

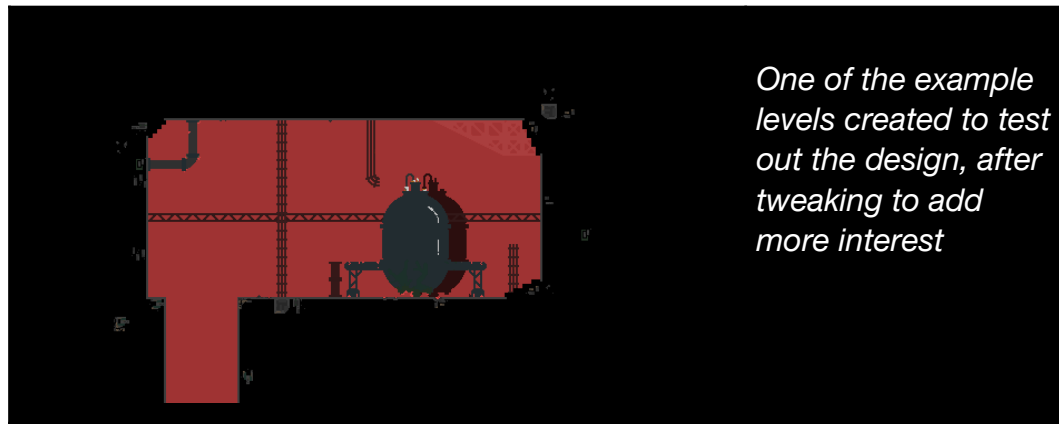


Patrick Bell - Ascendant - Logbook

Date	Work Completed	References / Links
7/03/2021 2 hours	I researched possible game concepts. Starting off with the game mechanics, it would be easiest to write a platforming game due to the inbuilt rectangle collision of Pygame. In researching I considered various inspirations from games I have played. Following the success of Hollow Knight it seems that metroidvania platformers with tight combat are rising in popularity. I also considered the setting of this possible game, due to low asset requirements it would be easiest to set the game underground, meaning most of the level design can be blank space. For this reason it would be best to create a platformer with tight controls and combat, set underground.	Hollow Knight - Game inspiration - https://store.steampowered.com/app/367520/Hollow_Knight/ Pygame - reference - https://www.pygame.org/docs/
9/03/2021 1 hour	<p>Achievements: Created gantt chart to plan out the project timeline</p> <p>Began researching and writing up game information, including: Core Concept; Lore and World Building; Genre and Style. In the process of researching this I discovered other possible game inspirations, Dead Cells is a great procedural platformer whose visual style is a good inspiration for the game. Celeste has great tile design and animations to add interest to levels and Super Meat Boy's tiles are simple to create and largely match the setting. I have decided the game will be set in an abandoned underground factory.</p> <p>Challenges: It is difficult to decide on elements of the game information such as its genre and style so far ahead of the development. To address this challenge lots of research is needed into other games similar to your vision.</p>	<p>Dead Cells - Visual Inspiration - https://store.steampowered.com/app/588650/Dead_Cells/</p> <p>Super Meat Boy - Visual Inspiration - https://store.steampowered.com/app/40800/Super_Meat_Boy/</p> <p>Celeste - Animation and Level Design Inspiration - https://store.steampowered.com/app/504230/Celeste/</p>
12/03/2021 1 hour 20 mins	<p>Achievements: Created the system development approach justification</p> <p>I made close reference to the Davis textbook and the classroom slides. I chose the RAD approach for many reasons, key of which are the project's short time frame, limited budget, and client that isn't the creator (excluding the End User approach).</p> <p>Achievements: Finished creating the story and setting for the game and writing these sections of the game information</p>	<p>Davis Textbook - Reference for System Development Approach Justification - https://drive.google.com/open?id=1o2IUtAzLETv01Pde7qYvXEJnZv2oV2y0&authuser=0</p>

