Days Are Numbered

TECHNICAL REPORT

Adam Kearney
X13748659@student.ncirl.ie

Declaration Cover Sheet for Project Submission

SECTION 1 *Student to complete*

Name:
Student ID:
Supervisor:

SECTION 2 Confirmation of Authorship

The acceptance of your work is subject to your signature on the following declaration:

I confirm that I have read the College statement on plagiarism (summarised overleaf and printed in full in the Student Handbook) and that the work I have submitted for assessment is entirely my own work.

Signature: ____________________________________ Date: ____________

NB. If it is suspected that your assignment contains the work of others falsely represented as your own, it will be referred to the College's Disciplinary Committee. Should the Committee be satisfied that plagiarism has occurred this is likely to lead to your failing the module and possibly to your being suspended or expelled from college.

Complete the sections above and attach it to the front of one of the copies of your assignment,

What constitutes plagiarism or cheating?

The following is extracted from the college's formal statement on plagiarism as quoted in the Student Handbooks. References to "assignments" should be taken to include any piece of work submitted for assessment.

Paraphrasing refers to taking the ideas, words or work of another, putting it into your own words and crediting the source. This is acceptable academic practice provided you ensure that credit is given to the author. Plagiarism refers to copying the ideas and work of another and misrepresenting it as your own. This is completely unacceptable and is prohibited in all academic institutions. It is a serious offence and may result in a fail grade and/or disciplinary action. All sources that you use in your writing must be acknowledged and included in the reference or bibliography section. If a particular piece of writing proves difficult to paraphrase, or you want to include it in its original form, it must be enclosed in quotation marks and credit given to the author.

When referring to the work of another author within the text of your project you must give the author's surname and the date the work was published. Full details for each source must then be given in the bibliography at the end of the project

Penalties for Plagiarism

If it is suspected that your assignment contains the work of others falsely represented as your own, it will be referred to the college's Disciplinary Committee. Where the Disciplinary Committee makes a finding that there has been plagiarism, the Disciplinary Committee may recommend

- that a student's marks shall be reduced
- that the student be deemed not to have passed the assignment
- that other forms of assessment undertaken in that academic year by the same student be declared void
- that other examinations sat by the same student at the same sitting be declared void

Further penalties are also possible including

- suspending a student college for a specified time,
- expelling a student from college,
- prohibiting a student from sitting any examination or assessment.,
- the imposition of a fine and
- the requirement that a student to attend additional or other lectures or courses or undertake additional academic work.

Table of Contents

Executive Summary 5

1 Introduction 6

1.1 Background 6

1.2 Aims 6

1.3 Technologies 7

2 System 8

2.1 Requirements 8

2.1.1 Functional requirements 8

2.1.2 Data requirements 17

2.1.3 User requirements 17

2.1.4 Environmental requirements 17

2.1.5 Usability requirements 18

2.2 Design and Architecture 18

2.3 Implementation 20

2.3.1 Player Object 20

2.3.2 Level Generator Script 24

2.3.3 Day/Night Cycle Object 27

2.3.4 Slimer Enemy Object 28

2.3.5 Shooter Enemy Object 30

2.4 Graphical User Interface (GUI) Layout 32

2.5 Testing 34

2.5.1 User Test – Robert Dillon 35

2.5.2 User Test – Eamon Coleman 36

2.6 Further Developments 37

2.7 References 38

3 Appendix 39

3.1 Project Proposal 39

3.1.1 Objective 39

3.1.2 Background 40

3.1.3 Technical Approach 41

3.1.4 Technical Details 42

3.1.5 Evaluation 43

3.2 Monthly Journals 45

Executive Summary

Days are Numbered is a 2D action survival game built in the GameMaker engine. The game is designed to give the user a new experience each time they play due to the randomly generated levels. Because of the unpredictable nature of the game it forces the player to master the mechanics of the game rather than memorizing levels and patterns like you would with many other games.

Some of the main features in Days are Numbered are as follows:

- Randomly generated levels that will increase in difficulty as the game days' progress, this will increase the replay value of the game as the levels will be unpredictable and change each time you play.
- Day and night cycle that will change elements of the game, it will be more difficult for the player to navigate the level at night as their view will decrease when it gets darker, this will encourage the player to be more cautious when it becomes darker.
- Large selection of weapon and item types, weapons will spawn randomly on the map as the game progresses the chances of better weapons spawning will increase.
- Multiple enemy types that will get increasingly difficult as the game progresses, the speed and damage values of enemies will also increase as the game progresses. Harder enemy types will appear more frequently as the game progresses.

The games setting takes place after a mysterious virus infects most the population and turns them into mindless monsters, you take the role of a survivor whose goal is to survive as many days as possible, you must defend yourself and scavenge for food to ensure you don't die of starvation, every journey will differ and none will be easy.

1 Introduction

1.1 Background

The game industry is an industry that is constantly growing, in 2014 there was an estimated 206 million euros in revenue from Ireland alone. Within this growing industry is a growing trend, this trend is indie video games. In the past, almost all video games would require a large team and a big budget to produce a game that is fit for market, but nowadays games are being produced by smaller teams or even a single person, these games are known as Indie games and many have been huge successes among many platforms.

Survival games have become one of the most popular genres of video games over the last few years and most these games are indie games. A survival game is a subgenre of action video games that generally start the player with minimal resources in a hostile environment, in these games the player must find resources to stay alive long enough to progress through the game.

Day are Numbered takes is inspiration from several successful indie games such as: Spelunky, FTL and Nuclear throne to name a few.

1.2 Aims

The aim of this project is to create a game that is fast paced and exiting but will offer the player a new experience each time they play. The look of the game will be a retro pixel type of art style which is inspired by classic SNES games. I wish to create an experience that will encourage players to change their playstyles depending on the situation that is presented to them, for example certain situations may encourage to user to avoid conflicts and remain stealthy.

1.3 Technologies

Game Maker Studio: Game Maker Studio is a video game creation tool that allows for the creation of cross-platform and multi-genre games, it features a simple drag and drop tool but also allows the user to write their own scripts using a scripting language called Game Maker Language (GML) which can be used to develop more advanced games. GML is the primary interpreted scripting language of Game Maker, it is quite similar to C++.

Game Maker accommodates redistribution on multiple platforms such as, Windows, Mac, Linux, Xbox, and PlayStation. Using Game Maker, you can implement complex game features similar to ones you would see in big budget games hassle free.

Many Game Maker games have gone on to be massive successes, such as Undertale which was created completely using Game Maker by one person, the game raised over $50,000 on Kickstarter and went on to sell 2 million copies worldwide. Another Game Maker success story is Hyper Light Drifter which raised over $600,000 on Kickstarter and has received numerous awards.

GameMaker
Studio

Aseprite: Aseprite is animated sprite editor and pixel art tool which makes it very easy to create and animate custom game sprites, I will create all the games original art and animation using Aseprite.

aseprite

2 System

2.1 Requirements

2.1.1 Functional requirements

Requirement 1: Play Game

<u>Description & Priority</u>

The user must be able to play the game (by controlling the playable character on the screen) which is the highest priority of the game

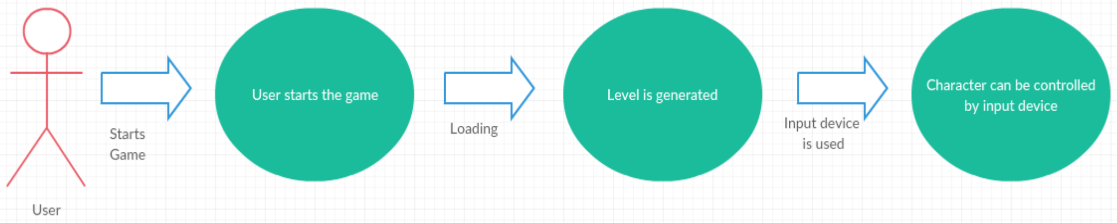
<u>Use Case</u>

Scope

The scope of this case is to allow the user to play the game

Description

The use case describes the functions of movement that the player must be able to do in order to play the game



Flow Description

Precondition

The game is installed and the start menu is currently being displayed. The user selects start game from the start menu.

Activation

This use case is started when the user uses their input device to control the main character

Main Flow

1. The game randomly generates a level
2. The user sends an input using their input device
3. The game reads the input and performs the correct action

Termination

The Process stops when there is no input being sent from the input device

Requirement 2: Save Game

Description & Priority

The user must be able to save the game from the start menu so that they can continue their progress later.

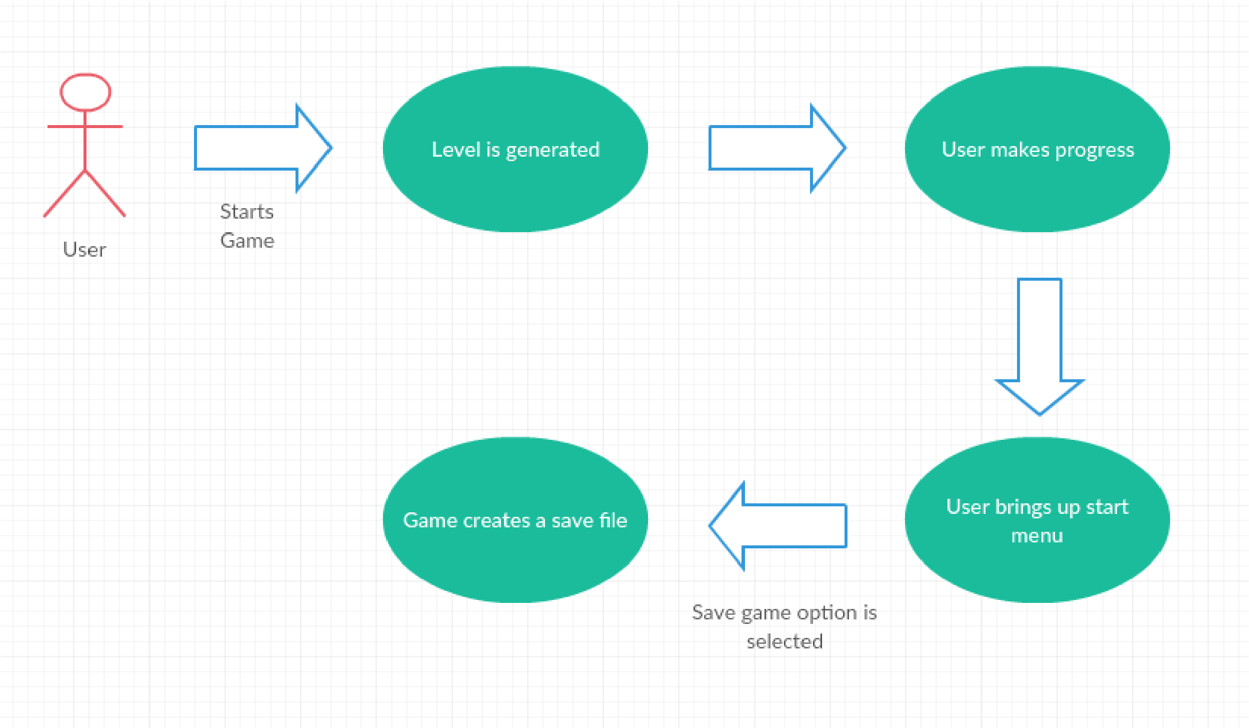
Use Case

Scope

The scope of this use case is to allow save their progress

Description

This case describes the function of saving an instance of the game, the save will store the users location and their stats



Flow Description

Precondition

The game is installed and running, the user must start the game from the main menu and have made some progress

Activation

This use case starts when the users brings up the in-game start menu and selects the save game option

Main Flow

1. The game has generated a level
2. The user can control the playable character
3. The user brings up the in-game start menu
4. The game displays the in-game start menu
5. The user selects the save game option
6. The game creates a save file
7. The game stores a save file on the user's machine

