Stickout:
A 2D Game Foundation

Peter Karwowski
CS 480 Senior Project
Northern Michigan University

Introduction:

For eight years I've been attending Northern and considering the day I would get to present a senior project. The idea has changed over the years and has shifted from a standalone modification of an existing title, a virtual driving simulator utilizing familiar methods of input such as a steering wheel and pedals, and an application that would streamline the dealership inventorying process by snapping a picture of a window sticker to automatically upload information to a database or an online application to schedule appointments for service, but I made the decision to create what will at some point be a modular 2D template game.

Video games have been ingrained as a passion throughout my early childhood to today; my earliest years go back to when I received a Nintendo 64 for Christmas with a copy of Cruis'n USA and an InterAct V3FX racing wheel. The journey would take me through modifying titles such as the Pokemon series for GameBoy Advance using toolkits I found online to creating mods for Fallout 4 in which I managed to squeeze out the almighty "CatBat" – a baseball bat that would "meow" as you strike opposition in the Commonwealth - but it wasn't until I started attending university where I would actually get my feet wet and attempt to develop my own game.

It was difficult coming up and settling on an idea that motivated me. 3D is a direction that if executed properly would allow the user to immerse into the world given to them. It was my first year here when I utilized World Machine to create an first open-world driving game; given the ease of use and scale offered, enthusiasm quickly soared and landed

straight on its face when I discovered how time consuming filling a 10km x 10km world is (given that even that was a desert).

I fell back to tools I was familiar with using. In high school I was introduced to Adobe Photoshop. This software is probably the only application in the world I can say that I could not live without. The magic one can produce is simply astonishing; allowing you to do everything from producing digital assets to whitening one's teeth. Naturally I dove deep into the rabbit hole and started pouring copious amounts of time into utilizing the software.

Asset production led to asset utilization. The years playing games I've heard of the tooling used to put the work into motion. Fusion, GameMaker, Godot, Unreal, and Unity were among the tools publically available. The time came to build my project and I decided that Unity would be the smoothest experience; it provided me with the ability to rapidly prototype games and found it to have a similar workflow experience to Photoshop along with a massive community available for support.

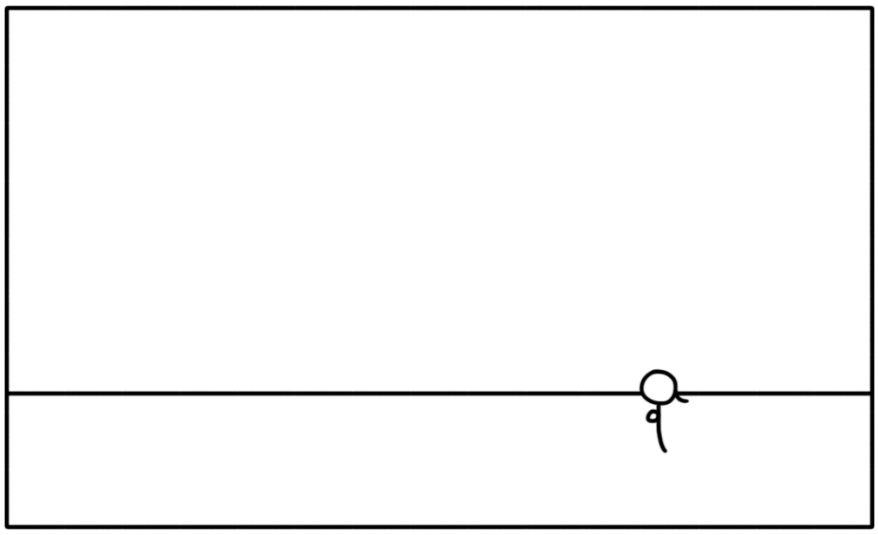
Covered front-to-end, the time came to develop my senior project.

Idea:

Since years before beginning the project, it became obvious that I lacked the time and energy to produce the 3D game I desired, I chose the route of 2D. Tabletop never appealed to me and with what little was left with the style, the platformer/role-playing genre was chosen.

History shows I've spent hours playing the style of game I so much desired to make. It was natural that I would finally decide to build a world of my own in a reasonable timeframe that would offer the experience I was looking for using the tools I felt comfortable with. Simplicity was always sexy to me; my passion of modularity and templatization drove me to create a minimal experience that provided all the necessities

such as the frame and everything else that wasn't non-sense – this led me to create the template below.



There are subtleties most eyes would not catch. Obsession with numbers, standardization, and future-proofing let me to choose the 16:9 aspect ratio. This is the standard ratio to display images in the 1080p resolution and practically what you will find on every workstation over the past fifteen years. The beauty of this ratio is that it can be displayed on a 4K resolution without ruining the artist's initial interpretation; hence, why I chose to harden my dimensions in it.

The ground line is my basis for level; it is easy to build upon and provides a visually appealing "center-of-action" that the eye can float to. It allows for content to easily float on the playing field or be set and act as background matter.

The character you see pictured is the female avatar; an option to select the opposite gender is possible with a liberal selection of tones to satisfy the users desire. While aiming for simplicity, I wanted figures that could be represented minimally. The actor walking, running, and jumping across the screen do not demand additional animations or components; when one sees it in action, it is simple enough to provide the desired immersion I was seeking.

The style you see in the figure date back to the time I was in fifth grade drawing scenes on 8.5x11 printer paper. I felt comfortable feeling that I could execute what I wanted to get across through years of playing with pen, paper, and ruler. Scenes such as lighthouses and downtown main street stretches were easy to implement and understood - it doesn't take much to glance at an image and envision a poetic scene between actors through to a battle going on with stick figure armies.