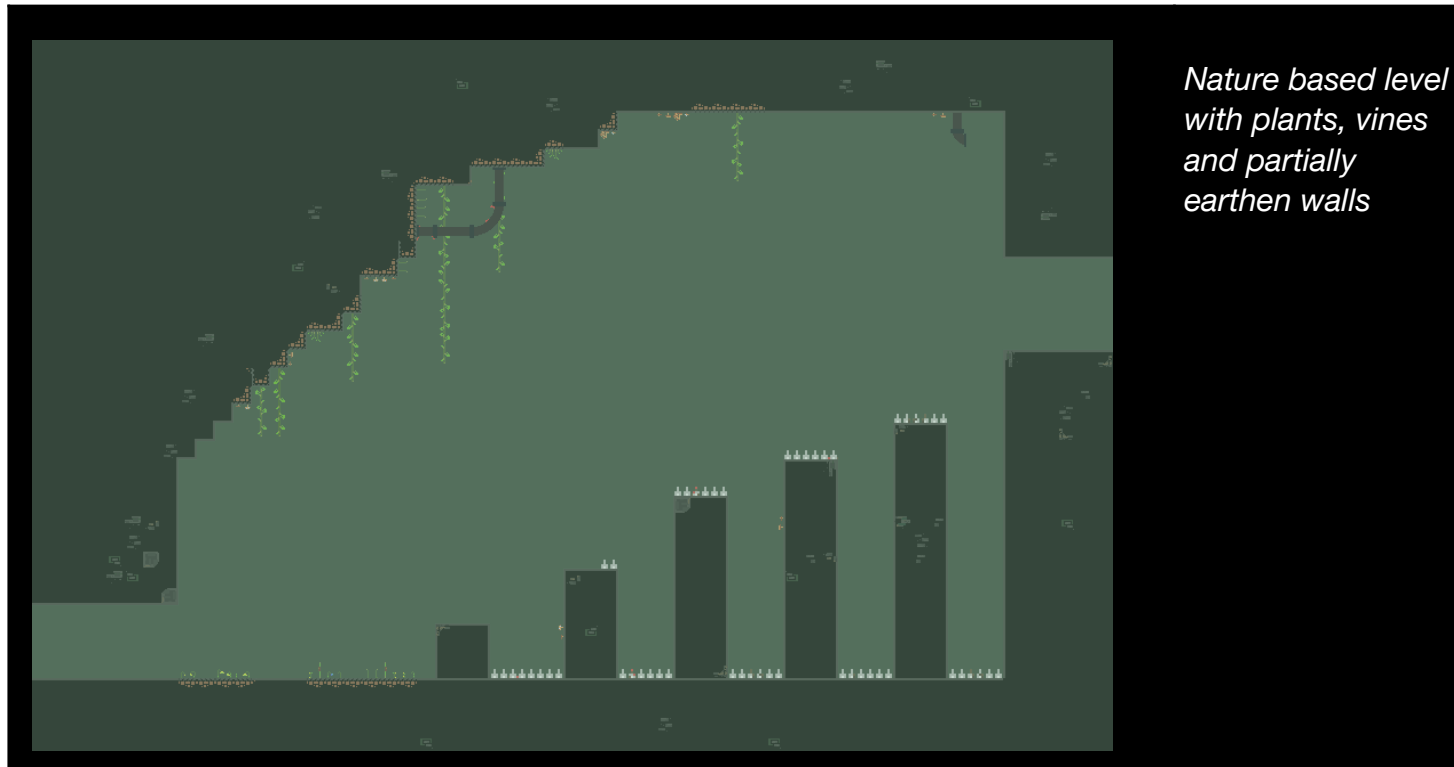
	<p>Finished creating the story and setting for the game. I am a large reader of dystopian fiction so the story is based around a future without humans and set within an abandoned factory deep underground, drawing inspiration from the game Portal. The story grew as a result of the project's restricted time frame, the underground setting makes level and tile design easier and the post human world makes both enemy design and AI easier. To justify the players and enemies existence long after humans I introduced a magical component to the factory, it used a magical red goo found underground to make robotic workers. Since your character is unlike the enemies, I decided that the main character would be made from the natural world while the enemy would be made from human parts. I called the game “Ascendant” because the character is moving from underground to the surface, since there are no humans on the surface it must be reclaimed by nature, so I made the main character based around a flower explaining their desire to get to the surface.</p> <p>After asking Mr Dunne to look over my Gantt I made some minor changes with the ordering of the rows.</p>	
17/03/2021 1 hour 20 mins	<p>Achievements: Finished writing the game information section, adding the game mechanic, audio and visual design and inspiration. Added an animated GIF to the document to show the animation style of Celeste. Celeste’s animation style is very fluid and adds lots of interest to tiles, even if the tiles are simple and repeating, this improves the visuals significantly with little development cost. Polished and corrected elements of task A, including the game information section, system development approach justification, gantt chart and logbook. Added references to the lore inspirations, including Metroid, Castlevania, Portal, Cradle and Brave New World.</p>	
19/03/2021 1 hour	<p>Final polishing step on part A documentation, checking for any grammar or spelling errors, and verifying all elements of assignment criteria are met. Submitted Part A pdf, unfortunately the Celeste animated GIF didn’t play in the exported pdf.</p>	
2/04/2021 3 hours	<p>Achievements: Finished the first draft of the structure chart using Diagrams.net and Mr Dunne’s provided symbols, consulting the Davis textbook to ensure correct format and symbol usage.</p> <p>Challenges: It is difficult to create the chart without having written the program because it makes you think ahead, anticipating the structure of the program and the flow of data.</p> <p>Did some searching for visual assets, mainly searching on https://opengameart.org which is great because all the assets are creative commons. I found this asset which fits the aesthetic of an underground factory. The sprite sheet includes tanks and pipes which will be useful for showing the “magic goo” travelling throughout the factory. The asset is just silhouettes so it doesn’t communicate the goo or setting properly so some color is definitely required.</p>	<p>Davis Textbook - Reference for structure chart - https://drive.google.com/open?id=1o2IUtAzLETV01Pde7qYvXEJnZv2oV2y0&authuser=0</p> <p>Structure Chart Symbols - https://classroom.google.com/u/0/c/MTYzNzM0NTQ1Mzk0/m/Mjk4MDE3NTg3MTg4/details</p>