Termination

The game creates a save file and the user is return to the in-game start menu

Requirement 3: Load Game

<u>Description & Priority</u>

The user must be able to load a game state from a save file

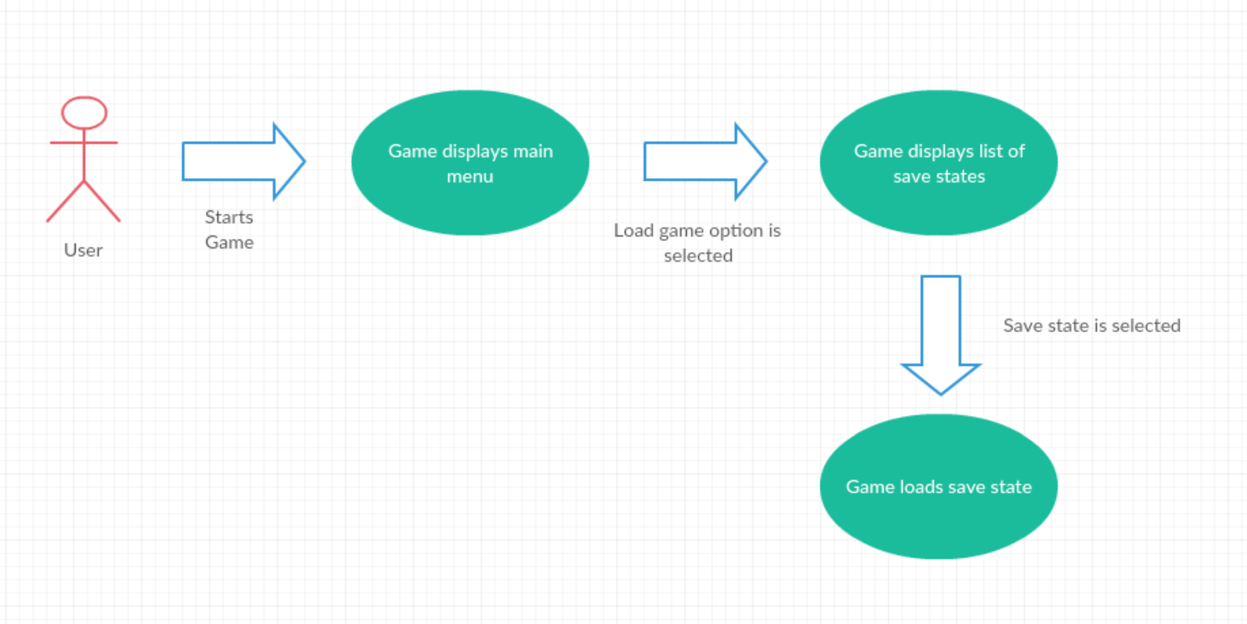
<u>Use Case</u>

Scope

The scope of this use case is to allow the user to load a game state from a save file and continue from where that save state was created

Description

This use case describes the function of loading an instance of the game from a save file and allowing the user to continue from when that save file was created.



Flow Description

Precondition

The game is installed and running, there must also be at least one save file

Activation

This use case is started when the user starts the game and selects load game from the games start menu

Main Flow

1. The user starts the game
2. The game displays the main menu
3. The user selects the load game option
4. The game starts the game from where the save file indicates

Termination

The game loads the save file and starts the user from that location in the game

Requirement 4: Exit Game

Description & Priority

The user must be able to exit the game from the start menu or if they are currently playing the game they should be able to exit from the in-game menu

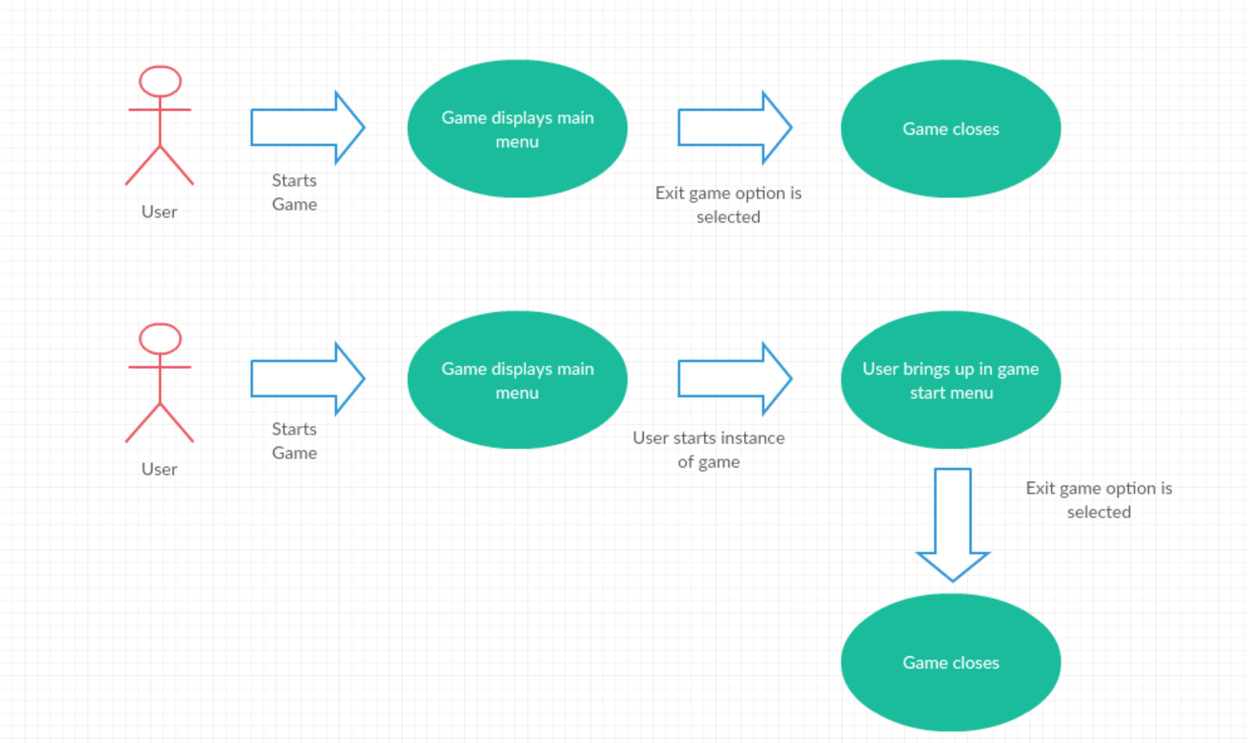
Use Case

Scope

The scope of this use case is to allow the user to exit the game from the main menu or the in-game menu and return them to their desktop

Description

This use case describes the function of exiting the game from either the in-game menu or the main menu when the game is started. Exiting the game should return the user to their desktop


User
Starts Game
Game displays main menu
Exit game option is selected
Game closes
User
Starts Game
Game displays main menu
User starts instance of game
User brings up in game start menu
Exit game option is selected
Game closes

Flow Description

Precondition

The game is installed and running, the user is currently on the main menu or on the in-game start menu

Main Flow

1. The user starts the game
2. The game displays the main menu
3. The user selects the exit game option
4. The game shuts down and returns the user to the desktop

Alternate Flow

1. The user starts the game
2. The user starts and instance of the game
3. The user brings up the in-game start menu
4. The user selects exit game from the start menu
5. The game shuts down and returns the user to the desktop

Termination

The game exits correctly and returns the user to the desktop

Requirement 5: Interact with the game items & systems

Description & Priority

The user must be able to start an instance of the game and during that instance the user must be able to interact with the in-game objects and systems such as: pick up items, open doors and read signs. These functions are a major priority of the game

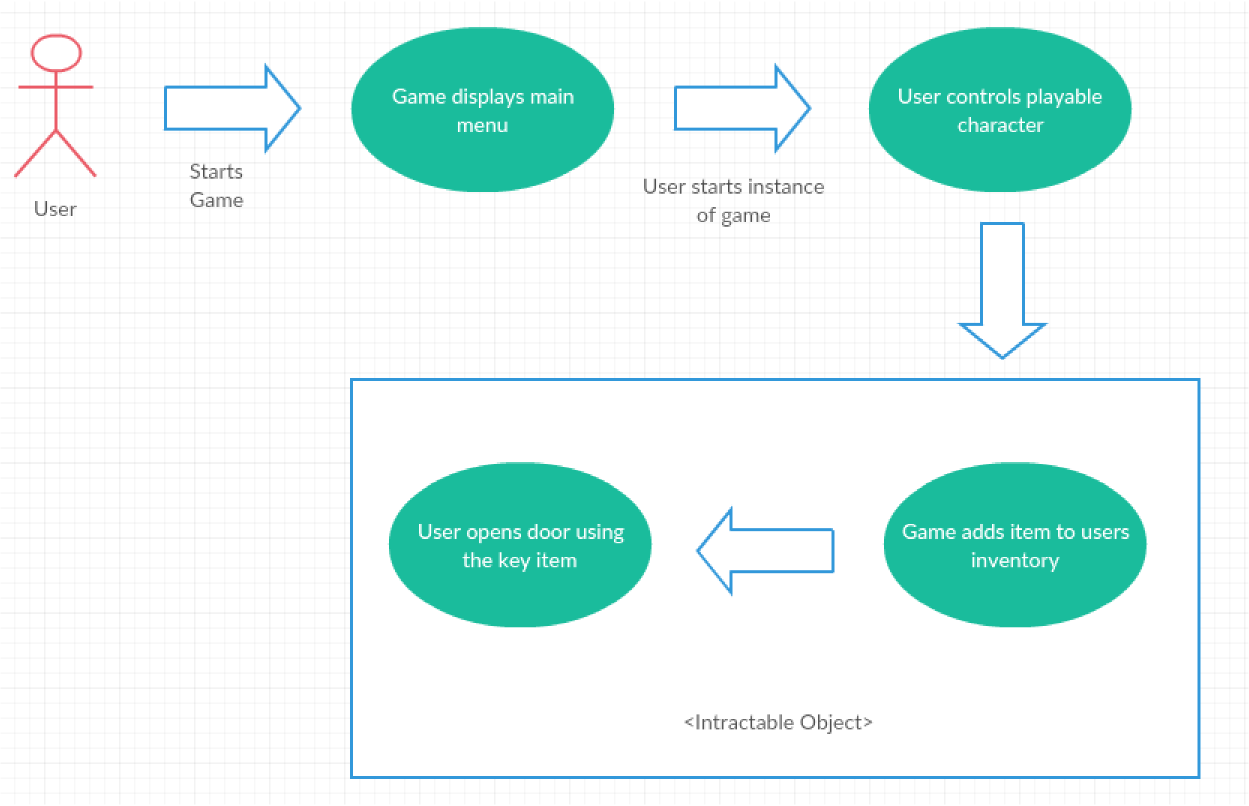
Use Case

Scope

The scope of this use case is to allow the user to interact with the various in-game items and systems

Description

This use case describes the function of the user interacting with the in-game items and systems. An example of this is finding a key and unlocking a door


User
Starts Game
Game displays main menu
User starts instance of game
User controls playable character
User opens door using the key item
Game adds item to users inventory
<Intractable Object>

Precondition

The game is installed and running. A level has been generated and the user is now in the game

Activation

This use case is started when the users starts the game, loads and level and interacts with an in-game object or system

Main Flow

1. The game loads a level
2. The user is controlling the playable character
3. The user find an object that can be interacted with
4. The game allows the user to interact with the object

Termination

The game allows the user to interact with the object

2.1.2 Data requirements

The user requires the ability to save their progress so they can return to where they left off last time they played. Game Maker has a built-in game save function, this function creates and INI file with all the variable values of the current game state, there is also a load function that will read the INI file to help recreate where we left off.

2.1.3 User requirements

The user will need the following in order to run the application:

- A desktop or laptop with the following specifications
  - Windows 7 or Later
  - A DirectX 9 (or later) compatible graphics card
  - 512MB of RAM
- A keyboard and mouse or a controller that is compatible with GameMaker
- A build version of the game

2.1.4 Environmental requirements

The game is being built to run on Windows. Game Maker's Windows exporter will ensure the game runs smoothly on any Windows system from Windows 7 to 10. Performance on other operating systems will be easy to achieve but for this project I am only going to focus on Windows operating systems.

2.1.5 Usability requirements

The game's interface should be easy to understand and give the player all the information they need from a quick glance, it is important to keep the interface simple so that the player can focus on the action of the game and not be distracted by a clunky interface with unnecessary information. The following usability requirements are also featured in the game

Understandability: The interface is easy to understand and the goal of the game is clear

Learnability: There are tutorials that clearly explain all elements of the game as well as documentation that is up to date.

Operability: The system settings should be customisable to suit specific player needs.

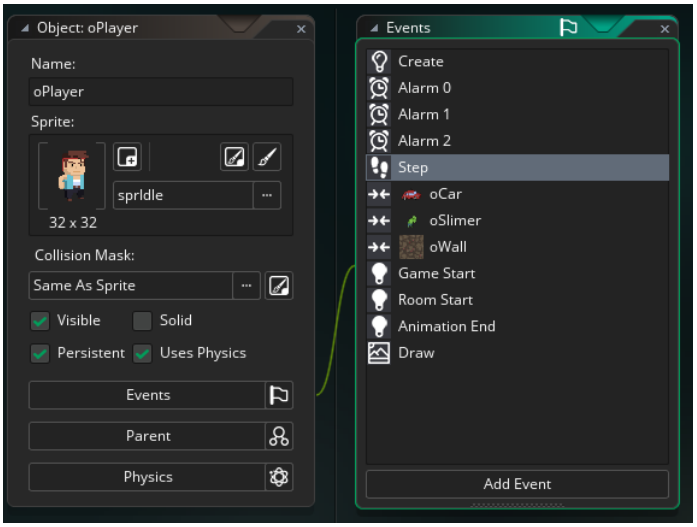
Attractiveness: The interface should remain simple but still look appealing and match the theme of the game.

