

Game Design Document

Nikkolas Diehl – 16945724

Auckland University of Technology – AUT

This game design document will describe and analyse the game concept for the game development paper (COMP710) at AUT by Nikkolas Diehl:

Game Name:

Really Warm

Game Rules:

- The aim of each level is to defeat all the enemies within the scene and then reach the end point of the level
- Each level will get progressively harder by introducing more enemies per level
- The player and AI will have the option of using either their fists, a sword, or a gun. Guns will have limited ammo and there will be different variations on the guns.
- The player can also throw a weapon such as an empty gun at an enemy to stagger them.
- The player can stop moving to slow the game down and move to speed the game up.
- You can run, jump, walk around, shoot and attack.

Game Mechanics:

- The player will be able to use mouse & keyboard or a controller to control the character. The game will be played in a 2D scene across multiple levels
- Each level will be created with multiple enemy spawn points and weapon spawn points. Your job as the *player* is to find and pick up a weapon to use against the enemies. The enemies are also able to pick up the same weapons you use.
- The main function of the game and what will make it challenging is that the game will run in *slow mode* and all bullets, weapons, and AI will move, fall and react in slow motion *as long as the player is not moving*.
- Any player movement will result in the games movements speeding up. When the player moves the cross hair (either via the mouse or controller thumb sticks) the game will speed up to about half speed. When the player is physically moving his/her character, the game will be at full speed and the player will have to dodge bullets and sword attacks.
- Whilst in mid-air, the player will be physically moving and thus the game will run at full speed.

Key Algorithms:

A lot of the mechanics in the game will be relatively simple such as level creation and so on. However, some of the core functionality will be driven by specific algorithms and functions:

- The slow down mechanic will be an algorithm that slows the game down when you're not moving, and only allows physics to process if the player is moving.
- The animation algorithm will be used to control how the AI and the player is animated. In both situations, the sprite used will be a large 2D sprite sheet array of movement. For the animated of rotation, 360 slightly altered sprites will be created to show the players body shifting through all 360 degrees of a circle.
- The collision detection algorithm will be diverse and action driven. Each bullet, player, and AI will collide with walls, floors, roofs and each other. And to note: every particle will still collide with the walls, floors and roofs but will not collide with each other.
- A state machine will be used to control the AI and players. There will be a limited amount of states such as STAGGERED, NO_WEAPON, HOLDING_SWORD, HOLDING_GUN, DEAD and ALIVE. The player and each AI will have two current states. One to check if they're alive or dead and one to check their current movement state as one of the other 4 states.
- The AI algorithm will be relatively simple. The AI will be able to attack in three different ways, will try follow you and run up to you, and will dynamically react to the player.

Specific Features and Rule based mechanics:

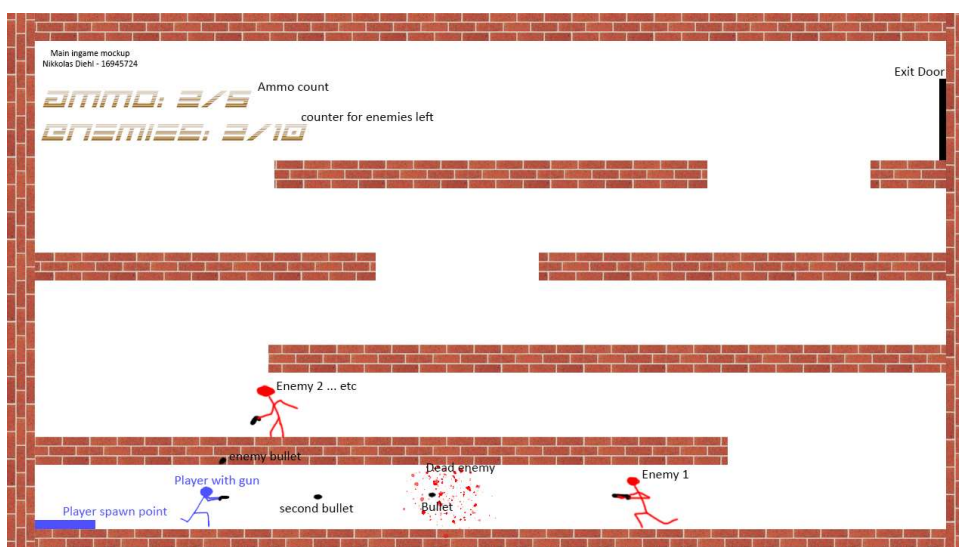
- Each level will be an upwards action driven level. The player must essentially reach the top whilst killing all the enemies to win. This is because of two reasons:
 - Being higher up gives a natural advantage and I want to display that as a feature
 - It allows me to make everything physics based. If an enemy dies, particles, bullets and etc will fall downwards and away from the player.
- When moving at all, the game will start to speed up and run at full speed. When not moving the game will either not run or run very slowly at sub 0.1 deltaTime multiplier.
- All bullets will both one shot the enemies as well as the player. This is to add challenge.