	<div data-bbox="669 33 1294 456" data-label="Image">  <p data-bbox="680 368 1211 443"><i>An example scene for the visual asset from opengameart.org</i></p> </div>	<p data-bbox="1877 41 2213 193">Visual Asset - https://opengameart.org/content/minimal-industrial-tiles</p>
<p data-bbox="47 533 188 603">5/04/2021 5 hours</p>	<p data-bbox="248 533 1843 727">Achievements: Coloured the underground factory visual assets by adding in greenery and red goo variants of each tile. Furthermore since the setting is underground some edge tiles/brick tiles are required so I drew these based on my game inspirations, mainly Super Meat Boy and Celeste. I just made the colour palette up as I went which was very easy with the pixel art drawing tool I used: Aseprite because it allows colours to be indexed into a palette which can be edited on the fly.</p> <div data-bbox="792 767 1346 1075" data-label="Image">  <div data-bbox="801 948 1337 1062"> <p data-bbox="801 948 1055 1023"><i>Newly designed underground tiles</i></p> <p data-bbox="1077 948 1337 1062"><i>Celeste edge tiles used as inspiration</i></p> </div> </div>	
<p data-bbox="47 1150 188 1220">6/04/2021 2 hours</p>	<p data-bbox="248 1150 1005 1182">Achievements: Finished work on the structure chart.</p> <p data-bbox="248 1235 1827 1465">After having coloured the tiles I wanted to get a feel for their usage in a level. Using a tilemap editor, I chose Tiled because of its wide toolset and standardised file formats, I created some small demo levels to see how everything looked and afterwards made some tweaks. The industrial tanks and other large structures were looking fairly bland so a vine plant and purple flowers added some interest. Furthermore by increasing the contamination of red goo and erosion on metal objects they became more interesting, especially adding rust to the wires and bars.</p>	<p data-bbox="1877 1150 2213 1305">Aseprite - Reference Page - https://www.aseprite.org/quickref/</p> <p data-bbox="1877 1358 2213 1513">Plant Tileset which was modified - https://opengameart.org/content/flowers</p>



One of the example levels created to test out the design, after tweaking to add more interest

Since the industrial aesthetic is fairly dark and dominated by reds I wanted to add some contrasting levels so I found a [plant tileset](https://opengameart.org/content/1-bit-plant-tileset) on opengameart.org which I modified slightly and added some plants to based on another [Kenney tileset](https://opengameart.org/content/1-bit-plant-tileset). These plants and a different (green) background contrasts well with the other levels.



Nature based level with plants, vines and partially earthen walls

Kenney tileset which only one or two tiles were taken from - <https://opengameart.org/content/1-bit-plant-tileset>