Game Mechanics:

The game makes use of Unity's built in physics to control everything in the environment; through the use of variable implemented in the objects implemented in the physics, key presses can change the velocity of the referenced object. A key pressed and in regards to the appropriate conditions will allow the actor to transform across the stage to visually display motion.

A simplified game such as Stickout will require the functionality to interact with other non-playable characters; this is offered through scripting for such events. Inheritance from the very root of an object allows for implementation of behaviors that revolve around predicable (or otherwise) behavior. The use of a tagging system in addition with colliders will take into account who is to be considered useful or not – for example, say you are holding a weapon of any sorts – be it a spoon or an axe; you may utilize this at any moment, but there must be a safe guard that will prevent you from causing mayhem (this is easily implemented through colliders that will be ignored on contact given a situation).

There is an option for the player to customize their avatar; this is done using serialization using SQlite – this is also tied in to the player's inventory which allows for saving of the consumables, quest items, and weapons. Lists were used as the primary container due to their versatility for nearly any component that made use of a collection.

Canvas was an issue since I couldn't seem to properly scale it, so I had to spend time working inside of a large box while the actual game scene was a tiny spec on the screen. This was a frequent issue I encountered throughout the community and simply must adapt to.

The project has adapted over time to utilize the available features of the tool. Coroutines have found their place as make-sense in the artificial intelligence to better control the behavior.

Time Management:

A professor introduced the idea of "analysis paralysis" when I was taking the special topics course "Software Engineering". It has been an idea that has stuck with me ever since and unfortunately I fall victim to it with nearly every software endeavor I take.

A majority of the time was dedicated to considerations scribbled into notebooks with ideas regarding the project - details as to the mechanics to lists of planned assets fill the pages.

I suffer a fear of programming that didn't help with putting key-to-code. "What if I didn't figure it all out" to "what if there is a better way to do this?" Actually starting on the code and implementing all the methods was my fear. The acknowledgement of lack of time to comprehend and understand design patterns put me in my own little corner. I was trying to figure out the mechanics of how the combat, interaction, and leveling systems worked; needless to say, it was complicated.

Aside from the mechanics, there was the preference for the visual. My view is that once something is seen on the screen, the ideas can begin to flow in regards to the code. A scene can ignite the imagination and stimulate the growth necessary to bring life to the situation at hand.

My preferences is construction assets through graphics applications – creating detailed stages. There is difficulty implementing environments to unique and alive - complexity grows from coding a door to walking up stairs or grasping a ledge.

Considerations:

There are big ideas for the game. The code I've come up with is absolutely mediocre and I intend on tossing most of it out. The classes I've come up with are my templates for expansion. I've managed to implement on screen what looks like the bare bones of a game with much potential; the project itself is very much open ended.

When I played with the name, I originally looked at the stick figures, and knew VR (virtual reality) needed to be implemented – this led to the title of "Stickout". I originally wanted to implement this feature into my senior project, but through time constraints and complexity, I quickly backed off and sidelined it; besides never seeing 2D virtual reality implemented, there was the draw of luring an audience in who might not get motion sick through wearing such a unit. Depth was the feature I wanted to implement and along with that, horizontal head-tracking which would allow the user to look over and use where they are looking as the view. Imagine a 3x16:9 setting and the actor is equipped with a sniper rifle, should I reserve two directional keys for scrolling to see if a target is approaching from the sides or would it be more ergonomic to glance over?

The idea that has struck me in the past is commissionable content; being the sole developer, I have full control of everything that goes into the mainline version. If there is a community following this game with a desire for new content, there should not be an issue starting a repository to enable growth for the project. I truly believe that if the material was made public, there would be a small section of enthusiastic followers who would consider dedicating portions of time to build upon the work.

Conclusion:

I've taken some time to fix the mistakes from the past and make progress with this project. I learned that with motivation, realistic projects can be completed. C# was a language that I've only briefly played with before the project. The right tools for the right job make it easy, but if you don't actually give it any throttle, it doesn't move.

If I could have done anything differently, I would have done something else completely. I am not against game development, but this project was conceived during a portion of my life which was happier. I've spent the past few years figuring out where I could be more useful and this project isn't what I feel makes best use of my talents.

The software I used were Adobe Photoshop CC2014, Unity 2019.2.18f1, Visual Studio 2019, and Sublime Text 3. The use of a Wacom pen and tablet let me rapidly and with ease create the assets in Photoshop. Unity tied-up with C# were comfortable environments for actually putting the meat on the bones.

My software is organized into folder structures. Assets are divided into the following catagories: animations, characters, items, scenes, scripts, sounds, stages, and UI.

The hardest part of the project was actually beginning the coding; where to start and what to work on that wouldn't break other parts of the game. I was very ambitious at the beginning and I wanted to have a complete and marketable game by the end and above all, the virtual reality view implemented, but real life challenges, constraints, and struggles kept me from dedicating as much time as I wanted to towards the project.

There aren't any complex structures or algorithms, but the very basics and what I envisioned, a foundation of a game. What I have to present is code that is gets what it needs to get done, done.

The project was harder than I'd imagined, but this is especially the truth when it comes to solely developing a game.

I do hope that when the time comes to present, the commitment to the vision is what shines. It's really intended to be a game for all audiences who want a simple and engaging experience; the assets are easy to work with and expand upon. It's a "pocket-sized game" with the ability to allow casual developers to easily integrate their own worlds and stories. The future of Stickout is in fact to open up to developers because there is only so much I can do; community creativity is much welcomed and encouraged. I hope when people see the game, they can see it as a tool which should be a natural progression from the old days of storyboarding with pen and paper and allow them to easily integrate their ideas into a system with a beating heart.