Progeny Prototypes

ALL PROTOTYPES ARE FOUND IN “Scenes” FOLDER

Character movement, camera follow, and weapon system can be experienced in the following scenes
Assets/Scenes/motionPrototype.unity
Operations of the following three prototypes are as follows.
Shoot: Use J or left mouse button
Move: A to the left, D to the right
Jump: Space or W
Crouch: S
Switching weapons: Q / E
Reload: R

Character Movement - motionPrototype

What are you trying to achieve?
The aim of this prototype is not only to create a movement mechanism but also to optimise the feel of the controls during gameplay.
Who worked on the prototype?
- Ruiyi

How long the prototype took to complete?
This takes about 2 days.
What decisions did you make based on the prototype? What worked, what didn't?
I have implemented a version of the prototype that includes left/right movement (including turning to the direction of movement), jumping (we decided it didn’t feel right for the game that we are trying to create. Instead we will develop a climbing feature where the user can climb obstacles) and crouching. When multiple operations are introduced, there may be a lot of collisions that we cannot expect.

Decision one: Is it allowed to crouch when jumping in the air? Or if you are only allowed to jump twice in the air to get over the higher obstacle? I have implemented both versions of the prototypes and finally chose the first one. That is, it is allowed to crouch, folding your left in the air to get over an even higher obstacle.

Decision two: Check the floor and ceiling. This is about when the player can jump and crouch. It can be assumed that the player can only jump when he is on the ground and can only stand up (after crouching) when there is nothing above his head. And here I learned several methods to

do the ground/ceiling check. In the end, I chose a more direct one, which is just to add foot and head colliders (check is triggered) and to see if any object in the ground layer overlaps.

Decision three: Animation. I try to use a simple state machine to express the jump and crouch animation. But it does not work as I expected. Crouching takes time, and if during the crouching duration the player releases the crouch key, he is now required to stand up in the same animation frame, or if just playing the idle animation, it can be much stiff. So I decided to use a blend tree to generate the middle frame of the crouch and idle animations.

What questions/prototypes arose from this one?
The biggest question here might be art and operation, in our game the character is pixel art, which means discreet, while the blend tree needs continuous parameters. It can be very strange in the future when adding the sprite.

Camera follows player - motionPrototype

What are you trying to achieve?
We want to implement a camera module that can follow the character naturally and with a low presence. At the same time, it gives a sufficient field of view for the player to provide space to operate.
Who worked on the prototype?

- Ruiyi

How long the prototype took to complete?
This takes about 2 hours.
What decisions did you make based on the prototype? What worked, what didn't?
Step 1: The first step is the implementation of the fixed camera, which unity has already done for us.

Step 2: When we realised we needed camera following, the first thing we wrote was a way to set the camera coordinates directly to the character coordinates.

Step 3: Directly binding the camera to the character coordinates is a bit rigid, and our characters are on the screen, often moving in one direction, the front view is more important than the rear. In this part, we directly put the camera in front of the character at a specified offset position. It works but there are still problems, for example, the camera should be offset to the left when the character turns around.

Step 4: Smooth the camera. This step makes a distinction between the current position of the camera and the target position. The target position of the camera is the coordinate that has a certain offset relative to the front of the character in the third step. The current position of the camera needs to chase the target position of the camera within a certain speed limit, which also achieves the purpose of smoothly transporting the mirror.

What questions/prototypes arose from this one?

When the character receives an attack, the screen may vibrate. Or other more visually appealing effects. This may need to be written in conjunction with the combat system.

Weapon System - motionPrototype

What are you trying to achieve?
We want players can switch firearms and use different weapons. Weapons can be discarded or picked up, and each weapon can have different ways of operation and special effects. The current weapon system only implements the switching of firearms and the difference between different types of firearms/bullets.
Who worked on the prototype?