<p>8/04/2021 4 hours</p>	<p>Began work on the title theme drawing inspiration from both Hollow Knights theme and the music of C418 (composer of the Minecraft soundtrack). In this vein I think the piece would be best if it was a quieter, more ambient and atmospheric piece with a simple piano melody. I have some experience with the DAW (Digital Audio Workstation) Ableton, so I used it to start off the ambient track with some chords. On https://opengameart.org I found an ambient piece which I thought fit well with the mood of the setting. Here is the Ambient Loop.</p> <p>Challenges: I am pretty much new to using Ableton so I found their support page useful for any questions I had about how to use the program.</p>	<p>Ambient Background Audio - opengameart.org - https://opengameart.org/content/dark-ambient-loop-13</p> <p>Ableton Support Page - https://forum.ableton.com/</p>
<p>11/04/2021 5 hours</p>	<p>Achievements: Recorded and edited all the sound effects for the game.</p> <p>I created a list of sound effects required for the game, these being: walk, attack, damage, death, falling, jump, landing hard, landing softly, landing in water, and walking in water. I brainstormed some ideas for ways to create these sound effects, eg. objects and sounds to record. To create the damage sounds I broke twigs and barks into the microphone. To create an attack sound I used a knife running against a sharpener and a rock. To create the death sound effect I edited one of Ableton's sound effects. To create the falling and jumping sounds I moved a jacket and crinkled it. To create the landing sounds I dropped rocks on the jacket at different heights. For both water sounds I splashed water in my sink. Each of these raw clips required editing to select the best sounds, remove background noise, normalise the audio, remove unwanted frequencies (such as applying a low pass filter to the knife sound), and add appropriate reverb.</p> <p>For the walking loop I found various sound effects on opengameart.org the best of which was this one which consists of individual clips of footsteps in the snow. To create an appropriate loop I edited the clips together and applied noise reduction to the sound of snow, allowing the effect to better suit the underground setting.</p> <p>To perform all the audio editing I used audacity.</p>	<p>Footstep sounds - https://opengameart.org/content/42-snow-and-gravel-footsteps</p> <p>Audacity - Free Audio Editing Software - https://www.audacityteam.org/</p>
<p>13/04/2021 3 hours</p>	<p>Achievements: Finished creating the title theme, brought the theme together by adding in some sound effects and percussive ambiance.</p>	
<p>15/04/2021 5 hours</p>	<p>Achievements: Created enemy and player sprite sheets using Aseprite. I was inspired to create the player and first enemy by the player sprites of the game Nidhogg, using a simpler, more blocky style than the background to distinguish them. Furthermore by using a simple blue and black color scheme for the player it has high contrast with the background and enemies, who have a red and black color scheme. The first enemy is intended as a patrolling enemy, making its more human design logical. In comparison the second enemy is intended to</p>	<p>Nidhogg - Visual Inspiration for Player and Enemy1 - https://store.steampowered.com/app/94400</p>

be jumped or bounced off, so I based it off the “Jump Crystals” of Celeste, making a red diamond which reforms after it is bounced off of.



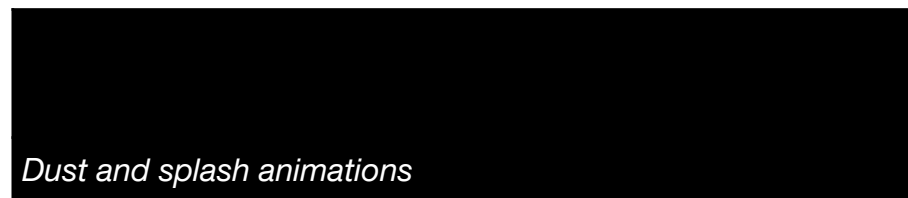
[/Nidhogg/](#)

16/04/2021
5 hours

Achievements: Created Gui visual assets and screen designs for each of the menu screens. The visual assets consist of an animated health bar, arrow, save icon, save animation, background and game logo. For the health bar I was inspired by the liquid “Soul” meter of Hollow Knight, creating a bulbous area of sloshing liquid, with a right facing bar which is changed in length by code. For the save animation I was partially inspired by the save animation of celeste, utilising the conventional symbol of a floppy disk to better communicate to audiences. In designing the menu screens I was inspired by Super Meat Boys simple plain text style and continued the pixel art visuals. Using Adobe illustrator I quickly linked together the screen designs with arrows to show the flow between menus.

17/04/2021
4 hours

Achievements: Added extra environmental tiles and began programming the rendering. After creating demo levels I decided some more variety was needed in the game and more animated tiles. I found [this tutorial](#) which while not very useful for programming, since it is based around unity, had great visual assets for water tiles and splashes. After changing the colour scheme and redrawing the bubbles and splashes I had a new animated environmental element. After creating these splashes I realised it would be great to add dust trails and puffs to the players movements and landings. After doing some research online and looking at other pixel art games I amalgamated these designs to draw two dust animations in Aseprite.




After creating these animations I got started on rendering sprite sheet animations with pygame. After doing some research I found this [code](#) snippet which loads an image from a file, converts it to be efficiently rendered

Mkdgames - 2d animated water tiles tutorial - <https://mkgames.blogspot.com/2015/05/making-water-with-simple-interactions.html>