2.2 Design and Architecture

The architecture of Game Maker can be split into three categories:

Data: This category is all the game assets such as the game sprites, sounds, backgrounds and fonts, essentially everything associated with the visuals and sound of the game.

Control: The control category is all the code of the game such as your game objects as every object will have numerous events and the game scripts which will be used to control many functions of the games and allows us to use function calls so we don't have to code the same functions individually for each object.


Object: oPlayer

Name: oPlayer

Sprite: sprIdle

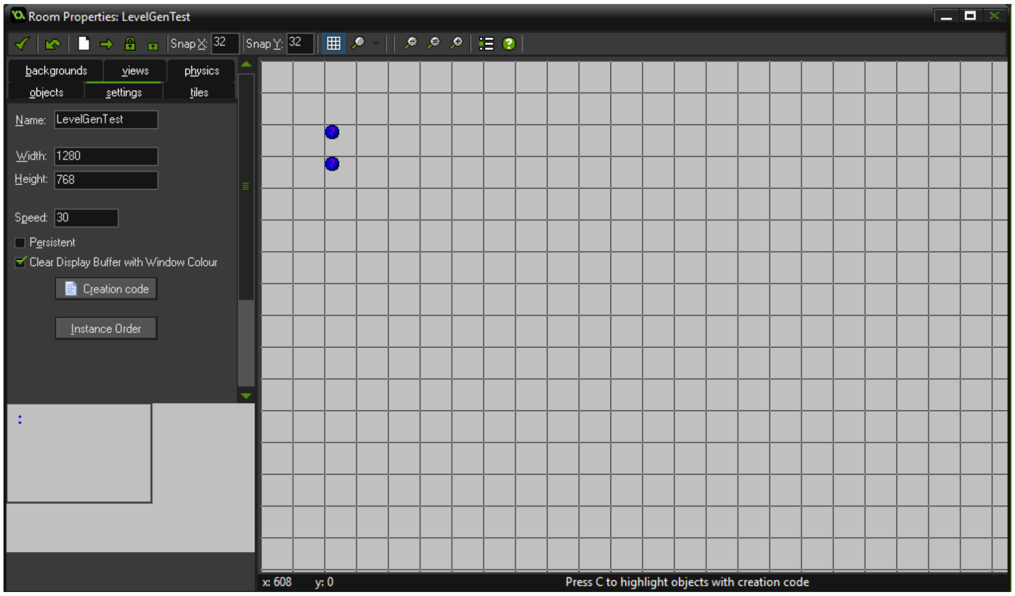
32 x 32

Collision Mask: Same As Sprite

☑ Visible ☐ Solid

☑ Persistent ☑ Uses Physics

Levels: Within the levels category we have our rooms which will be each individual level, each level has a view, this controls the camera of the game. For this game, there will be only one main room as the random level generator script will handle all the level building in the game, this one room will hold the objects which will run all the generation scripts. The pause and main menus are also rooms but only display text for the user to start or leave the game.


Room Properties: LevelGenTest

Snap X: 32 Snap Y: 32

Name: LevelGenTest

Width: 1280

Height: 768

Speed: 30

☐ Persistent

☑ Clear Display Buffer with Window Colour

x: 608 y: 0

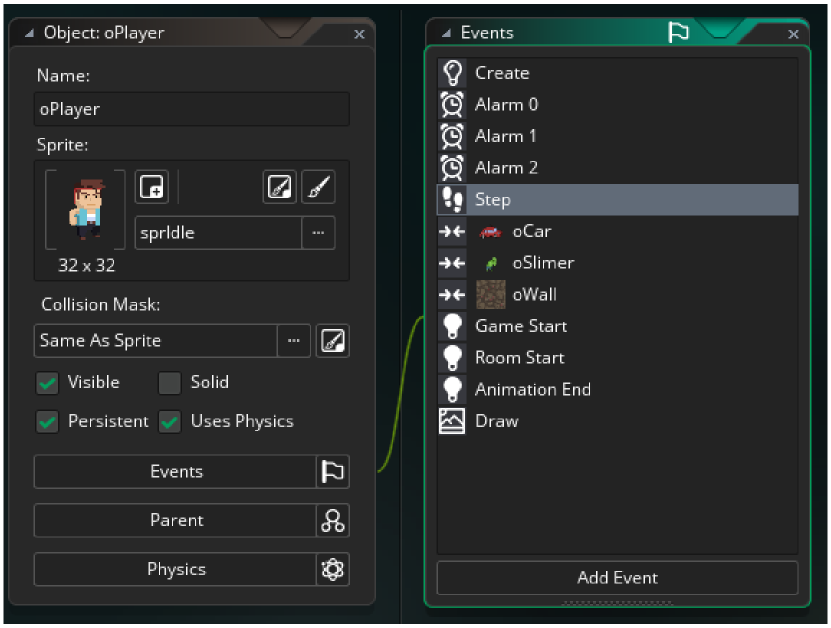
Press C to highlight objects with creation code

2.3 Implementation

In this section, I will explain how I implemented some of the more important aspects of the game.

2.3.1 Player Object

The player object is the most important object in the game and has the most events of all the game objects, below you can see the player object and the large number of events associated with that object:



Object: oPlayer

Name: oPlayer

Sprite: sprIdle

32 x 32

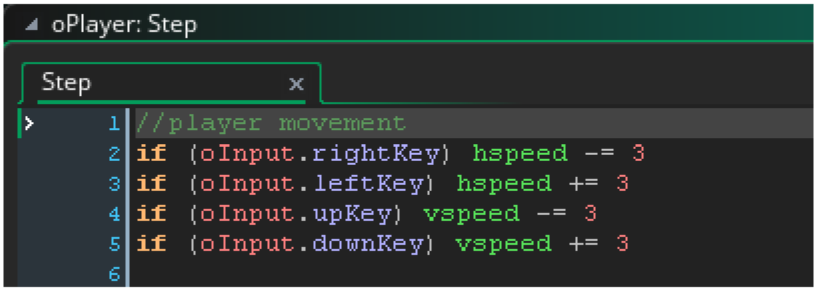
Collision Mask: Same As Sprite

☑ Visible ☐ Solid

☑ Persistent ☑ Uses Physics

Player Step Event

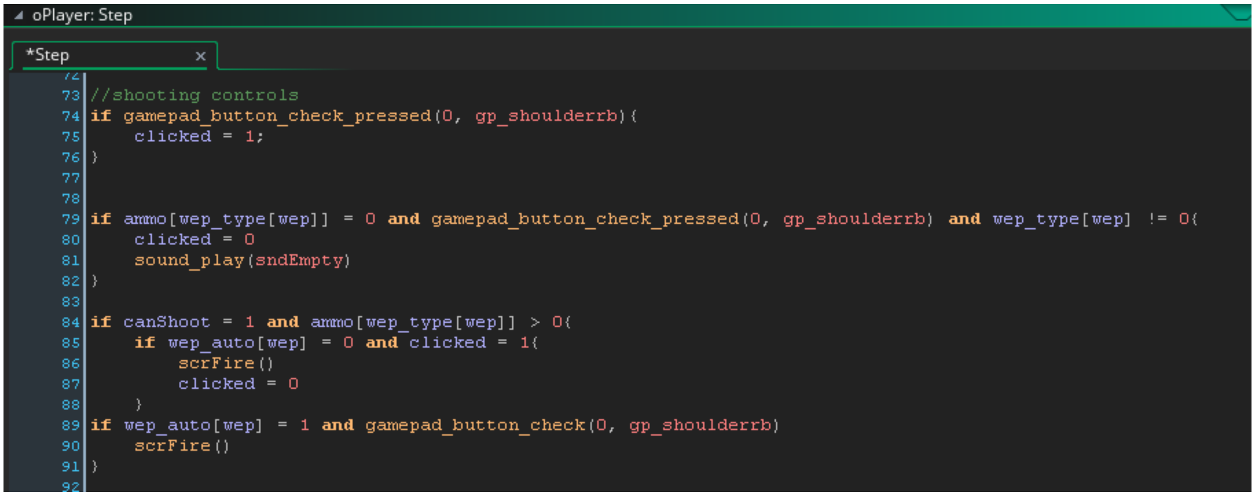
A step event is basically a loop that runs constantly with all the events being checked and triggered as necessary while the game runs. The player step event is responsible for all of the player’s movements and actions, it calls the check input script and then runs the function based one which button has been pressed.



```
//player movement
if (oInput.rightKey) hspeed -= 3
if (oInput.leftKey) hspeed += 3
if (oInput.upKey) vspeed -= 3
if (oInput.downKey) vspeed += 3

```

This segment of code applies speed to the correct direction depending on which key is pressed (the check input script supports keyboard and gamepads).

Another important part of the player step event is handling the shooting mechanics.

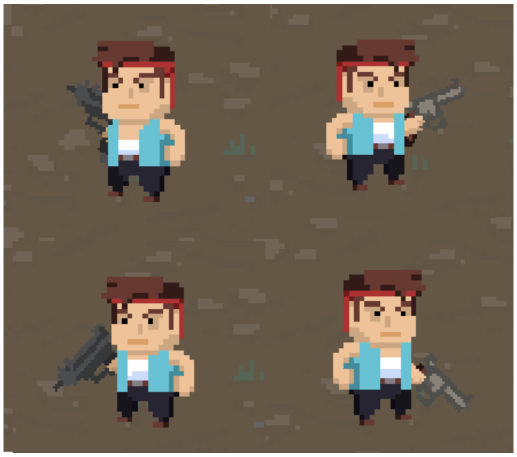


```
//shooting controls
if gamepad_button_check_pressed(0, gp_shoulderrb){
    clicked = 1;
}


if ammo[wep_type[wep]] = 0 and gamepad_button_check_pressed(0, gp_shoulderrb) and wep_type[wep] != 0{
    clicked = 0
    sound_play(sndEmpty)
}

if canShoot = 1 and ammo[wep_type[wep]] > 0{
    if wep_auto[wep] = 0 and clicked = 1{
        scrFire()
        clicked = 0
    }
if wep_auto[wep] = 1 and gamepad_button_check(0, gp_shoulderrb)
    scrFire()
}
```

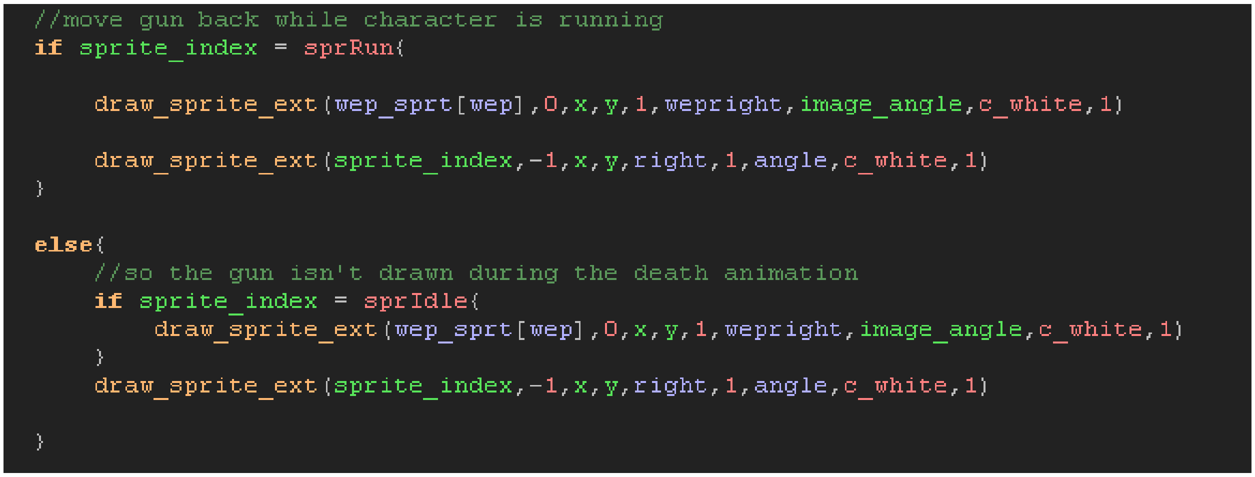
The shooting sections of this event first checks if the shoot button has been pressed, if the button has been pressed it then checks if the weapon which the player is holding has ammunition, if so the fire script will be called and if not the gun will play an empty noise which alerts the player that they have no ammunition.

Player Draw Event

A draw event governs what you see on the screen when the game is running. The player draw event is a section of the game in which I had a lot of trouble trying to achieve the 360 rotation of the players weapon to have it blend well with the game sprite seamlessly.