- Ruiyi

How long the prototype took to complete?
Currently, only two days to develop, and it may take a week after the completion of development.
What decisions did you make based on the prototype? What worked, and what didn't?
There are actually trade-offs to be made about the shooting style. When I really played the game, I found that the left-handed operation is complicated enough, if the right hand also needs to use the mouse to refine the aiming, then I will probably have my hands full. So the weapon I've implemented is for now a version that doesn't require the right hand to aim, and bullets are currently fired directly from the player's face (i.e. to the left or right). In order for the horizontally oriented bullets to hit the air target, we need to be able to jump up and shoot in the air.
Of course, my teammates want to implemented another version where bullets can be fired directly at the mouse. This should be modified in subsequent versions, and this one cuts out the jumping function and replaces it with climbing.
What questions/prototypes arose from this one?
The weapon system must support the use of a variety of weapons, and here I would like to be able to abstract the operation of weapons into four phases: preparation (aiming/charging to increase power and accuracy), attack (firing and maybe including the moment of throwing a grenade bazooka), reloading and ammunition replenishment. However, the display of the aiming effect is still open to discussion.
Players will be able to use more than just weapons after the expansion of the weapon system, which can resemble a kind of backpack.

Enemy AI - enemyAIPrototype

What are you trying to achieve?
Enemies that feel fun and engaging to play against, which have different states of behaviour which are triggered by certain events and conditions. Each unique enemy type should have a key gimmick which makes the player's strategy different.

Ground Enemy State List
1. Idle (Standing still)
2. Approach (player has been found, walks towards player until in range of attack)
3. Prepare (preparing to pounce attack)
4. Pounce (jumps in a parabolic motion that player will be able to dodge by crouching)
The enemy then goes back to the Approach state, regardless of how far away the player is.

Flying Enemy State List
1. Idle (Stays stationary in the air, open for this to be dealt with differently)
2. Approach (When the player gets in range it starts flying horizontally towards the player)
3. Prepare (Waits for a very short moment so an attack animation can be played)
4. Attack (Throws down a projectile)
5. Wait (Waits for a very short moment so a post attack animation can be played)
6. Return (Returns to its original position, in game it could collect a projectile from here again)
Unlike the ground enemy, the flying enemy returns to Idle unless it is still within the Approach range.

Who worked on the prototype?
- Oliver

How long the prototype took to complete.
2 hours for ground AI, 1 hour for flying AI. Making the ground AI provided a good interface for the flying AI so development time was massively improved.

What decisions did you make based on the prototype? What worked, what didn't?
Initially when making the initial ground AI the AI got very close to the player and only did a very small jump. Doing an exaggerated larger jump made the AI feel more fun while giving the player the same amount of time to prepare as it is further away but faster. In game the player will be able to dodge this jump using the crouch from the other prototype.
I also used an enum system to set the state of the enemy. I felt this worked very well as it made the code encapsulated into different behaviours, and easy to read.
The decision was made for the flying enemy to have a slightly different behaviour than the ground one so each enemy type had a different approach and did not just feel like rehashes of the same enemy. This led to the idea of having the flying enemy return to its original position, which when animated will represent picking up another projectile to fire at the player. This makes each enemy fun to play against.

What questions/prototypes arose from this one?
A question would be to consider making even more states of the AI. Currently the AI somewhat loops their states and it does not feel as dynamic as it could be, as it is very cyclic in nature.

Perhaps the AI could be adjusted so that there is more variability in their actions and they have chances to do actions such as special attacks or retreat.

The combat system and crouch system were also not implemented in this prototype so it will be interesting to see in development what playing against the AI is actually like, rather than just seeing the actions of the AI.

Player Aiming and Shooting - combatPrototype

What are you trying to achieve?
Intuitive aiming and shooting controls that still meet the needs of the game's combat system by providing 360-directional shooting capabilities.
Who worked on the prototype?