	<p>and then splits it into frames. Based on this code I created two classes: AnimatedSprite and ImageSprite which handle rendering and advancing frames in animations. The AnimatedSprite loads a json file which describes the various animations (eg. x, y loc, frame size, num frames, time) and an image file path which the class loads into separate frame arrays for each animation. When an animation is pushed the correct image buffer is set and playback is started. The class is controlled by the render and update_animation functions which increment the frame when appropriate. Overall this works well and is fairly efficient, but to improve performance I may have to implement a dirty rectangle system rather than redrawing the whole screen every frame.</p>	
<p>18/04/2021 4 hours</p>	<p>Achievements: Programmed rectangle collision algorithm for handling collision between entities and a level. The level object contains an image background which is rendered and a precomputed array of rectangular colliders. Spawning an AnimatedSprite to represent the player we apply gravity by adding some amount to the object's velocity every frame, causing it to fall through the level. By giving the animated sprite its own custom rectangle collider we use pygame function pygame.Rect.colliderect to test collision. If the player is colliding we reset its velocity to zero and push it out of the rectangle it collided with. Comparing the top and bottom of each collider and sides of colliders then choose the smaller transformation. Reposition the character by this transformation.</p> <p>Challenges: Coming up with an algorithm was fairly difficult and while my initial algorithm worked it didn't set the velocity to zero so eventually the player would build up enough velocity to pass through the ground. To ensure all the colliders were correctly positioned I need to make a small debug overlay which could be toggled with the tab key.</p>	
<p>19/04/2021 1 hour</p>	<p>Achievements: Organised and put together all the assets for the Part B submission. Wrote up the sources for all the creative commons assets.</p> <p>Challenges: Yesterday I created a collision algorithm but this relies on a precomputed list of rectangular colliders for the environment. Initially to combat this I wrote a script in Python which loaded an image file and with a custom algorithm broke it into efficiently sized rectangles whose coordinates and sizes were output to a json file. This worked but was very slow for even slightly large levels, especially for levels with large blank spaces, since it has to loop through every pixel in these regions. For this reason I rewrote the script in C++ using this bitmap library to load the image, leading to almost instantaneous collider generation. Since this is precomputed and isn't part of the game the language restriction to Python doesn't apply.</p>	<p>Open source Bitmap Library - https://github.com/AraShPartow/bitmap</p>
<p>21/04/2021 40 mins</p>	<p>After consulting with Mr Dunne about my structure chart I added data parameters coming back from subroutines if changes were made to the data which want to be carried on. Furthermore I added a save routine and a decision for handling not updating enemies and players when paused.</p>	
<p>22/04/2021</p>	<p>Further refinement of the structure chart by positioning symbols more properly. Polished and checked logbook</p>	

1 hour 40 mins	for errors, organised file structure of submission, created zip, and submitted.	
24/04/2021 5 hours 30 minutes	<p>Achievements: Finished level loading. Wrote class which loads a level from a list of images and an entity file. The level class loads a level by opening a json file called level.json which contains a list of images, depths and parallaxes. It loads these image files, sorts them by depth and renders in reverse order so layers with low depth are rendered above. Additionally some layers should lie in front of entities so I split the renderer so negative depths were rendered behind entities and positive in front. The level class must also load all the collision and entity info from the entities.json file which is generated by the C++ script. One of these entity rects is the player's position when the level is loaded. To add visual interest I also added a simple particle system like Celestes. Particles are generated randomly across the level with color and density according to the leve.json, and given random velocities in a range from level.json. Particles just move with velocity until they leave the screen and are deleted, then new particles are generated randomly.</p> <p>Challenges: It was very tedious to constantly run the C++ script on different entity layers and manually export image layers from Tiled so I wrote a meta script which converted each tmx level file into a set of image layers and an entities.json. Furthermore loading water was difficult since it is animated unlike other parts of the level. To load water I created a class which takes a rectangle and tiles it with water tiles randomly. Unfortunately this means water can't be deeper than one tile and look good. The water class takes a similar approach to the level class, constructing a static image and list of animated objects which are rendered in two passes for in front and behind entities.</p>	<p>Reference used for making meta script run C++ exe - https://stackoverflow.com/questions/15928956/how-to-run-an-exe-file-with-the-arguments-using-python</p> <p>Reference for loading json files in python - https://docs.python.org/3/library/json.html</p>
25/04/2021 5 hours	<p>Achievements: Wrote input handling for all the actions a player can perform by passing in events to the player class. To make the controls changeable later I created a user_settings.json file which stores settings including the bindings for controls. I added the basic side slash attack when the attack button is pressed. To add jumping I just added a velocity when users initially press jump and are on the ground and then continue to add velocity for a certain time when they hold it. This means holding jump adds some extra height. To know when players are holding a button I had to add a dictionary to the player which tracks which buttons are held. To add up and down slash use the key state dictionary when the attack button is pressed. If the player is in the air and pressing down perform a down slash, if they are holding up perform an up slash. For these larger attacks the player must partially finish an attack before triggering the next to combat spamming buttons. Adding walking is fairly simple, just add a constant velocity to the player in the correct direction while the arrow keys are held.</p> <p>Since the level loader is complete and it has all the entity colliders I could add the saving input. When the player enters a level's save collider it has the ability to save the game by pressing down, which causes a sitting animation and state change. To make sure players could exit the sitting animation I had to add a case to all the other inputs which causes an unsit animation if the player is sitting. The actual saving just consists of writing the save level, and player name to a json file. When the game is started the save is loaded and the save level loaded</p>	<p>Reference for pygame input - https://www.pygame.org/docs/ref/event.html</p>