To achieve this effect, I used the GameMaker draw_sprit_ext function which draws sprites at a given position with customizable scaling, rotation and blending.



```
//move gun back while character is running
if sprite_index = sprRun{

    draw_sprite_ext(wep_sprt[wep],0,x,y,1,wepright,image_angle,c_white,1)

    draw_sprite_ext(sprite_index,-1,x,y,right,1,angle,c_white,1)
}

else{
    //so the gun isn't drawn during the death animation
    if sprite_index = sprIdle{
        draw_sprite_ext(wep_sprt[wep],0,x,y,1,wepright,image_angle,c_white,1)
    }
    draw_sprite_ext(sprite_index,-1,x,y,right,1,angle,c_white,1)

}
```

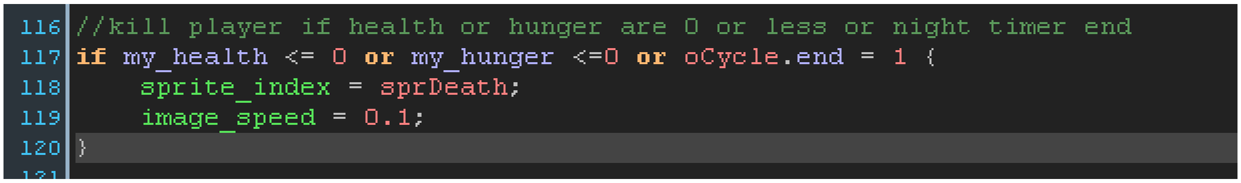
Basically, this segment of code decided where and how to draw the gun sprite by checking if the player is currently standing or running, what gun they are holding and what coordinates they are pointing it towards.

Kill Player Event

The game can be lost in three different ways:

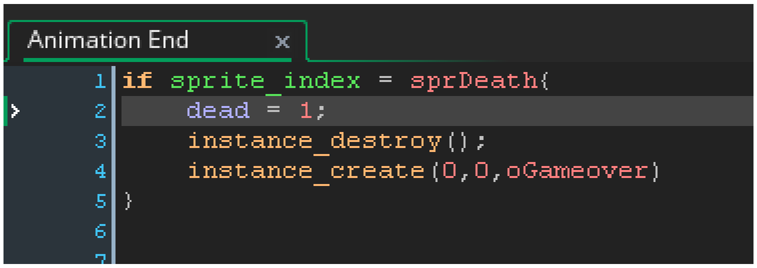
- The player runs out of HP
- The players hunger bar reaches 0
- The night timer runs out

All these variables are checked within the player step event and if one of the requirements are met the player's sprite changes to the death sprite.



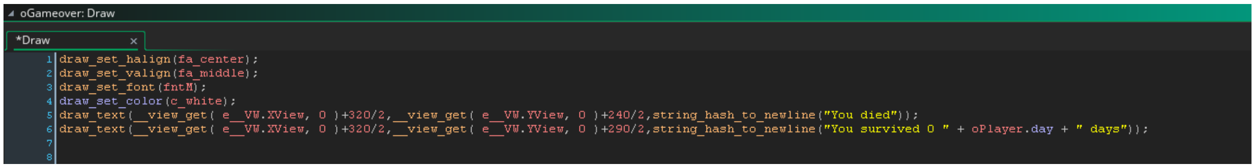
```
//kill player if health or hunger are 0 or less or night timer end
if my_health <= 0 or my_hunger <=0 or oCycle.end = 1 {
    sprite_index = sprDeath;
    image_speed = 0.1;
}
```

There is an end animation event in the player object which checks if the player's death animation ends and if so it will create an instance of the game over object.



```
if sprite_index = sprDeath{
    dead = 1;
    instance_destroy();
    instance_create(0,0,oGameover)
}
```

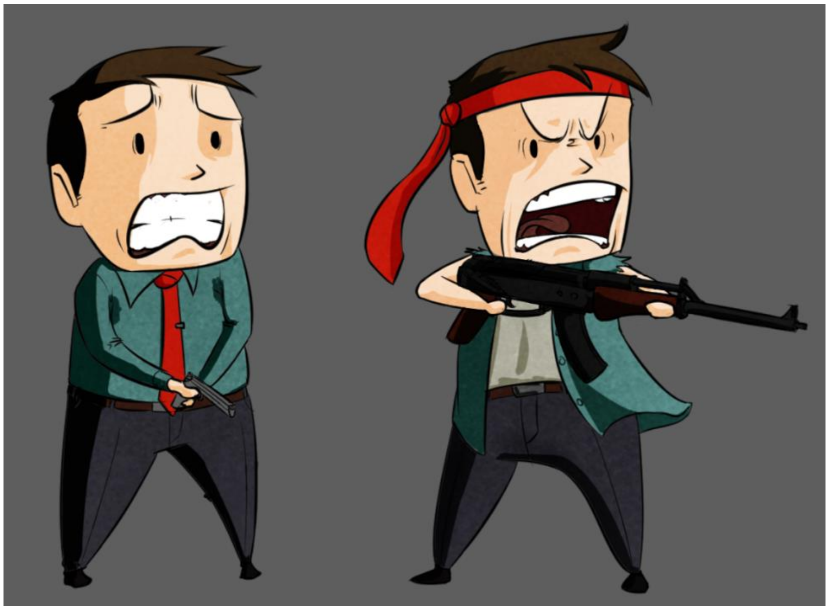
The game over object simply draws the users score and displays a message prompting them to press a button to continue.



```
draw_set_halign(fa_center);
draw_set_valign(fa_middle);
draw_set_font(fntM);
draw_set_color(c_white);
draw_text(__view_get( e__VW.XView, 0 )+320/2,__view_get( e__VW.YView, 0 )+240/2,string_hash_to_newline("You died"));
draw_text(__view_get( e__VW.XView, 0 )+320/2,__view_get( e__VW.YView, 0 )+290/2,string_hash_to_newline("You survived 0 " + oPlayer.day + " days"));
```

Player Design

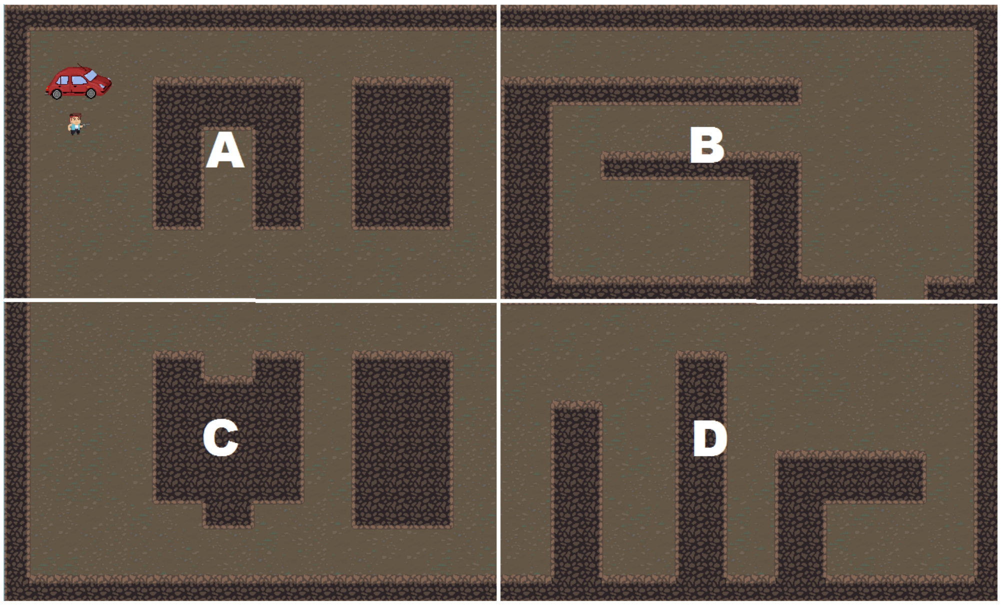
I wanted the theme of the game to be less serious than most games you see in the survival game genre and have a more cartoon like vibe, it was important for the main character to represent this theme.



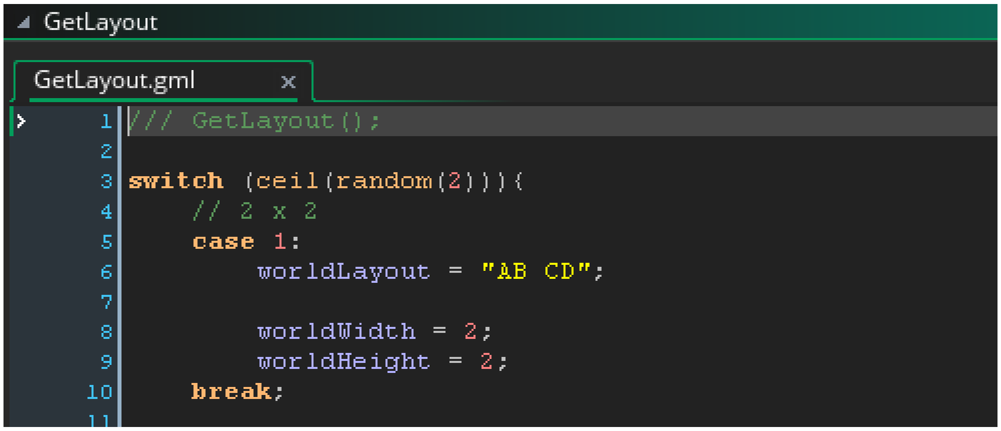
Above is the concept art for the potential main characters, I went with the second design as I felt it was the most unique and would match the theme of the game accurately.

2.3.2 Level Generator Script

One of the main features of this game is the random level generation, it is a crucial aspect to keep the game fresh and different with each playthrough. The implementation of this system was by far the hardest aspect of this project.

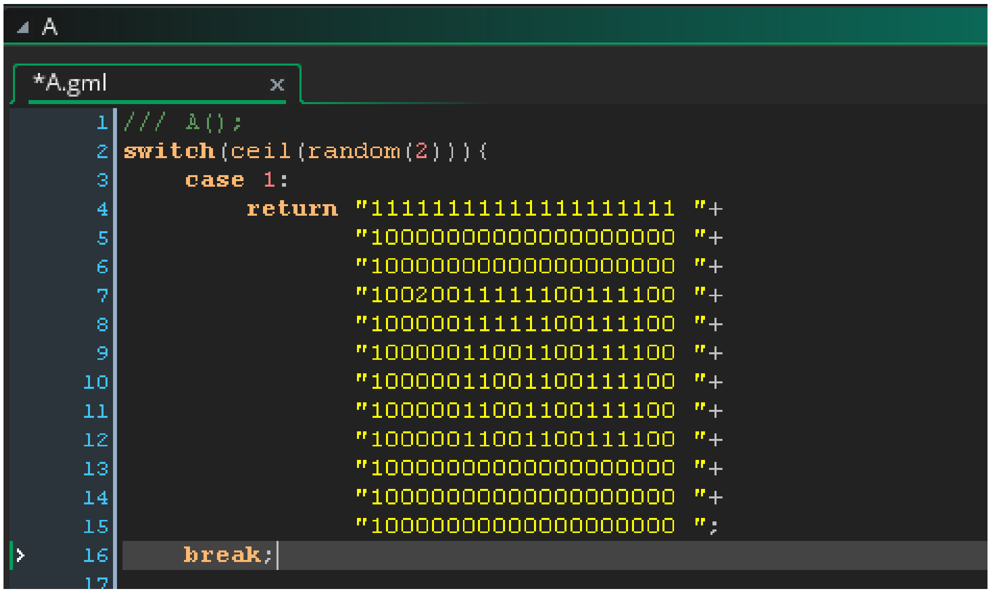


Above is an example of one of the many possible level structures that can be created by the random level generator (enemies and objects have been removed for demonstration purposes). Each level is made up by 4 sub areas which are laid out in a 2 X 2 grid. The level generator is made up by four different scripts. The first script is the GetLayout script that chooses which sub areas to piece together.