- Jack

How long the prototype took to complete.
The circum-rotational aiming took approximately 2-3 hours and the projectile shooting took closer to 30 minutes.
What decisions did you make based on the prototype? What worked, what didn't?
Due to a miscommunication, our team prototyped a horizontal-only shooting system as well as the omnidirectional aiming which fortunately gave us the opportunity to compare the two. We ultimately decided on the 360-degree model as we planned on having flying enemy types and other targets at varying heights. This choice also requires more input from the player, to aim with the mouse cursor, which results in a more immersive and fun combat system.
What questions/prototypes arose from this one?
The singular straight-line bullet path posed the question of whether or not a multi-shot weapon such as a shotgun would benefit the game without being too overpowered.
The simple-enough point and click shooting mechanics raised the question of whether the game would benefit from some more complex combat mechanics such as throwable objects, explosions, or environmental interactions.

Health and Damage - combatPrototype

What are you trying to achieve?
A basic health and damage system in order to test combat mechanics which can be tweaked and/or expanded on later on in production if necessary.
Who worked on the prototype?

- Jack

How long the prototype took to complete.
Player bullet collisions took approximately 30 minutes.

Enemy melee collisions took approximately 1 hour of testing different solutions, but only about 30 minutes of that was the final solution.
The health system took less than 30 minutes as they were just floats to be subtracted from.

What decisions did you make based on the prototype? What worked, what didn't?

Right from the beginning I knew that the health system would be a simple float which gets subtracted from when taking damage and can be added to in the case we add healing.
For player bullet collisions I stuck to what I was familiar with and used trigger-enabled box colliders.
For enemy melee collisions, I tried a couple of different solutions: making the entire enemy's box collider inflict damage on the player; the same solution, but requiring the enemy to be facing towards the player to inflict damage; but I ultimately decided to go with the simple solution of creating a child object on the front of the enemy which triggers the damage.

What questions/prototypes arose from this one?

As the enemy in this prototype was without pathfinding, the enemy's method of inflicting damage was mostly based off what should work rather than what does work. So, naturally the question arose of whether or not this solution would actually work with enemy pathfinding and attack animations.

Checkpoints and Respawns - checkpointPrototype

What are you trying to achieve?

A clean solution to respawning back at the start of the scene after having progressed through a large number of enemies and obstacles. This should solve the (hypothetical) problem of levels becoming tedious and/or annoying to play through after having already done so.

Who worked on the prototype?

- Jack

How long the prototype took to complete.

Approximately 2 hours.

What decisions did you make based on the prototype? What worked, what didn't?

Having the player respawn into the exact same scene (obviously) didn't work well as the player could come up against already weakened enemies or even be "spawn-trapped" by enemies. Instead, having the scene manager reload the scene and then move the player to the last checkpoint they walked over seemed far more fitting and allows us, the designers, to orchestrate the level and decide when the player approaches which enemies.
While creating the checkpoint system, I tested announcing/displaying when a checkpoint has been reached to the player, but it didn't align well with the gritty movement system, which is how the player progresses through the levels from checkpoint to checkpoint. Nor would it fit the minimalistic UI design we had proposed, so I ultimately decided against it and had created "silent" checkpoints which would operate without the player's knowledge.

What questions/prototypes arose from this one?
While the prototype level for reaching checkpoints and respawning was very simple, the placement of checkpoints did raise the question of whether some levels would benefit from having less checkpoints than others in order to raise the difficulty.

Flashback Tiles - flashbackPrototype

What are you trying to achieve?
When the player stands on a flashback tile (in the prototype they are a different colour from the others), the background and foreground textures change to show a flashback. It is supposed to support and help tell the player the story without any dialogue by showing that this is a memory the player is remembering.

Who worked on the prototype?
- Poppy

How long the prototype took to complete.
Excluding the time creating the textures for both the current time and flashback, maybe an hour.

What decisions did you make based on the prototype? What worked, what didn't?
At first I removed a tile from the main tilemap and then replaced the missing tile with a new tile map (which was only one tile) for the flashback tile to give it the trigger. It visually worked but if the player stayed standing on the flashback tile they would fall through the floor.

