[COSC360 - Computer Game Design](http://www.cs.otago.ac.nz/cosc360/assignments.php):

**Assignment 03 - Rapid Prototyping**

In our game design document a few issues were obvious at conception; however more issues showed up during implementation than expected.

We broke our prototype into 3 core sections. By doing this we were able to create a prototype that would address the issues within our assigned sections but without having to create all the code for the whole scenario individually to get a working concept ready for this week’s deadline.

The three sections were broken up as follows:

**JAMES GOODMAN – ENEMY/A.I:**

**Please see JamesVillageThrowers scene as to the testing for the issues presented below.**

**Issues:**

1. **2d Rotation:**
   * This was something we did not expect when creating a 2d game; the library is only setup for 3d rotation. (A side-scroller would not have encountered this)
   * The issue this caused was that our image would simply disappear when rotating as the image would turn on its side, there was no depth (z axis) to the image to keep the illusion that it was turning; You can’t use the 3d rotate library to turn a 2d image.
   * **Fix:**
     1. We had to find code that would allow the image to keep it’s perspective, this was not as easy as it sounds as understanding the vectors and what is actually going on is a sticking point for our team.
     2. Estimated fix time: 24hrs
2. **2d Rotation + follow:**
   * Although we fixed the rotation the issue when moving and turning opened up further weird issues, e.g the enemy AI would just rotate around the player in a circle, something I cannot really explain as to what was going on, partly due to my understating of the math behind the code.
   * **Fix:**
     1. This was partly fixed while completing the rotation. More code was investigated form the Unity community and a solution was found
     2. Estimated fix time: A combined 24hrs with the above solution.
3. **AI village thrower:**
   * The village thrower character is like a turret in the game, seems we had fixed the rotation obstacle we were now able to move onto shooting in the direction of the player. Some of the code could be used from the space invaders lab eg timing and shooting but the issue I had was that shooting in a direction of up and down was not going to satisfy the feeling of a “turret” as shooting needs to happen in the direction of the player.
   * **Fix:**
     1. Added a rigid body to the shotprefab which allowed me to create a force to the shot with 0 gravity.
     2. Estimated fix time 7hrs.
   * **Creation of objects(projectiles):**

As there were going to be many villagers the use of the probability of shooting still appealed to me, the feel of this code shows that although there are many villagers around not all of them need to throw something as some of them would just be watching the “monster” run through town. However, we need to make sure that objects when collide or when they miss the player it does not hang around in the system taking up memory.

* **Fix:**
  1. Created code which removes the objects from screen if they happen not to collide with anything.
  2. Estimated time to fix, 1hr.

1. **Other issues to address for AI but outside of prototype:**
   * AI path finding – this could be an issue.

As the AI would be in groups, losing the AI behind a wall or building as the Player is running away may not be noticed. (the player running past the AI may also address the AI being “stuck”)

* + Thrower bullet physics – do we need to add this to make it look more real?

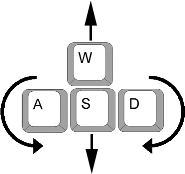
**Summary:**

For me there was not really any choice about choosing a way as such, (apart from different ways to code the same problem) everything was chosen by it working e.g. we needed our AI to face the player or it simply would not have looked right. This is the statement for all of our AI issues, it has to work, there may have been other ways to attack it but at the end of the day the finished product would have looked/acted the same to the user.