	<p>by the level class.</p> <p>Challenges: Thinking about a more event driven approach is difficult especially with timing the jump hold time. Furthermore to detect that the player is on the ground I had to add another collider which is at the player's feet and is half the player's size, since otherwise walking into a wall would allow you to jump up it. To be more generous to players I made attacks changeable part way through.</p>	
26/04/2021 2 hours	<p>Achievements: Began work on enemy classes and ai. The first enemy to do was the patrolling enemy, whose AI should be simple and predictable. The Enemy classes, like the player, inherits from AnimatedSprite and has similar basic physics and collision. When the player isn't near the enemy should patrol across the platform it stands on. To detect when the enemy should flip which direction it faces, test how close it is to the edge of the colliders it collides with, if it is closer than a certain distance flip which way it faces. When the player enters close enough, change the enemy state to alert, meaning it readies it's weapon and faces the player. If the player gets even closer the enemy attempts to attack by thrusting towards the player. Enemies are spawned by the level class from the entities file and stored in a list which is updated every frame.</p>	
28/04/2021 2 hours 30 minutes	<p>Achievements: Added second flying enemy. Rather than having a jump crystal, I decided there is going to be spike jumping, so the second enemy is going to fly towards the players to damage it. When the player isn't nearby the enemy wanders randomly within a certain distance. When the player gets close enough, to emulate a simple kind of swooping behaviour, the enemy accelerates towards the player's last position. When the enemy collides with a wall it is reflected meaning if it misses the player it will bounce back away allowing the player to escape.</p> <p>Challenges: While the attack algorithm is fairly simple it took a while to come up with one that felt good. I tried researching other methods for creating flying enemies but didn't find any useful resources. The original visual design of the enemy didn't suit this swooping design so I made a different sprite sheet. I may end up reusing the other's design elsewhere in the game, specifically I'm thinking of adding challenges if I have time and making them collectables.</p>  <p><i>Part of spritesheet for new design of flying enemies</i></p>	

<p>29/04/2021 3 hours</p>	<p>Achievements: Finished enemies and added player interaction. When the player attacks a collider is spawned in the damage area which is passed into the enemy class. The enemy class then determines whether it has been killed and changes state and plays a death animation. The enemy is then deleted from the enemy list. Knockback is then applied to the player, meaning players can down-slash off of enemies like the original jump crystal idea. The patrolling enemy's collider and weapon's collider, and the flying enemy's collider are passed to the player to determine if the player should be damaged and health is taken off appropriately. Furthermore when damaged the player flashes white. To implement this I added an optional "whitened" version of all frames when creating the AnimatedSprite. If the player loses all their health they are respawned from their last save level. Since I added attack colliders to the player attacks I decided to add spike bouncing. Similar to the enemy colliders, the level's hitable colliders are passed to the player physics function and if the player hits them a velocity in the correct direction is added depending on the attack. Players can then bounce using side and up slashes, causing a sideways arc and downwards velocity respectively.</p> <p>Challenges: The order in which colliders and attacks are calculated is important so I had to be careful about updating in the correct order and had to pass flags to functions signalling damage was dealt and so on.</p>	
<p>1/05/2021 4 hours</p>	<p>Achievements: Finished transition system between levels. Transition colliders were added to the entities.json and level.json for each level. In the level.json each transition has a direction and the transition and level it goes to. When the player collides with a transition they are removed from controlling the player and a constant velocity is applied in the appropriate direction so the character passes through the transition. The next level is then loaded and another transition applied, when going from a level the player loses control, is positioned correctly and a velocity is added to the player so they exit the transition. To smooth over the loading of the next level I added a screen fade by filling the render surface with a black of different alphas and a screen unfade by doing the reverse when loading into a level.</p> <p>Challenges: The upwards and downwards transitions were difficult to get the right velocity so the player exited and looked natural. For the downwards transition I used a constant velocity at 1/4 normal gravity. To exit upwards I added sufficient velocity upwards and to the right to create an arc so the play lands next to the transition.</p>	
<p>1/06/2021 7 hours</p>	<p>Achievements: Created all the rest of the levels. Mostly just placing tiles and colliders then playing to check that parts were possible. I reused and edited the demo levels I created and was left with 12 levels: Factory1, Factory1Challenge, Factory2, FactoryEntry, FactoryEntryChallenge, Subterranean1, Subterranean2, Subterranean3, Subterranean4, SubterraneanChallenge, Tutorial1, Tutorial2. The transitions between screens is mostly linear with increasing difficulty, with the challenge levels being short diversions from the main path which as the name implies, provide extra challenge. To create each level I drew a small sketch then blocked out the basic features, tested out the level, and tweaked until I had a good design before finally adding the visual details.</p>	<p>Some research on making platformer level designs: http://devmag.org.za/2011/07/04/how-to-design-levels-for-a-platformer/</p>