So instead I kept the original tile map (no missing tile for the flashback tile) and added the flashmap tilemap on top of the original one (so there are two tiles on top of each other, the original and the flashback tile) and it still worked as the trigger and the player no longer fell through the floor.

The original size of the box collider for the player (which is the same size as the player's width) meant that the flashback tile was triggered sometimes when the players feet weren't directly on top of the flashback tile. So for the prototype I made the box closer to the size of the player's feet.

What questions/prototypes arose from this one?
A question could be to think about whether we need to adjust the size of the players box collider to make walking on the tile more believable, however, in the actual game the flashback tiles will look the same as the floor (so they are a surprise when you come across them) so it may not affect the actual gameplay because the player cannot see the tile.

Parallax - tilesPrototype

What are you trying to achieve?
A parallax effect when the player is walking to make the game look more dynamic.

Who worked on the prototype?
- Poppy

How long the prototype took to complete.
About 30 minutes.

What decisions did you make based on the prototype? What worked, what didn't?
Was able to determine how far from the camera each element should be in order to look natural. Anything in front of (and including) the buildings will have no parallax feature as we don't want them to move like the background does.

What questions/prototypes arose from this one?
No questions or prototypes. Was tested and behaved like expected.

Cutscenes - cutscenePrototype

What are you trying to achieve?
When the player stands on a cutscene tile it will activate a cutscene and stop the player from moving until the cutscene is completed.

Who worked on the prototype?
- Poppy

How long the prototype took to complete.
Around 3-4 hours.

What decisions did you make based on the prototype? What worked, what didn't?
Originally I tried implementing it by making the cutscene a separate scene, and when the player stands on the tile it would cut to that scene and then once done it would jump back to where the player was on the original scene. I found that to be tricky so I included the cutscene in the same scene and had the player movement turned off for the duration of the cutscene and then turned on again. The player standing on the tile triggers the StartCutScene method to be called which stops the player's movement and plays the cutscene. I also added a boolean to the tile so that

once the cutscene has played once the tile cannot be triggered to start the cutscene a second time. I used a signal emitter at the end of the cutscene to call the EndCutScene method which allows the player to move again, and in the prototype activates the thought bubble.

What questions/prototypes arose from this one?
How long are the cutscenes able to before the player gets bored? We are utilising cutscenes to help tell the story and make the audience care more about what happens to the main character. How often can we use cutscenes without making them repetitive is also another question we need to think about later during production.

Story Components - cutscenePrototype

What are you trying to achieve?
When the player stands on a story tile the player movement is stopped while the player reads and cycles through the story textboxes until the text is finished and the player is able to move again.

Who worked on the prototype?
- Poppy

How long the prototype took to complete.
1-2 hours.

What decisions did you make based on the prototype? What worked, what didn't?
Was able to determine the size of the textbox to display the story which felt the best as well as using a script that gets given a text file to determine what text shows on each slide of the text box. I tried having the return key to skip the text, but someone testing the prototype mentioned that it felt like space was the more natural way to skip story text so I changed the key to skip the text to space. Later in production we will probably have several options that will skip the text, i.e., space, return or mouse click. Once the story text has been read a boolean is used so that standing on the tile again will not trigger the story text a second time.

What questions/prototypes arose from this one?
Something we need to think about is how much information/text can we put in a story dialogue before the player loses interest and stops reading the text. We need to make sure to include all the important information to the story at a time and in a way that the player is interested in what is being said. Too much text might scare the audience from reading it all as it may seem overwhelming.

Climbing Objects - climbingPrototype

What are you trying to achieve?
A smooth way to climb on top of climbable ledges.
Who worked on the prototype?

