implement it?

I feel I write quite professionally as it has been drilled into me from a young age and I check there are no grammatical or spelling errors by using Google. Afterwards I read through what I wrote, then again aloud, taking note of my breathing, how well it flows, and if I am repeating myself too much. Punctuality is also my forte and I am able to follow what I plan to do each day with the time available.

Knowing how to use specific software is an area where I am comfortable and not comfortable in. I know how to use a large number of software - as stated in my portfolio, but they are still at a basic level.

Weak Areas

That being said, I am quite happy to teach myself new software. The majority of university modules are “here’s a new software, one or two exercises to familiarise yourself with the program, and then build something according to the assessment”. It is a good and bad experience since I wanted to develop something, didn’t know how to do it and asked around, was introduced to an entirely new area I didn’t know about, and now can what was learnt quite proficient. However the bad experience is finding out about something which could have been taught at the very beginning, and there was this whole other shortcut available but not used.

I also feel my programming / scripting skills are not as developed and my current programming has been following a lot of online tutorials and modifying and implementing code in ways it will fit my current game. Recently I have been thinking my coding is full of “if statements” and not performance-efficient in a large scale game. However all of my games have been quite small due to the time allocated towards them so I didn’t have to worry about it being efficient. The simple remedy is practice, practice, practice, and knowing how to use specific software also falls under that belt as a lot of the software I have learnt - Unity, Adobe Photoshop, Adobe Premiere, Version Control - have all been self-taught and not in a “professional setting”.

Being able to talk technically with other industries can be fixed by networking and reading, and now would be the best time to do it as I am surrounded by relevant university students but I know when it becomes my full-time job, I will become professional in time and pick up on industry-specific jargon, but what I am not sure about is if my current knowledge is good enough.

The person whom has impacted me the most is not part of the games development industry, but an artist and a storyteller. Through his work I have followed through many years, he taught me a romantic view of humanity, relationships, and bringing out the best in other people. He has inspired me to leave a mark in the world, and through my own personal creative way, I wish to blend education and games in a way it will reshape the genre of Serious Games.