3/06/2021
5 hours

Achievements: Created title gui's, settings pages and pause menus. I did some research for the best approach to gui in pygame and found the library pygame_gui. Using pygame_gui I first created a class which initialized all the guis into a dictionary of menus. Making reference to the pygame_gui documentation I wrote a method which handles all the gui events. To indicate that you were hovering over a button I went for a similar approach to Super Meat Boys UI design by adding an arrow with pygame_gui's hovered_image theming parameter. For buttons which transition between pages I had to make a function which changes the currently displayed menu by looping through all the gui elements and hiding appropriate ones. When guis are open the mouse should also be displayed but hidden while playing. To give a different feel I downloaded a different mouse cursor which is based on Hollow Knight's cursor but is CC0 and loaded it with pygame by converting to an XBM file. This cursor is hidden and shown according to the gui state.

To add a main menu settings page I added some sliders for volume and a drop down for screen aspect ratio and fullscreen toggle. The sound options don't do anything but the screen settings do. When these settings are edited and saved, the new settings are written to file and the display has to be reset to be correct.

Pygame_gui documentation - <https://pygame-gui.readthedocs.io/en/latest/>

Pygame cursor reference - <https://www.pygame.org/docs/ref/cursors.html>

Custom cursor - CC0 - https://www.cursor.cc/?action=icon&file_id=98618

Challenges: Changing display options after pygame has been initialized is challenging, especially trying to change to fullscreen on Windows because the screen size isn't fetched properly and so the surface size fails to match the screen size. After a lot of research I found a method which accounts for Window's dpi scaling which worked. While I was at it I made the screen both resizable and movable. While resizing was fairly easy and just consisted of scaling one display onto another display, stretching the content, moving was difficult. When the display is moved the render loop is paused but the delta time isn't so time is skipped on Windows. After more searching I found a solution for Windows which gets the display rectangle and checks if it moved, and then resets the delta time to zero.

Challenges: Placing the gui elements with code correctly was tedious and difficult. Furthermore pygame gui does not support animations so I had to initialize some AnimatedSprites which are hidden and shown by the gui for the animated title background and logo. Furthermore the ingame gui health bar is animated so I had to create an AnimatedSprite which was cropped according to how much health the player had left. To indicate to the player they can save I added the press down to save sprite, and when they actually save I play the save animation in the corner of the screen.



Solution to resizing display in pygame after initialization - <https://gamedev.stackexchange.com/questions/105750/pygame-fullscreen-display-issue>

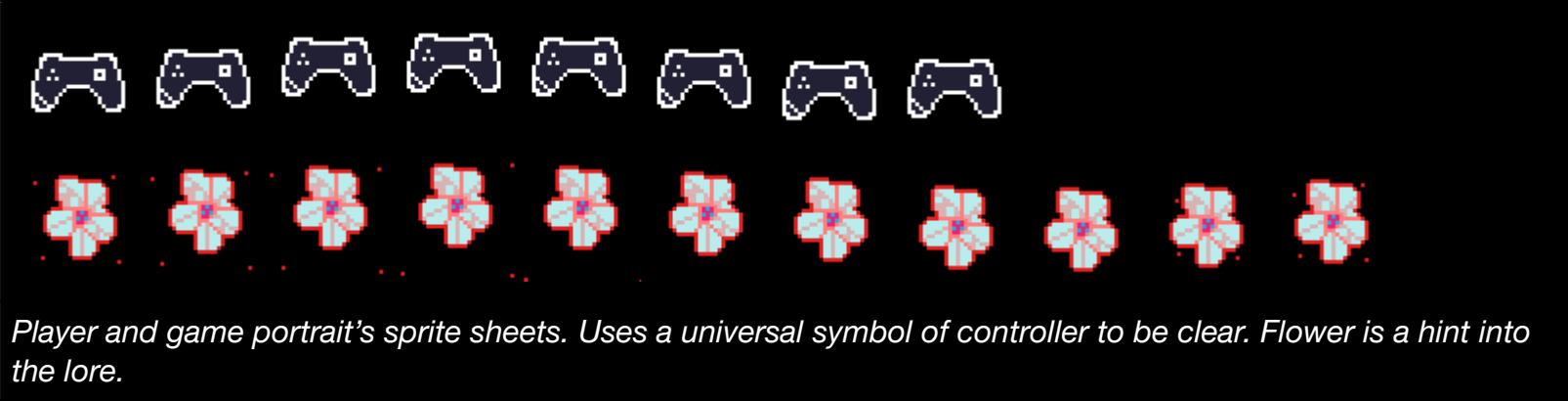
Solution to moving window, Alexandre's answer - <https://stackoverflow.com/questions/4135928/pygame-display-position>