- Jack

How long the prototype took to complete.
Approximately 6-8 hours after trying each solution; a lot longer than it needed to take.
What decisions did you make based on the prototype? What worked, what didn't?
To begin with, I tried to implement climbing by using box colliders on empty child objects of a ledge because I was most familiar with working with box colliders, however I ran into a lot of issues with this method.
After doing some research, I began trying to implement a raycasting solution but this, once again, ran into a lot of issues and this time I was way out of my comfort zone to try and debug them.
Finally, I found the solution of having two boxes float in front of the player, one around eye level, and another above where any climbable ledge would extend. Checking whether the bottom box overlapped with a “ledge” layer object while the top box did not, meant the player was in a suitable position to climb said ledge and would be teleported on top of the ledge.
I decided not to code a smooth transition for the climb while still in prototype as it would probably take too long and teleporting is fine for the purpose of the prototype.
What questions/prototypes arose from this one?
While coding the climbing mechanic, the question arose of how much interaction is required from the player. For example, should the player have to hold onto the climbing button throughout the entire animation in order to complete the climb?, or should they have to press once to grab the ledge and again to pull themselves up?, or should it be a one-and-done control scheme. There was not enough time to prototype these different climbing interactions, although this is definitely something I intend to test while integrating the mechanic during production.
Through this process, I also realised it would be best for our player’s actions/states to interact with a finite state machine in order to ensure the player isn't doing too many actions at once. For example, in the final game, we would not want the player to be able to climb, crouch, and reload all at the same time.

Thought Bubbles - cutsceneProtoype

What are you trying to achieve?
At certain points in the game whether it be to help with the story or to hint the player in the direction they should be heading a thought bubble can appear above the player to show the player's inner monologue. It follows the player and then after a while it disappears.

Who worked on the prototype?

- Poppy

How long the prototype took to complete.
I had already done the story text prototype so this one was much faster, around 1 hour.

What decisions did you make based on the prototype? What worked, what didn't?
Decided that the time for the thought bubble would depend on the length of text, but in the prototype I used a short line and found 3 seconds to be a sufficient time. At first I used the player's velocity to move the bubble, but then found that if the player runs into a barrier the bubble continues to move in the direction the player is trying to move in. Instead I made the bubbles position equal to the players with an x offset.

What questions/prototypes arose from this one?
One thing we need to think about closer to production is how we can use these thought bubbles to keep the player on track. If the player is heading further and further in the wrong direction, or has not realised that there is an interactable object nearby, how can we use this element to hint the player to what they are supposed to be doing?

Tilemap/Stairs - tilePrototype

What are you trying to achieve?
Creating a tile map for the level to allow the player to roam freely, while remaining is the bounds of the level. Some of the tiles we are using are not squares so we needed to make sure that the player is able to walk over all these tiles as expected.

Who worked on the prototype?
- Poppy

How long the prototype took to complete.
Around an hour.

What decisions did you make based on the prototype? What worked, what didn't?
The default physics shape for the tiles that were not squares was not very accurate so we needed to make custom physics shapes in the sprite editor. This means that the player is able to walk up/down stairs, holes, etc without getting stuck on any tiles. The player's mass/gravity scale also needed to be increased to prevent the player from going up and down the stairs too quickly and looking unrealistic.

What questions/prototypes arose from this one?
A question that this prototype prompted was, should we animate the player differently when travelling on an incline? This is something we can think about closer to production and determine whether or not the normal animation of the walking character looks weird while going up or down an incline.

Interactable Objects - tilePrototype

What are you trying to achieve?
Having the ability for the player to interact with objects in the scene by pressing E.

Who worked on the prototype?
- Poppy

How long the prototype took to complete.
30 minutes.

What decisions did you make based on the prototype? What worked, what didn't?
Decided on a range that feels close enough to the object to naturally interact with it.

What questions/prototypes arose from this one?
How we should alert the player that there is an interactable object is range. There are many ways we could do this and after we get other people to play our game we can determine which is the option that people prefer the most.