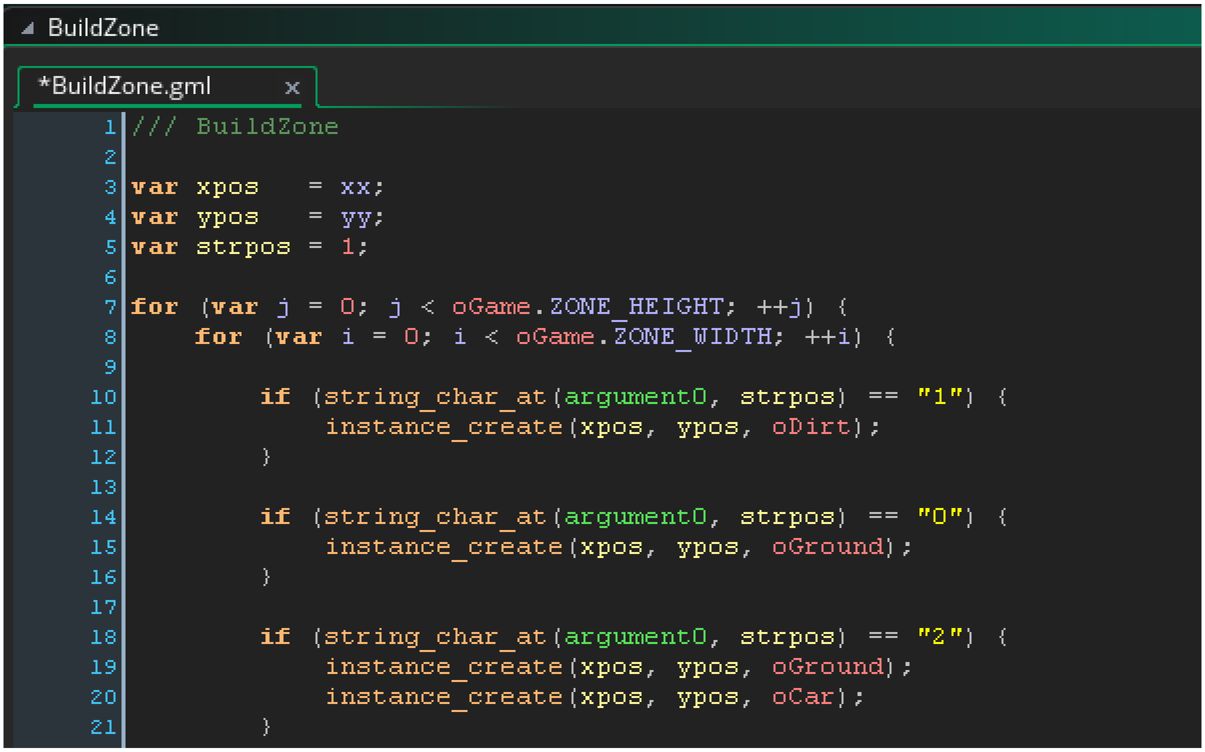
```
/// GetLayout();

switch (ceil(random(2))){
    // 2 x 2
    case 1:
        worldLayout = "AB CD";

        worldWidth = 2;
        worldHeight = 2;
    break;
```

The system can produce many types of layouts but for this game we only use 2 X 2, above you can see the code which gets the layout. A,B,C and D refer to different zones, a zone is simply a list of strings which the system reads and places objects down depending on which number it reads (1 = Block, 2 = Dirt, 3 = Weapon). Below is an example of a zone.



```
/// A();
switch(ceil(random(2))){
    case 1:
        return "11111111111111111111 "+
               "10000000000000000000 "+
               "10000000000000000000 "+
               "10020011111100111100 "+
               "10000111111100111100 "+
               "10000011001100111100 "+
               "10000011001100111100 "+
               "10000011001100111100 "+
               "10000011001100111100 "+
               "10000000000000000000 "+
               "10000000000000000000 "+
               "10000000000000000000 ";
    break;
```

Once the layout and zones have been read the system then goes to the BuildZone script which pieces together the area depending on the strings in each zone. Creating new zones is simple as you only need to input 1’s and 0’s to build the general layout, it is easy to see what the zone will look like as the string values are like a blueprint of the zone.



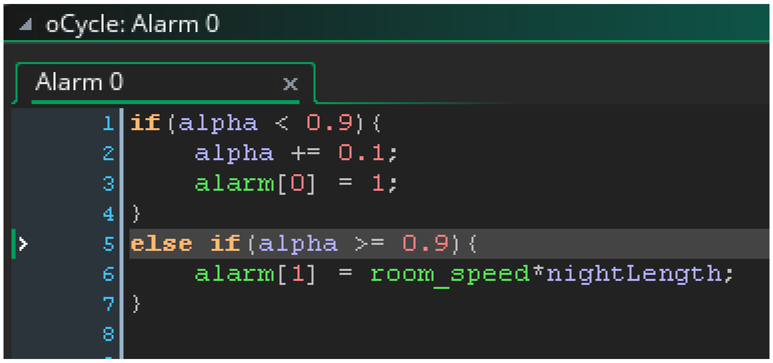
```
/// BuildZone

var xpos   = xx;
var ypos   = yy;
var strpos = 1;

for (var j = 0; j < oGame.ZONE_HEIGHT; ++j) {
    for (var i = 0; i < oGame.ZONE_WIDTH; ++i) {

        if (string_char_at(argument0, strpos) == "1") {
            instance_create(xpos, ypos, oDirt);
        }

        if (string_char_at(argument0, strpos) == "0") {
            instance_create(xpos, ypos, oGround);
        }

        if (string_char_at(argument0, strpos) == "2") {
            instance_create(xpos, ypos, oGround);
            instance_create(xpos, ypos, oCar);
        }
```

Here you can see the system gets the height and width of the level from the Game object and then proceeds loop through the zone strings and place object depending on the read values. It will be easy to add more variation to the levels, for example you could use for loops within the if statements to place random objects which would further expand upon the unpredictability of each level.

This system has a lot of potential and could be used to create a wider range of level layouts which is something to consider if I decided to expand upon this game.

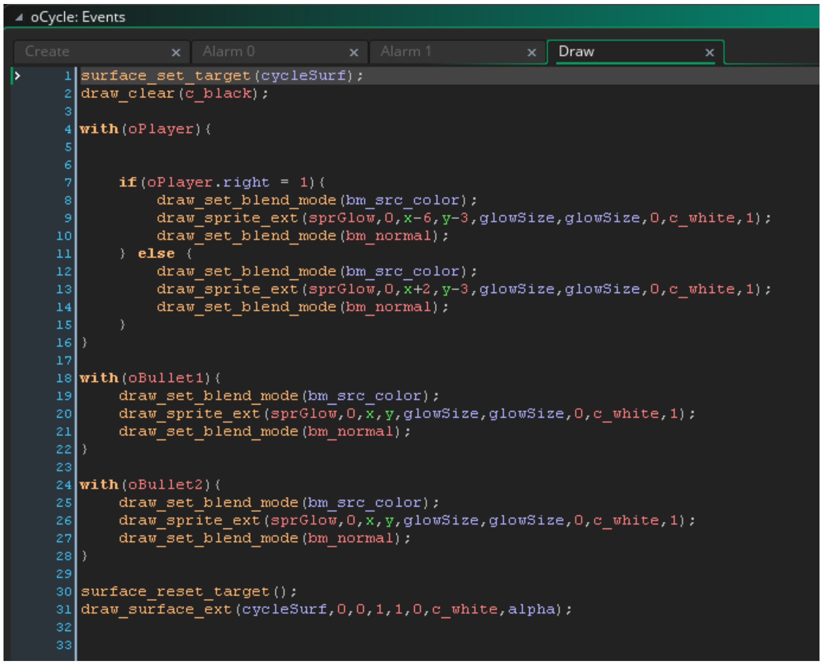
2.3.3 Day/Night Cycle Object

The cycle object is responsible for applying the night time effect to the game. This object has four events, two of them are similar alarms which either increase or decrease the light of the game to give the effect to the day changing to night and vice versa.



```
if(alpha < 0.9){
    alpha += 0.1;
    alarm[0] = 1;
}
else if(alpha >= 0.9){
    alarm[1] = room_speed*nightLength;
}
```

This alarm is responsible for changing the day to night, it increases the alpha value by a set number with each step to mimic the effect to a night transaction.

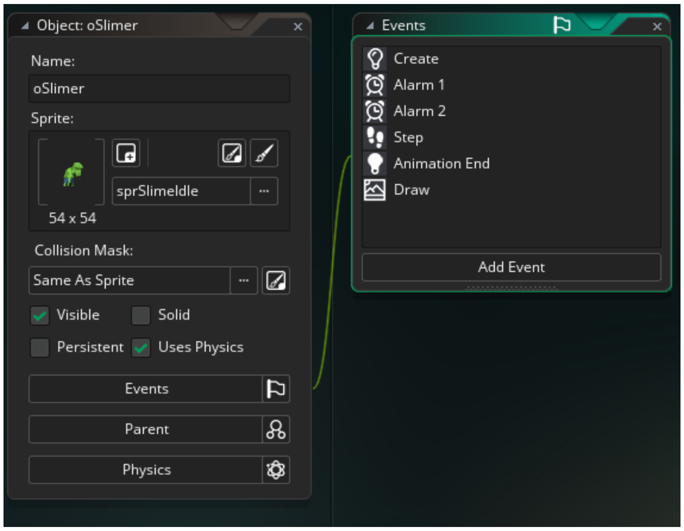


```
surface_set_target(cycleSurf);
draw_clear(c_black);

with(oPlayer){


    if(oPlayer.right = 1){
        draw_set_blend_mode(bm_src_color);
        draw_sprite_ext(sprGlow,0,x-6,y-3,glowSize,glowSize,0,c_white,1);
        draw_set_blend_mode(bm_normal);
    } else {
        draw_set_blend_mode(bm_src_color);
        draw_sprite_ext(sprGlow,0,x+2,y-3,glowSize,glowSize,0,c_white,1);
        draw_set_blend_mode(bm_normal);
    }
}

with(oBullet1){
    draw_set_blend_mode(bm_src_color);
    draw_sprite_ext(sprGlow,0,x,y,glowSize,glowSize,0,c_white,1);
    draw_set_blend_mode(bm_normal);
}

with(oBullet2){
    draw_set_blend_mode(bm_src_color);
    draw_sprite_ext(sprGlow,0,x,y,glowSize,glowSize,0,c_white,1);
    draw_set_blend_mode(bm_normal);
}

surface_reset_target();
draw_surface_ext(cycleSurf,0,0,1,1,0,c_white,alpha);
```

Above is the cycle object's draw event which draws the effect to the whole game except certain objects which have a glow to them such as the player object and the bullets. A glow is drawn around to player which allows the user to see some of the map, this effect makes the game harder to play which encourages the player to try reach their vehicle before it gets too dark.

2.3.4 Slimer Enemy Object

The Slimer is the most common enemy you will encounter, it is a basic zombie like enemy which charges and attacks the player when they are within a certain distance to them.



Object: oSlimer

Name: oSlimer

Sprite: sprSlimeIdle

54 x 54

Collision Mask: Same As Sprite

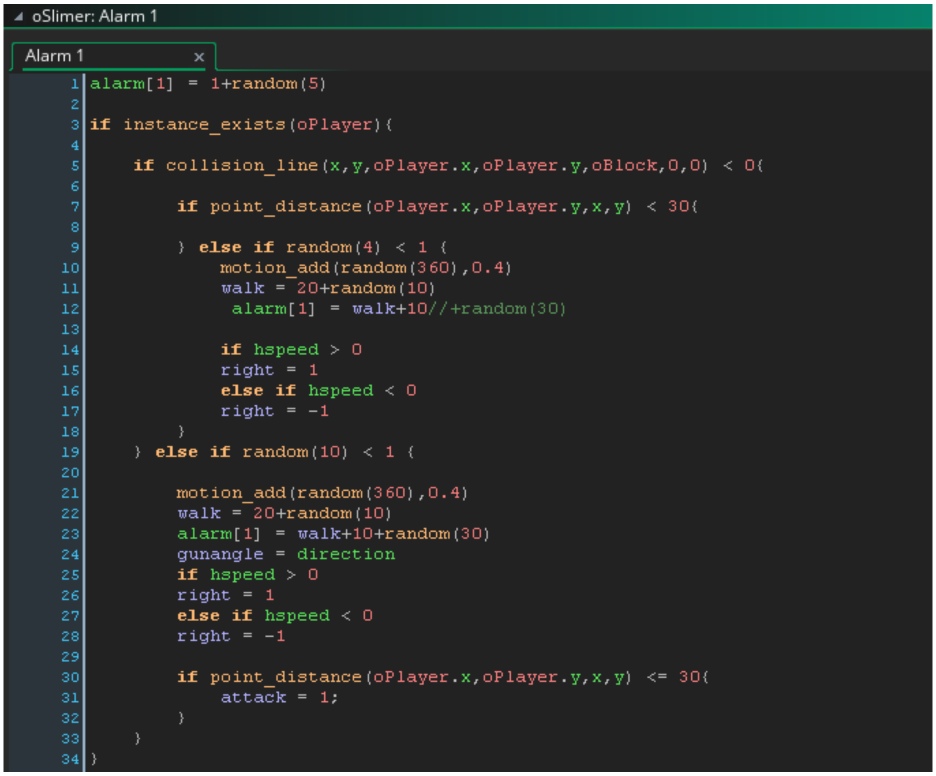
☑ Visible ☐ Solid

☐ Persistent ☑ Uses Physics

Events

- Create
- Alarm 1
- Alarm 2
- Step
- Animation End
- Draw

After the player, the enemies are the most complex objects in the game as they are responsible with keeping the game difficult and fun. How the Slimer moves is controlled within an alarm event.