Mock-up Interface details:

- The main menu will contain a play button to start the game, an instructions button to read the control scheme as well as the rules, a scores button (the game will track the highest time survived as seconds) to view the top score and an exit button to quit the game.
 - The mock up is shown attached:



- The in-game scenes will be generated from level data files as a data driven programming style. Each level file will contain things such as level name, level count, all image links used in a level, and wall/floor/roof positions from X_1, Y_1 to X_2, Y_2 .
 - The mock up for the in-game scene is shown attached:




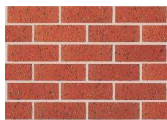


Cheat Features:

- For debugging purposes, a small set of cheat abilities will be included with the builds. Only available in the documentation:
 - A button to instantly kill all enemies in a level.
 - A button to immediately spawn a new enemy within a level.
 - A button to turn on infinite ammo.
 - A button to become immortal.
 - A button to teleport to the end of the level.
 - A button to skip to the next level.
 - A button to give yourself a weapon.

Required Asset list:

Sprites:

- Player sprite sheet – Self drawn in Adobe illustrator as a gif and then exported as multiple images and combined into one sprite sheet. I am going for a stick figure look to the game.
 - Example: 
- AI enemy sprite sheet – Self drawn in Adobe illustrator as a gif and then exported as multiple images and combined into one sprite sheet.
 - Example: 
- Bullet sprite. Self-drawn in photoshop or paint. A simple black bullet shaped dot that will be rotated at the angle of fire via programming.
 - Example: 
- Walls/Roofs and floors will use a brick texture found online.
 - Brick to be used:  (Austral Bricks, 2019)
- Exit door sprite. Self-drawn in photoshop or paint. A simple black flat doorway.

Sounds:

- Music. Any and all music used in the game will be affected by the deltaTime multiplier and so will sound slower when slow-mo is running and sound normal when the game is at 1 times speed. Any and all sound will come from free channels and non-copy write audio libraries such as audio library on youtube (Audio Library, n.d.).
- Bullet sounds will be gathered from free sound libraries as well, such as the soundbible.com and for the guns, I will probably only be implementing a 9mm pistol so the gun shot sound will be gathered from soundbible. (Koenig, 9mm Gunshot Sound, 2016)

- Sword sounds will be gathered from the same place as the gun shot (soundbible.com) and will use the sound listed here: (Swoosh 3 Sound, 2009)
- AI and player immediate death noise. This sound will play immediately on death of both the AI or the player. The sound will also be gathered from soundbible.com and will use the sound listed here: (Vladimir, 2011)
- When a player dies, as the death screen comes up, a beeping noise will be heard as a sign of *termination* and to give power to a death of the player. The sound will also be gathered from soundbible.com and will use the sound listed here: (Koenig, ECG Sound, 2011)

Bibliography

Audio Library. (n.d.). *Audio Library - Music for content creators*. Retrieved from Youtube.com:
<https://www.youtube.com/channel/UCht8qITGkBvXKsR1ByIn-wA>

Austral Bricks. (2019). *Homestead*. Retrieved from australbricks.com.au:
<https://australbricks.com.au/sa/product/homestead/>

Koenig, M. (2011, February 09). *ECG Sound*. Retrieved from soundbible.com:
<http://soundbible.com/1730-ECG.html>

Koenig, M. (2016, March 20). *9mm Gunshot Sound*. Retrieved from soundbible.com:
<http://soundbible.com/2120-9mm-Gunshot.html>

MvG. (2017, December 20). *Direct way of computing clockwise angle between 2 vectors*. Retrieved from stackoverflow: <https://stackoverflow.com/questions/14066933/direct-way-of-computing-clockwise-angle-between-2-vectors>

Mykhailo, D. (2011). *Angle between two vectors*. Retrieved from OnlineMSchool:
<https://onlinemschool.com/math/library/vector/angl/>

Swoosh 3 Sound. (2009, July 06). Retrieved from soundbible.com: <http://soundbible.com/706-Swoosh-3.html>

Vladimir. (2011, November 29). *Neck Snap Sound*. Retrieved from soundbible.com:
<http://soundbible.com/1953-Neck-Snap.html>