**DYLAN PRINGLE – PLAYER CONTROL/ARTWORK –CHARACTERS/LEVELS**

**Please see DylansTest scene as to the testing for the issues presented below.**

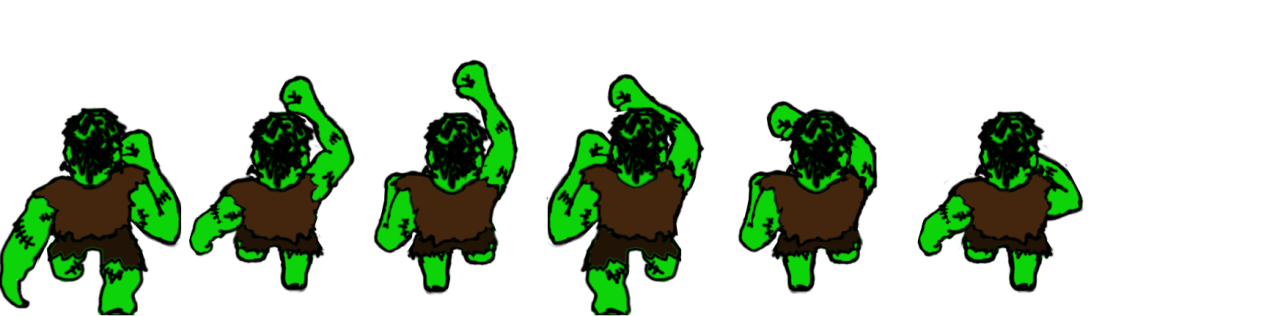
1. **Player movement:**
   * The original idea was to let the player move forward and sideways using the W, A and D keys, but refuse any backwards movement. This design was to suggest a sense of urgency that the player should only move forward, but however this strategy made the movement feel slightly limited. Our final idea was to have the player move forward using the W key but the sideways keys (instead of moving the player sideways) would rotate the player in the respective direction. This design felt more comfortable than the other idea, so we committed to this.

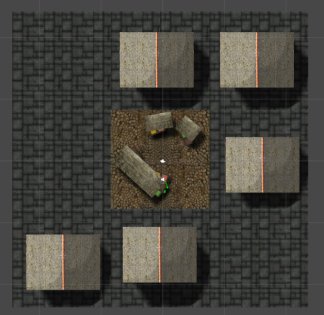


1. **Player collisions:**
   * Player collisions proved to be unnecessarily tricky. We set up colliders and had the player sort of bounce back from a wall if they made contact, so that they would keep executing the collider script with each new collider contact. If the player persistently ran into a wall however they would eventually bypass the collider and run straight through the wall. This approach was proving buggy so we needed a different solution.
   * **Fix:**
     1. The problem was that I was overthinking the problem. When the first snippets for this collision code was being written, we were still rolling with the idea that the movement keys made the player run in those directions and had not implemented the rotating player idea yet. So I was trying to continue movement on other axis but stop the movement on the axis that came into contact with a wall (hopefully that makes sense). The solution was solved by James C when we introduced the new rotating player movement design, it was fairly simple to just end the players movement, when they hit a wall.
2. **Player attack:**
   * I based the attack script from the space invaders shooting script, but destroy the shot a few millaseconds after its instantiated. This idea was to quickly create a collider infront of the player to collide with enemies, and then quickly disapear. This approach worked fine, but I found problems trying to instantiate an (attack)object as a child to its parent(the Player) so that the shot would fire infront of the player in whatever direction the player parent was facing.
   * **Fix:**
     1. Attached the attack collider as a child to the parent initially without being instantiated and instead of creating and destroying objects whenever the player attacks, simply activate/deactivate the attack object. This worked very effectively and was also more friendly to the engines processing resources.

*(Above –Player with attack collider infront with rock sprite attached for visual convenience, final attack will not have sprite attached. Space bar is used for the attack button).*

1. **Artwork/Animation**
   * We already had a pretty clear theme for the game so I started experimenting with sprite designs for our main player monster in photoshop.
   * I ended up really liking a colourful cartoonish look for the player. I then made an animation out of the monster sprite. I only had to draw a second sprite to make a running animation for the player. All I did was flip the sprites horizontally in a order to give the impression hes running when used in a loop. 
   * I also tried to implement a punching animation based off the running animation to use when the player attacks an enemy. *(This animation is currently not executing the way it needs too. There are timing issues between when the hit button is pushed and when the animation runs. This is being addressed but may not be fixed by the time this prototype report is handed in.)*
   * First enemy animation.*(This enemy is running and attacking at the same time.)*
2. **Level Design**
   * Using photoshop, I eventually came to like the look of realistic textures to build elements of the levels with. This realistic look, contrasting with the cartoonish characters, I feel gives a nice juxtaposition of visuals. I also think that if the level textures were drawn in a similar style to the characters, the level would almost have too much colour. The textures also in contrast to the characters look almost uninteresting which keeps the characters more eye catching, and easier to focus on.