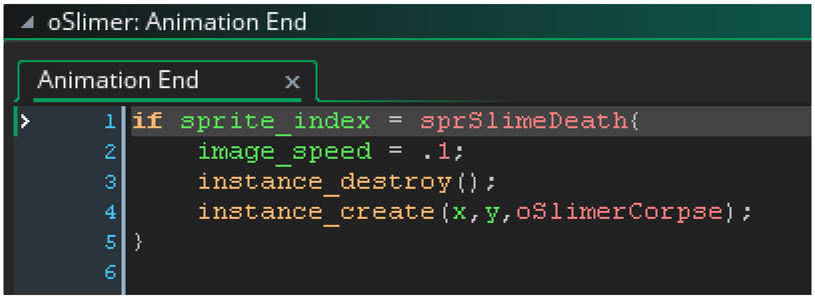


oSlimer: Alarm 1

Alarm 1

```
alarm[1] = 1+random(5)

if instance_exists(oPlayer){

    if collision_line(x,y,oPlayer.x,oPlayer.y,oBlock,0,0) < 0{

        if point_distance(oPlayer.x,oPlayer.y,x,y) < 30{

        } else if random(4) < 1 {
            motion_add(random(360),0.4)
            walk = 20+random(10)
             alarm[1] = walk+10//+random(30)

            if hspeed > 0
            right = 1
            else if hspeed < 0
            right = -1
        }
    } else if random(10) < 1 {

        motion_add(random(360),0.4)
        walk = 20+random(10)
        alarm[1] = walk+10+random(30)
        gunangle = direction
        if hspeed > 0
        right = 1
        else if hspeed < 0
        right = -1

        if point_distance(oPlayer.x,oPlayer.y,x,y) <= 30{
            attack = 1;
        }
    }
}
```

An alarm event triggers the enemy movement, I used an alarm that triggers over a random time to make the enemy movement more unpredictable. The enemy won't move unless the player is in their line of sight otherwise the enemy would travel the entire distance of the map at the start of the game. When the enemy is close to the player it will attack the player.

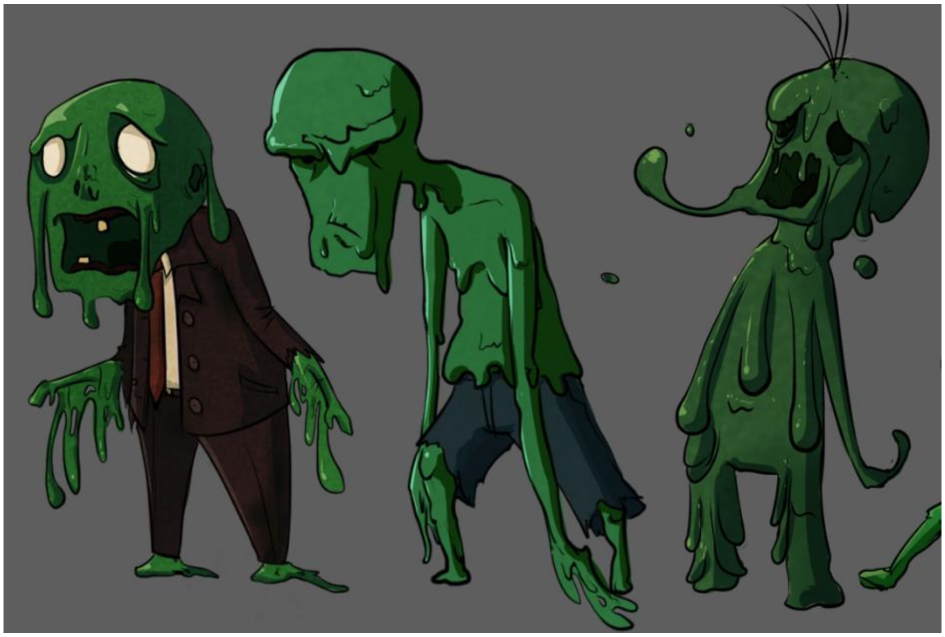


```
if sprite_index = sprSlimeDeath{
    image_speed = .1;
    instance_destroy();
    instance_create(x,y,oSlimerCorpse);
}
```

When the enemy dies, it leaves a corpse which is essentially green goo, I did this because I wanted the player to be able to retrace their steps by following the remains of the enemies they fought throughout the level.

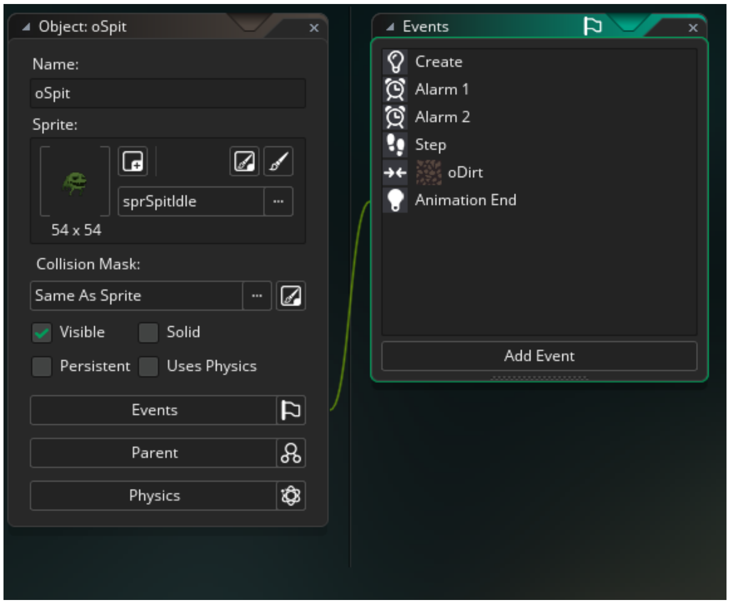
Enemy Design

I wanted to create a unique design for the enemies as I felt that the typical undead zombie like enemy was over used in many games, I had the idea to create green slime monsters as I thought they would be visually appealing and add some originality to the game



2.3.5 Shooter Enemy Object

The shooter enemy is a bit more complicated than the Slimer enemy as it doesn't run mindlessly towards the player, instead it keeps its distance from the player and fires towards him.



Name: oSpit

Sprite: sprSpitIdle

54 x 54

Collision Mask: Same As Sprite

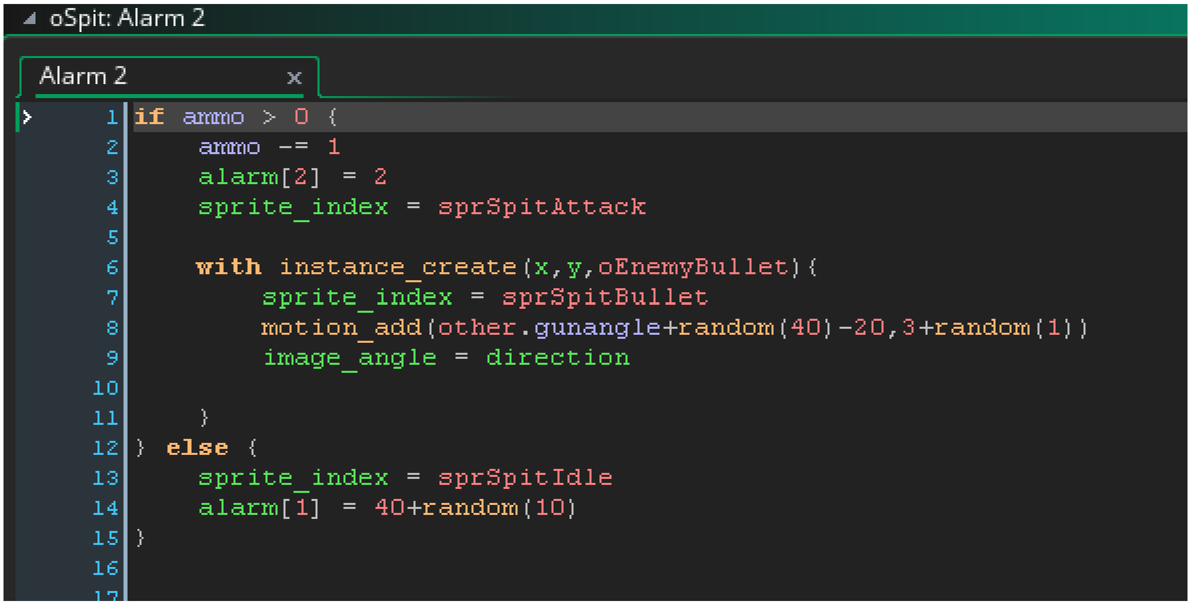
☑ Visible ☐ Solid

☐ Persistent ☐ Uses Physics

Events:

- Create
- Alarm 1
- Alarm 2
- Step
- oDirt
- Animation End

The shooter's movement event is like the Slimer except instead of running towards the player it runs away and shoots, this enemy was designed to increase the difficulty of each level dramatically as the shooter enemy can fire a barrage of bullets.



```
if ammo > 0 {
    ammo -= 1
    alarm[2] = 2
    sprite_index = sprSpitAttack

    with instance_create(x,y,oEnemyBullet){
        sprite_index = sprSpitBullet
        motion_add(other.gunangle+random(40)-20,3+random(1))
        image_angle = direction

    }
} else {
    sprite_index = sprSpitIdle
    alarm[1] = 40+random(10)
}
```

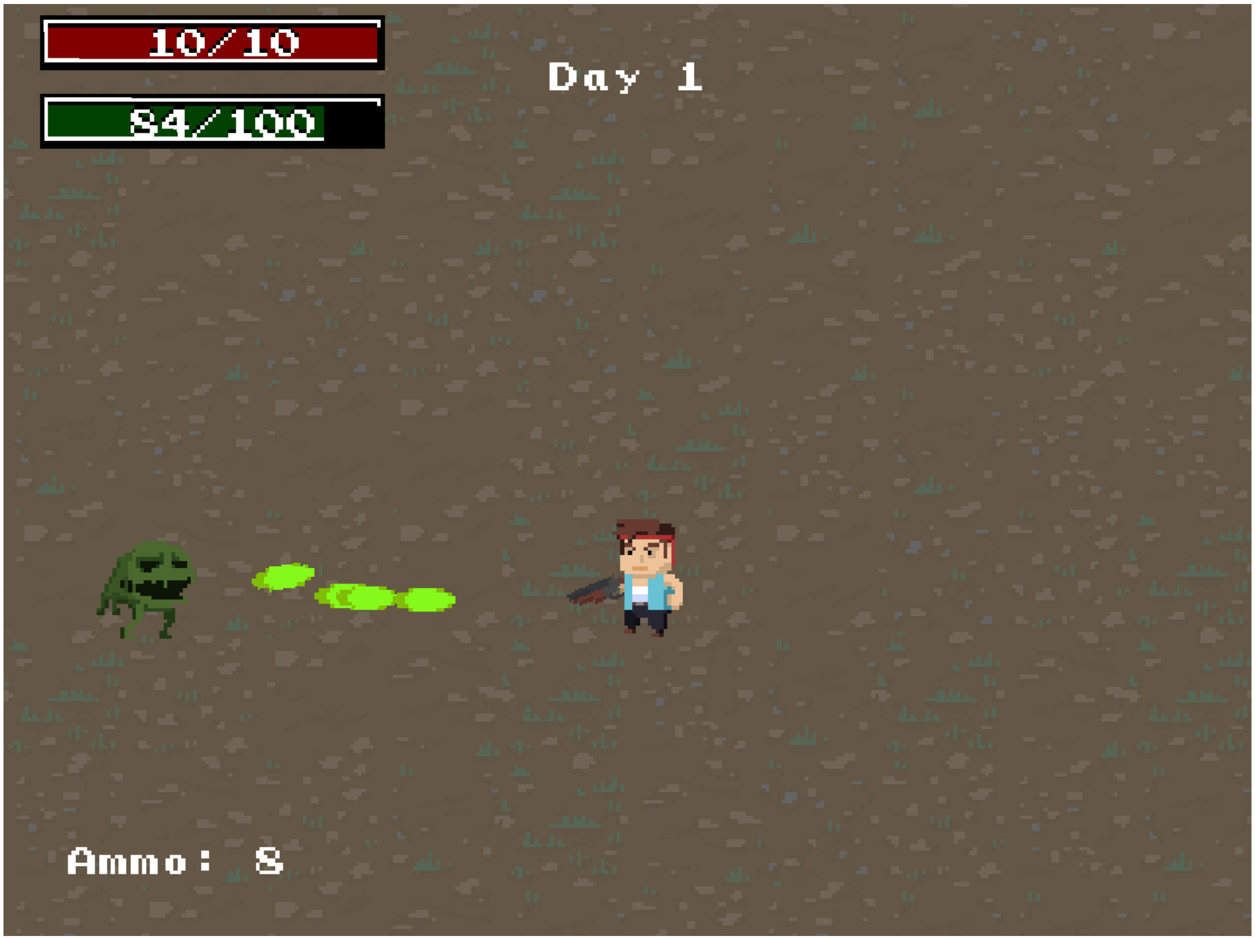
This above event checks if the enemy has ammo and if so it will create a bullet and send it towards the enemy, there are random variables applied to make sure the bullets are 100% accurate otherwise the game would be unfair to the player.