6/06/2021
4 hours

Achievements: Implemented asynchronous sound engine and caching system for sounds. Added sound for each of the players actions, theme song during appropriate guis and ambient music while playing. I knew from early testing that pygame's audio capabilities were pretty poor, my main problem being that it popped and crackled when starting audio and it didn't handle multiple sounds well. To overcome this I did some research and after rejecting standard python libraries for either being too cumbersome or not properly asynchronous I found a small script which used pyaudio to buffer and play audio in a different thread. After making some tweaks I added to the game. Unfortunately this came with another problem that other libraries faced, long load times for audio. To overcome this I used pickle to save the loaded audio data to a binary format which takes python much less time to load. Just directly pickling the audio objects failed because they had associated threads to play the audio so I had to write a custom pickling and unpickling method which ignored the thread objects.

Audio script - https://github.com/steveway/papagayo-ng/blob/working_vol/SoundPlayer.py

Pickle documentation - used to write custom pickle and unpickle methods - <https://docs.python.org/3/library/pickle.html#pickle-inst>

<p>10/06/2021 3 hours 30 minutes</p>	<p>Achievements: Added dialog boxes and user name input. After looking over the assignments requirements I realised I need a way to convey both the lore and a tutorial to the player. To do this I decided to add dialog boxes after finding TextBoxify, a small pygame library. Similarly to the transition objects I added dialog to the level.json which is mapped to entities which the level loads. When a player enters a dialog collider the box is activated, displayed and a dialog sound is played. The player can't control their character while the dialog box is open and presses a key to continue and exit dialog. This was all fairly simple with this library. Since the assignment requires a name that is used in the game I added a gui menu with a text entry line which is opened when the player loads a new game save. The player name is saved to file and substituted into the dialog boxes so it is used in the game. To substitute the name I just made the dialog into a template string which the name is substituted into when it is displayed. To add some more visual interest to the dialog boxes I added animated portraits with the set_portrait function from TextBoxify. Furthermore there should be a divide between dialog that is instruction and lore so I made a game and player portrait which is switched based on whether the message begins with ellipses.</p>  <p><i>Player and game portrait's sprite sheets. Uses a universal symbol of controller to be clear. Flower is a hint into the lore.</i></p>	<p>TextBoxify - Source and documentation - https://github.com/hnrkcode/TextBoxify</p>
<p>17/06/2021 3 hours</p>	<p>Achievements: Fixed a bunch of bugs and added the final end of the game screen. To add the final screen I created an empty level and a new menu to the gui. I made it an empty level so there could be particles which moved up the screen, symbolising ascending. When the player transitions to the end screen a flag is set that disables user input and the save is overwritten with the default. While the flag is set the player and ingame gui is not rendered or updated and the menu screen is shown. The player then clicks a button on the gui to return to the title. Some of the bugs I fixed were with collision at high velocities and with updating the menu screen after saving and exiting to title.</p> <p>Challenges: I had trouble solving a bug with clearing the save after finishing the game, sometimes it would work and other times not. I realised (after a long time) that I was using a reference to the default save, which was being overwritten so when writing the default save I was actually writing the previous save. This was easy to</p>	

	solve by copying the save object rather than referencing but hard to see.	
22/06/2021 5 hours	Achievements: Final polishing of the game. I added in the splash animation when the player entered water and edited some of the levels to add more visual interest. To get splashes I had to create AnimatedSprite which were properties of the player and change their location when the player entered the water, and while they moved in the water a smaller splash was periodically added. The procedure was very similar for adding dust trails but I also had to determine how hard the player was landing by testing their vertical velocity at impact. For large landings two dust clouds were placed symmetrically by copying the object. I also added a different dust trail when the player changes direction instantaneously. While I was at it I added sounds for each of these actions. While players move in water the swimming loop plays and a splash plays on entry. A different landing sound plays for hard and soft landings. Furthermore while players fall, meaning sufficient downwards vertical velocity while not on the ground, a falling sound plays. To get some extra sounds I got some from this dropbox.	Sound dropbox - https://www.dropbox.com/sh/faqjj2ekftj1nb4/AADu5kD3mmbAJ-G-J84AmzBra?dl=0
27/06/2021 3 hours	Achievements: Added extra health powerup when player complete a challenge room. After playtesting with Joseph he said that the challenges were disappointing because players didn't receive anything at the end so I reuse the previous enemy animation and flying enemy code to create an oscillating floating crystal which when players collide with they get a permanent boost to their max health. The collection of a challenge crystal is added to the save file and when the player next loads or saves they get extra health, furthermore the crystal no longer spawns.	
28/06/2021 2 hours	Achievements: Finalised all the elements of the project including adding pictures to the logbook and submitted the final copy to classroom.	