  **

*(Two map block section ideas, and a combination of them joined together to make a draft map block.)*

* + An additional feature I added was the ability for characters to run under certain texture sprites giving the impression the sprites are higher above them. This feature looked great when running under the market stalls fabric roofs and under the roof eaves of houses.

1. **Other issues to address for Game but outside of prototype:**
   * Different looking enemies, map obstacles (static as well as destructible).
   * More diverse set of map block sections.
   * A stamina component attached to player, that decreases if player attacks or other attacks are made but slowly regenerates.
   * The ability for player to initiate a charge attack that will increase players speed and inflict automatic attacks on colliding enemies for a short amount of time. This charge will quickly decrease the players stamina so this charge attack should only be used sparingly.
   * Comic strip storytelling at the beginning and end of level.

**Summary:**

Overall I’m very pleased with the current state of the prototype. I think as a team, we are all setting realistic, individual goals. Each task brings a new learning curve with initial difficulties, but we have all so far managed to overcome them. We each decided on a clear direction the game was going to go and have stuck to that direction, and this is why we only needed to implement one prototype.

**James Chalmers – Procedural Generation, Pathfinding, Mouse Controls**

**Procedural generation**

Scene: TileLoaderTest – Walk Upward to see generation, controls are W to accelerate, A an D to turn and spacebar to punch.

Wanted to achieve seamless transitions between map blocks(aka tiles) that load and deload depending on how far the player has made it through the level.

This took a few hours although idea for solving it came to me very quickly as it seemed obvious to use an array to store prefab tiles and random numbers to select the index to load.

It was clear that I need to be able to get the size of the most senior object in the prefab tiles hierarchy in order to be able to load the tiles in the correct places. I figured out that using the renderer of this object was the best way to do this as it allows size to be obtained. Using this and the camera height I was able to calculate the right place to load the tile along the y-axis. This also allows for the correct place to initiate loading and deloading to be calculated.

Some issues and questions this introduces that are:

* If the player returns to a previous tile after it has deloaded it will not be there.
* Enemys deload with tiles so it is possible the player will witness them disappearing if the enemy makes it onto the next tile.

These issues are best fixed when we have a feel for the rest of the game.

**Pathfinding**

Scene: JamesCPathfindingTest - controls same as previous.

Wanted to improve AI so they do not walk into walls.

This prototype was very quick as pathfinding code and infrastructure was provided, I just changed AI to rotate to face the player.

As a team we liked the simple AI because when using a moderate grid size, the enemies would shift around too much, which is not what we wanted. With a finer grid size things improved however the game will slow down if we load large numbers of AI at once as we plan to. I think we could potentially use this for specific enemies such as dogs or a stronger enemy if we have time to implement them but I think for the mob of villagers we are going to go with the simple AI. If some get stuck behind walls it doesn’t really matter because there will be many of them.

Some issues and questions this introduces that are:

* If used with the procedural generation we need to find a way to shift the grid to the correct location.
* This could cause the game to run slow if not implemented properly.

**MouseControls**

Scene: MouseControlsTest - Controls are W to move, mouse point to change direction, mouse wheel to zoom in and out and left click to punch.

Wanted to test an alternative to keyboard controls and also test zoom controls

This took a few hours mostly to get the zoom to work properly.

I think the mouse control could be improved over the current state but they are not a priority, as I do not think they have the right feel for the game. In particular the click to attack feels unsatisfying to me. The zoom function however could be useful as it gives the player more options on how they prefer to play.

Some issues and questions this introduces that are:

* It might not be a good idea to let the player choose the zoom as it could effect the difficulty if they can see what is coming. It may become more of a necessity to zoom out as apposed to an option.
* It might be better to add a zoom function that is related to the speed of the character to incentivize the player to run faster as they will be able to see further ahead if they do.

**Other tweaks**

- Added cooldown and slight knock back to melee AI attack to stop the player getting stuck (and getting pushed around by AI) and allow AI to continue attacking.

- Managed to simplify wall collisions so they are not longer an issue.

- In scene LevelBoundTesting, I have started work on adding boundaries to the levels but with the procedural generation it has proven to be a challenge. The bottom part of the screen needs to be temporarily deactivated to allow the player to pass through but then activate when previous tile deloads.