Enemy Design

It was important to ensure all the enemies had the same green slime-like style which meant they shape and size of each enemy must vary so they are easily distinguishable to the player.

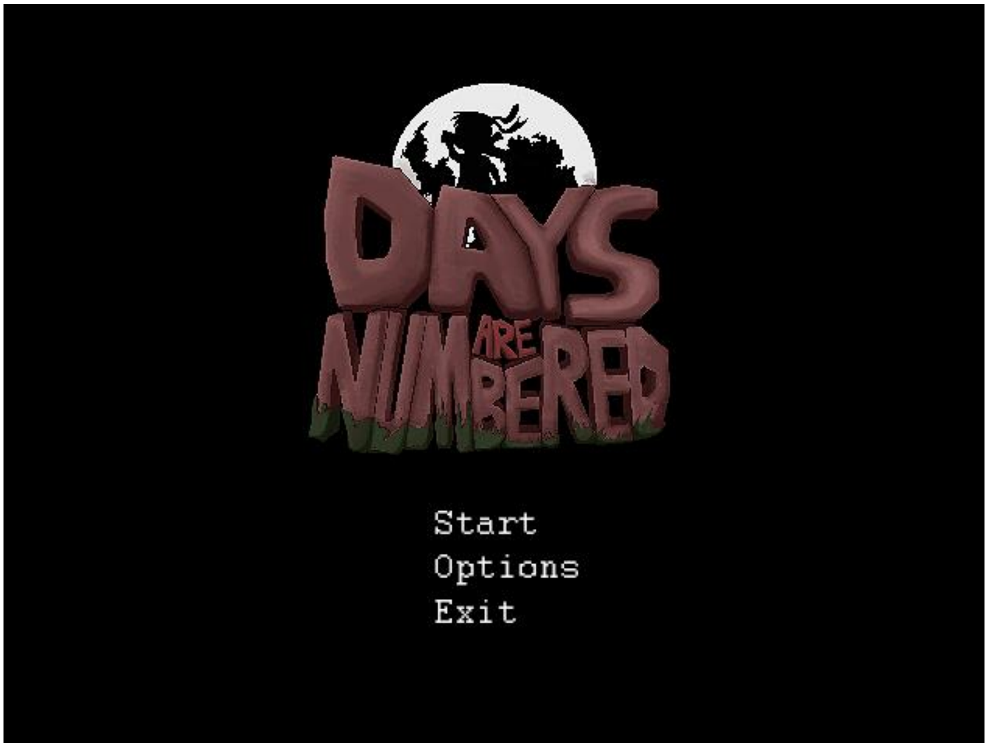
2.4 Graphical User Interface (GUI) Layout

It's important to keep the main GUI simple so it does not distract the player from the gameplay. There are 2 simple progress bars to display the player's health and hunger which are the most important values to keep track of, on the top centre is a day counter which is basically the players score in this game as the objective is to reach the highest day possible and in the bottom left hand corner the ammo of the equipped weapon is displayed.

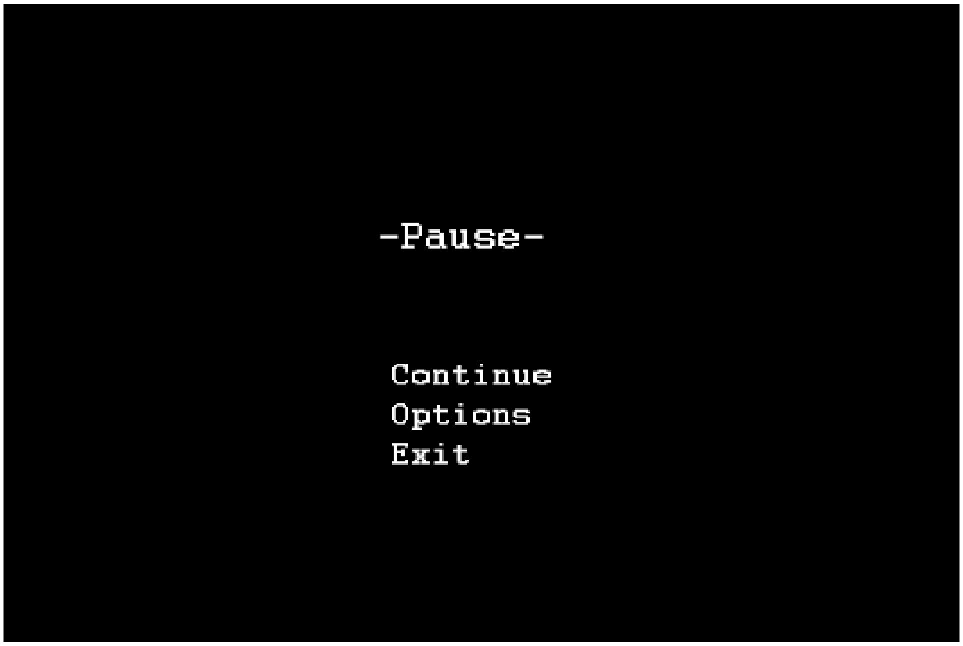


I made the colour of the tools bars a much darker shade of red and green than I had in the demo so it would not be as distracting to the player, I also decreased the size of the GUI items to make more room for the action.

The main menu is very simple, it features the games logo and presents the user with the 3 different choices to make.



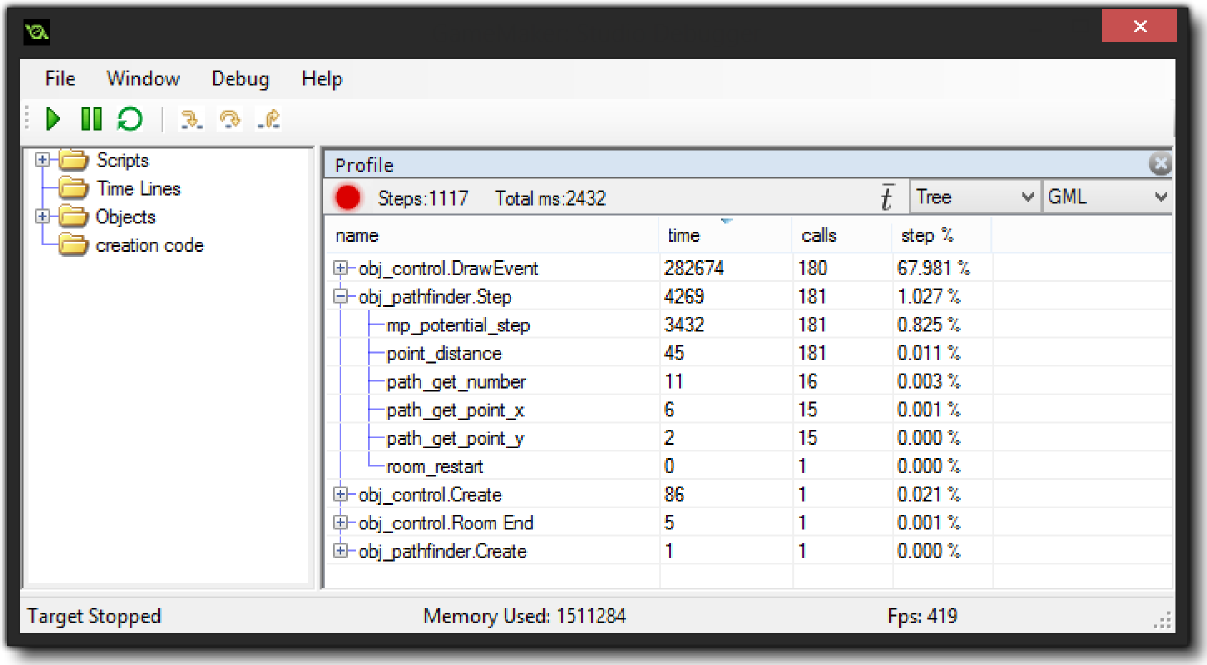
Like the main menu, the pause menu is also simple as it just displays the few options the player can make from the pause screen



2.5 Testing

Throughout the development of the game I have been extensively testing each component whenever a new feature has been added, I will need to take this further once I come closer to completion.

Game Maker has a debugger tool which is useful for tracking down errors and bugs as well as checking scripts run correctly. During my own testing, I run the game in debug mode which allows me to monitor the global performance of the game, it allows me to see each variable of every game object and monitor its performance.



Steps:1117 Total ms:2432

name	time	calls	step %
obj_control.DrawEvent	282674	180	67.981 %
obj_pathfinder.Step	4269	181	1.027 %
mp_potential_step	3432	181	0.825 %
point_distance	45	181	0.011 %
path_get_number	11	16	0.003 %
path_get_point_x	6	15	0.001 %
path_get_point_y	2	15	0.000 %
room_restart	0	1	0.000 %
obj_control.Create	86	1	0.021 %
obj_control.Room End	5	1	0.001 %
obj_pathfinder.Create	1	1	0.000 %

Target Stopped Memory Used: 1511284 Fps: 419

As well as testing the game myself through the development cycle it was important to conduct other tests with users to ensure that all the game requirements have been successfully met. It is important to get feedback from casual and avid gamers so that I can do my best to make the game appeal to both types of gamers.

Below are the prototype testing results provided by two students.

2.5.1 User Test – Robert Dillon

Robert is an avid gamer so he will be able to provide some good feedback as he has a good eye for detail in games and will help me know what a serious gamer expects from a game like this.

Event	Type	Issue	Suggestion
Gameplay	Polish	Object collisions does not feel fluid	Player collision detection doesn't seem to be pixel perfect as character bounces off walls without fully touching them sometimes.
UI	Clarity	The objective of the game not clear	The objective of the game needs to be clearer to the player from when the start a new game
Gameplay	Adjustment	Player moves to slow	The player speed should be sped up so it is possible to outrun the enemies
Gameplay	Adjustment	Enemy bullets hard to avoid	It would be good if you could destroy the enemy bullets by shooting at them
UI	Animation	Enemy animations too fast	The enemy attack and dying animations should be slowed down for better effect
Gameplay	Difficulty	Difficulty increase too slow	The difficulty increase of the game with each level should be more noticeable and dramatic
Gameplay	Polish	Player glow	The glow around the player at night should be slightly larger as it is too difficult to notice enemies running towards you
Sound	Variation	Repetitive music	The same song playing in every level is very repetitive it would be good to have different tracks every couple of levels
Gameplay	Polish	Enemy hit detection	It should be made clearer when an enemy takes damage, the enemy should flash for a second to show the player it took damage.

Roberts feedback was very beneficial to the game as he pointed out many changes that would improve the quality of the game as well as pointing out issues that went unnoticed throughout my development of the game.

2.5.2 User Test – Eamon Coleman

Eamon is a casual gamer who mostly plays big release games from time to time, he is a good candidate to help me understand the needs of more casual games.

Event	Type	Issue	Suggestion
Gameplay	Difficulty	Hard starting level	The first level should be made easier to give the player a chance to get familiar with the controls
UI	Clarity	Controls not clear	The only way to known the controls is by figuring them out yourself, there should be something to explain to the user what buttons do what
Gameplay	Difficulty	Enemies too fast	The enemies seem to be very fast it would be good to make them slower at the start then increase the speed later in the game
Gameplay	Difficulty	Finding car	It's hard to find your way back to the car at times it would be good if there was an arrow to direct you once you collect the fuel
UI	Clarity	Objective not clear	The overall object of the game is not made clear from the beginning
Gameplay	Difficulty	Not enough ammo	The player should start off with more ammo as it runs out quickly

Eamon's feedback was helpful and convinced me to tweak the difficulty of the game to make it more accessible to casual gamers.

All suggestions from both user test was considered and I used them to make the appropriate changes in hopes to make the game more accessible for both types of players.

2.6 Further Developments

Throughout of the development of this game I have learned a lot about game development and have become far more proficient in creating games through GameMaker. As of the current build of the game I believe it would make a good free to play game but lacks enough content to warrant the price of a regular independent video game.