Sound Prototype - soundPrototype

What are you trying to achieve?
Composing music that feels dark, yet nostalgic. The game should have a solitary feel that makes the player feel like they are isolated, with only enemies and a monster deep in the distance keeping them company.

The prototype also achieved experimenting with spatial and dynamic in-game audio with sound effects by playing sound effects on in-game audio.

Who worked on the prototype?
Oliver

How long the prototype took to complete.
1 and a half hours to compose the prototype background music, 5 minutes to make the prototype jingle being played by the shop object.

What decisions did you make based on the prototype? What worked, what didn't?
At one point I experimented with the pitch function on Unity. I felt this massively improved the background music that I had created and the slowed down, reduced pitch song fitted with the dark and ominous nature that the game has, and most of all, created an unsettling feeling that something was wrong.

When experimenting with the volume rolloff for the sound effect, I initially tried the logarithmic rolloff curve, but under these settings the volume never really faded out, meaning you could always hear a sound effect even if the emitter was off screen and very far away. For this reason I ended up deciding that linear rolloff will be the best implementation for our game.

The footsteps audio could definitely be implemented better, in order to improve this it would either be completely removed so players can focus on the sound of the background music and other objects in the game, or the footsteps would alter slightly upon each play so it sounds more natural than playing the same audio file over and over.

What questions/prototypes arose from this one?
A question that had arisen from this one is the use of bass and drums in the music. They give the music a more rhythmic feel, and turn the song into a hard hitting beat. However, there could be a possible loss for the original intent of the emptiness and the sparseness that you can hear when it is only the chords playing. Because of this, it might be more fitting to only play the bass and the drum elements whenever you are in combat. Perhaps a system could be implemented where these are played whenever enemies appear, and go away after you kill them, so players feel more excited when fighting monsters, and more solemn when alone.

How the results of our prototyping have altered/confirmed your project plan. What aspects of the project are still unknowns?

When constructing our Game Design Document and creating our project plan, we were originally a little bit sceptical as to whether we were able to implement all the features that we wanted to do. However, after this week of prototyping, many features were much easier to implement than anticipated, and has made us very excited to be able to move onto the development phase where we can start merging our ideas together to create a more cohesive experience. We are also very happy with how our prototypes have turned out, and we believe that combining all of our combat mechanics together will create a very fun game. Further integrating these mechanics with the story aspects that we were able to prototype will create an overall immersive and fun experience.

Some aspects that have altered our plan have been:

- Adding crouching
    - Originally, we did not anticipate adding this feature, but Ruiyi ended up adding it to the movement prototype and all the group members agreed it would be a cool feature to add to the game. This made us also come up with the idea of having the standard ground enemies pounce, but if you timed your crouch right you can dodge them.
- Flying enemy type

- The flying enemy type was originally designed to just fly around in the sky and drop projectiles, however it was changed so after it drops it flies back to a certain location in order to collect another projectile to shoot, in order to distinguish it better from the ground enemies' behaviours.

An aspect of the project that is still unknown is how the enemy waves will be implemented. We are not sure if we will have a GameObject that manages and spawns the waves at certain locations or if we will manually put in instances of the enemy prefabs at certain locations. This could also be level dependent.
Another unknown aspect of our game is environmental interactions. The opportunity to outwit enemies with the environment excited us a lot but we did not focus on this aspect of the game while prototyping as it is a feature that would be very nice to have, but not necessary to the gameplay.

Another aspect we are unsure of is the GUI. The cutscene prototype featured the use of textboxes and speech from the player, however, we are still uncertain as to what approach we will implement the GUI for menus and in game. Our goal is to have a minimalistic GUI that doesn't distract the player too much from the game, so the player can have an immersive experience.

We are also unsure on how the menu system will work. None of our prototypes featured a menu system, so we will have to determine this in the production stage. However, this feature is not one that needs to be implemented immediately and will likely be fully implemented towards the end of the development stage when the game is being polished.