I feel that the gameplay and mechanics of the game are solid and well-built but the game would benefit greatly with more art assets. It would be great to continue the development of this game with the help of an artist who could give the game even more character and help from more developers as the biggest limitation of the game is the development team which was comprised of only one developer over the span of 9 months.

With some more content and art added to the game I believe it could see success on a crowd funding platform like Go Fund Me or Kickstarter which would help fund further development of the game and make it robust enough to warrant a good price on Steam.

2.7 References

GameMaker Studio user manual:

https://docs.yoyogames.com/source/dadiospice/

GameMaker video tutorials used:

https://www.youtube.com/watch?v=VYghh9yWTsw&list=PL5KMAbZBx6C3_xMQb4TIcGLQeoXwu1Bwu&index=4

https://www.youtube.com/watch?v=mf95pjaEAL8&t=49s&list=PL5KMAbZBx6C3_xMQb4TIcGLQeoXwu1Bwu&index=5

https://www.youtube.com/watch?v=xU6w0oXRoz0&index=1&list=PL5KMAbZBx6C3_xMQb4TIcGLQeoXwu1Bwu

Video on game feel:

https://www.youtube.com/watch?v=216_5nu4aVQ

GameMaker testing blog:

http://www.yoyogames.com/blog/44

GameMaker community:

https://forum.yoyogames.com/index.php

3 Appendix

3.1 Project Proposal

3.1.1 Objective

The objective of the project is to create a fast paced action survival game with randomly generated levels to increase replay value.

The goal of the game is to survive as many days as you can, this won't be easy as you will need to collect food and defend yourself from hordes of enemies. The game will have a top down view perspective with 2d graphics and will feature the following:

- Randomly generated levels that will increase in difficulty as the game days' progress, this will increase the replay value of the game as the levels will be unpredictable and change each time you play.
- Day and night cycle that will change elements of the game, it will be more difficult for the player to navigate the level at night as their view will decrease when it gets darker, this will encourage the player to be more cautious when it becomes darker.
- Large selection of weapon and item types, weapons will spawn randomly on the map as the game progresses the chances of better weapons spawning will increase.
- Multiple enemy types that will get increasingly difficult as the game progresses, the speed and damage values of enemies will also increase as the game progresses. Harder enemy types will appear more frequently as the game progresses.
- Online Leaderboard as well as achievements that will unlock new weapons and items.

Each day in the game will last less than 5 minutes, this is because I wish to create a game that is fast paced and can be played in smaller sessions rather that a game which requires 50+ hours to complete. The game draws inspiration from other indie games such as Spelunky, FTL and Binding of Isaac.

3.1.2 Background

I’ve always enjoyed playing and learning about videos games, since I began studying Computer Science I have been learning to make games in my own spare time. I have never created a game as big as the one I am proposing as I never have the time due to being in work and college but the 4th year project is an excellent opportunity for me to finally do this. I feel that the 8-month period in which we have to develop this project is the perfect amount of time for me to create this game while learning and developing the skills I will need to achieve this.

The reason I chose this for my project is because I wish to one-day work as a game developer, having a complete game by the time I graduate will be an excellent addition to my portfolio which will increase my chances of landing a job in the games industry. I believe that if I produce this game to the best of my ability it will demonstrate my programming skills along with my knowledge and understanding of game design.

Indie gaming is becoming increasingly more popular and accessible for developers, it’s also becoming more and more common for smaller games with teams of only one or two people to become a success, I feel that this project would have the potential to do well or at least showcase my abilities in the indie gaming community.

Although the features of my game can be seen in many other games on the marketplace, the combination of all these features has not been done in a single game as of yet. I believe that all these features will complement each other nicely and result in a very fun and exciting gaming experience.

I am a huge fan of indie games and believe that a small team with a good idea can create better gaming experiences than big budget AAA games, my love of indie games is what inspired this idea and it will encourage me to create this game to the best of my ability.

3.1.3 Technical Approach

*Research**:*** I have already chosen the game engine in which I intend to work with but I still need to conduct some research on other tools to help me develop this game. I will need tools to help with the development of many aspects of the game including sound, effects, art, animation and more. Many tools exist for game development but finding the right one best suited to my game will be a difficult task

Literature: I have purchased "GameMaker Language: An In-Depth Guide" which is one of the most popular books on creating games using GameMaker, I plan to use this book to gain as much knowledge as I can about GameMaker language as it will be the primary coding language used throughout this project

Preparation: I have been using GameMaker for almost a year now so I am quite experienced developing with it, I will need to learn much more about the application as I have not made anything near as big as this project. I am subscribed to many GameMaker developers on YouTube who release tutorial videos regularly, I intend to learn as much as I can from these creators throughout the project development.

Requirement Capture: Once I have finalized all the mechanics and features of the game I will begin to lay out the complete list of functional requirements. I plan to conduct more research on the key non-functional requirements every good game must have to ensure the best performance from the finished product.

Prototyping: Once I have captured all the functional and non-functional requirements of the project I will create a simple prototype of the game to show off the main mechanics of the game for my mid-point presentation, this will be a rough version of the game just to demonstrate the idea and potential of the game

3.1.4 Technical Details

GameMaker Studio: This will be the main tool used in the development of my game. GameMaker accommodates the creation of cross-platform and multi-genre video games using drag and drop action sequences or a sandboxed scripting language known as Game Maker Language. GameMaker accommodates redistribution on multiple platforms such as:

- Windows
- Mac
- HTML5
- Android
- IOS

- Xbox One
- Playstation

The platform I intend to release on will be Windows, a release on another platform is a possibility depending on how well the development goes.

GML: GameMaker language is the primary scripting language used in GameMaker. Although GameMaker features I drag and drop systems I intend to only use the scripting language as GML is used to further enhance and control the design of the game through more conventional programming, as opposed to the drag and drop system.

Aseprite: Aseprite is a sprite editor and pixel art tool, this will be used to create all the art and animations that will be used in the game.

Third Party Tools: There is a wide range of third party tools created for GameMaker, I plan to use these only for the game aesthetics such as the animations and effects as I wish to be able to spend the majority of my time on game design and programming.

3.1.5 Evaluation

Iterative Testing: During the development period, I will insure that I have the time to test new features as they are added into the game. I will make sure to test each new feature rigorously to make sure there is no game breaking bugs within the new features

Regression Testing: Iterative testing is a good way to test if a new feature works when it is added but it is still important to set time aside to go back and re-test older features to ensure older additions have not been affected by newer features. I plan to do this less frequently than iterative testing.

End User Tests: I am very active in the GameMaker forums and plan to use to community as much as I can throughout the development of the game, every Tuesday people volunteer to test games that are currently in development. Having other developers test my game will be a great asset, I will also make sure to have non-developers also test it.

3.2 Monthly Journals

Reflective Journal

Student name: Adam Kearney

Programme: BSc in Computing

Stream: Software Development

Month: September

My Achievements

For the first month of the software project I spent the first couple of days trying to think of ideas that I would like to work on for the rest of the college year, I was trying to think of ideas that related to the modules I am taking this year but what I really wanted to do was make a game, I was worried I would not be allowed to make a game for my final year project because I didn't choose the gaming specialization for my final year but when I explained this to Eamon he said that I could work on a game once it would be approved after the project pitch. I really want to create a game as I hope to someday work as a game developer, if I can create this game for my project it would be a great addition to my portfolio.

I am really happy with the general idea for the game I hope to work on but I must wait for approval as the judges for my project pitch said they need to first evaluate the idea.

My Reflection

I feel that my project idea is a solid one and if executed well it could result in a game people would really enjoy. Looking back on the project pitch I feel that I could have sold it a little better but my nerves must have gotten the better of me during it. My main concern going into the pitch was that the judges would not approve the idea because I am not in the gaming stream, but I explained the reasons why I chose software over gaming.

Intended Changes

Next month hopefully I can start working away on the project, I am eager to start. I am told I will know whether the idea is approved or not on Monday, I am really hoping I get the approval.

Reflective Journal

Student name: Adam Kearney

Programme: BSc in Computing

Stream: Software Development

Month: October

My Achievements

I was delighted to have received approval from all three judges for my project pitch, I was worried there would be issues with me making a game and not being in the gaming stream, but that did not seem to be an issue.

Throughout the month I fleshed out the idea for my game so I could clearly describe it in my project proposal which was uploaded just before reading week. Having the proposal done before reading week was great as it allowed me to more time to start developing my game.

I spoke with my project supervisor before the reading week and she seemed to like the idea and gave me some good feedback and tips to help me get started.

Progress on the game has been going a lot smoother than I expected, I feel that I am close to having a very rough demo completed.

My Reflection

I feel like October was a success in regards to the project, I honestly wasn't expecting 3 judges to approve my project pitch and was even prepared to change the idea if it was declined.

I am very happy with the feedback I received from my first supervisor meeting and feel that these meetings will be a huge help throughout the remainder of the project development.

Intended Changes

Throughout the next month, I plan to focus on the requirement specification until it is due on the 11th, from there I will begin to spend most of my time coding the game. I hope to have some examples to show to my project supervisor by the end of this month.

Reflective Journal

Student name: Adam Kearney

Programme: BSc in Computing

Stream: Software Development

Month: November

My Achievements

For the first half of the month of November I spent most my software project time working on the requirement specification. The document took a lot longer than I expected as it required so much detail, but I managed to complete it in the end.

I intended to spend a lot of the time after the upload of the document working on the game itself, but I feel like I have not done as much as I should have because I've been spending most my free time working on all my other projects.

I have added a couple of new elements, but I will need a good few more to have a decent demo for my midpoint presentation.

My Reflection

I feel like the month of November has not run a smoothly as the previous two months as I'm feeling overwhelmed with the number of other projects at the moment. I think I should have managed my time a lot better, but this is something I will keep in mind going forward.

Intended Changes

Throughout the month of December I hope to be able to have a successful midpoint presentation with the work I have done so far, I plan to set a few days aside to focus on the game to make the demo as good as possible.

Reflective Journal

Student name: Adam Kearney

Programme: BSc in Computing

Stream: Software Development

Month: December

My Achievements

For the first half of December I spent most my time working on a demo to demonstrate the project for my midpoint presentation, I wanted to ensure that I could give the presentation supervisors a good idea of what the final product could look like. After creating the demo, I worked on my presentation slides and gave my presentation on the 20th.

My Reflection

I felt that my presentation went okay although I do think I may have rushed it. I was eager to start working on the game right away after the presentation but I think it is best that I hold off until the exams are over.

Intended Changes

When my exams are over I intend to start working away on the project again, once classes start back I will arrange a meeting with Lisa to get some feedback on the presentation and make any of the changes that are suggested from the presentation supervisors.

Supervisor Meetings

Date of Meeting: 13/12/2016

Items discussed: Presentation expectations and advice.

Reflective Journal

Student name: Adam Kearney

Programme: BSc in Computing

Stream: Software Development

Month: December

My Achievements

For the first half of December I spent most my time working on a demo to demonstrate the project for my midpoint presentation, I wanted to ensure that I could give the presentation supervisors a good idea of what the final product could look like. After creating the demo, I worked on my presentation slides and gave my presentation on the 20th.

My Reflection

I felt that my presentation went okay although I do think I may have rushed it. I was eager to start working on the game right away after the presentation but I think it is best that I hold off until the exams are over.

Intended Changes

When my exams are over I intend to start working away on the project again, once classes start back I will arrange a meeting with Lisa to get some feedback on the presentation and make any of the changes that are suggested from the presentation supervisors.

Supervisor Meetings

Date of Meeting: 13/12/2016

Items discussed: Presentation expectations and advice.

Reflective Journal

Student name: Adam Kearney

Programme: BSc in Computing

Stream: Software Development

Month: February

My Achievements

Throughout the month of February I spent most of my time coding the random level generator which is one of the main features and most difficult aspect of the project, I also did a small amount of work on some of the art of the game also.

My Reflection

I am very pleased with the progress on the level generation system as the main functionality of it is complete.

Intended Changes

The main functionality of the level generation system is complete but I still need to flesh it out and add more variety to the types of levels it creates and ensure all possible level variations make sense to the overall game. I will set up another supervisor meeting for the month of March to get some feedback on the progression of the project.

Reflective Journal

Student name: Adam Kearney

Programme: BSc in Computing

Stream: Software Development

Month: March

My Achievements

I have successfully fleshed out the level generation system to now include a much larger amount of possible levels. I have also fixed some various bugs I was having with the main character object.

My Reflection

I am happy that I have completed the level generation system which is one of the biggest features of this project. There is still quite a bit of work left to do in April and May once I have finished my other projects.

Intended Changes

Once I have finished my other projects and exams I intend to add the final touches to the game. I hope to be able to add a third enemy type and some extra weapons to increase the variety of gameplay elements, I also need to finish the design of the games logo and add music and sound effects.