Mind of Karl

Flynn Whelehan

24/04/2020

Table of Contents

Section I - Game Overview

Game Concept

Mind of Karl is going to be a maze game where Karl (the main character) finds himself trapped in different mazes when he goes to sleep and starts lucid dreaming. In his dreams he must escape the maze before he can wake up. Therefore, until Karl makes it out of a maze, he is trapped in a deep sleep forever…

The initial release of Mind of Karl will feature two of his dreams (two levels) that you must escape. Karl will encounter various obstacles and collect certain coins before he can escape the dreams.

These are the basic concepts of Mind of Karl.

Genre

Mind of Karl is going to be a top down (perspective), 2D maze game. This is an ideal genre to work on with the game engine that I am using to create it. It is also a great genre for the single player gameplay that the game includes. This is because part of the fun of a maze game is the exhilarating challenge of being trapped alone and having the challenge of escaping through only your actions – no one is there to help you.

It will follow a similar concept to other maze games like the well-known 'Pac-Man' (Pac-Man is a maze arcade game developed and released by Namco in 1980).

Target Platform

This game will be targeted for the PC platform. With PC, players can have full control over the performance of the game, meaning that graphics and gameplay elements aren't limited to the hardware specification of a console, for example. Adequate options should be included in the game that allows players to adjust settings to customise the performance of the game. This means that Mind of Karl will be accessible to players who don't have the latest and greatest gaming PC. For the players who are fortunate to have a powerful PC, they can run Mind of Karl with the highest graphics and intense gameplay elements. This would not be possible if Mind of Karl was targeted for console platforms.

PC gaming can sometimes create an unfair playing field. If one PC can run at ultra-high speeds and graphics, and one can run at low speeds and quality, this off-sets the balance in multiplayer gameplay. For example, if a PC is powerful enough to have a high rendering distance, and an enemy must set theirs to a small distance – who's going to see who first? The player with the expensive system of course. However, the difference in hardware would have to be on either side of the spectrum for it to be truly noticeable. This does not disregard the fact that players can have an advantage based on their system, though. However, Mind of Karl avoids this issue as there won't be any unbalanced gameplay as there is no PVP (player vs player) combat.

Gameplay can also be easier with a keyboard and mouse (which is not available on consoles) – things like inventory management can be tedious with a controller and ruin the user experience. A keyboard and mouse can be used to manage a variety of duties using many quick shortcut keys too. The mouse provides more precision and reaction than most console controllers would. I think that the keyboard input device will best accommodate for the quick movement and reactions that the game will require to avoid enemies. PC also supports many console controllers anyway, even more reason to choose PC over console.

Finally, I think that releasing a game on one console limits the audience immensely. Most games exclusive to one console are unheard of by players on other consoles.

There is potential that this game could be popular on mobile platforms too. That could be something to consider in the future if the game gains traction within the gaming industry and an opportunity for growth occurs.

Target Audience

Mind of Karl will appeal to most age ranges, as it does not include any adult content, (violence, etc) nor does it have any childish focus. I think that this is perfect as it gives the game the most potential and doesn't narrow down the potential players. The only age ranges that might not be interested is ages around 50+, as they do not typically game anyway. By keeping the game neutral and targeting all age ranges, it will also help to avoid criticism on, for example, how violent the game is.

Visual Style/Look and Feel of the game

This game will follow arcade-style graphics, with a 2D, top-down perspective. The terrain, objects, architecture, and overall environment will use pixelated graphics to maintain a neutral feel of the game.

The overall look and feel of the game will aim at an abstract, artificial, alienated experience. This is in efforts to try and make the player feel as if they are in Karl's lucid dreams (a dream where you gain some amount of control over dream characters, narrative and environment).

Project Scope

Number of locations/Levels

There will be two levels to the game. These will be two of Karl's dreams that he must escape. They include:

1. Karl finds himself in hell. He must collect all the coins and reach the portal while avoiding danger to escape.
2. Karl placed within an alien environment where he must collect all the coins and reach the portal while avoiding danger to escape

Number of NPC's

- Alien orbs
- Fireflies

Section II – Gameplay and Mechanics

Gameplay

The gameplay of Mind of Karl will consist of main objectives, side objectives and exploration.

Objectives of the game

So, the main objective of Mind of Karl is to escape his lucid dreams. To do this, you must collect the relevant coins and avoid danger. Once you have collected all the coins you must make it to the escape route. Each dream comes with its own narrative that you are faced with upon entering each level.

Aside from avoiding danger, there is the other side objective that you can collect power ups that can help you in escaping the dream.

Game Progression

As each dream (level) comes with its own narrative, the only element of game progression would be the journey from entering a dream and escaping it. Therefore, the rate of game progression is determined on how the player approaches the level. If they are successful and collect the coins quickly, the game will progress faster than if they are struggling to survive and collect the coins. Upon escaping the dream, Karl is freed from the dream and can live to see another day…until the next dream.

Missions/challenges/puzzles

When players start, they will have to become familiar with their environment and figure out how they are going to approach the dream. They will need to collect power ups and coins to escape and avoid the challenge of some dangerous NPCs.

There will be a challenge on each level to collect a special power up that is heavily guarded but gives the player a substantial boost in ability. Upon obtaining this power up, the player should be able to escape the dream much easier.

Mechanics & Physics

Core Mechanics

Mind of Karl will feature different core mechanics for the player to use in gameplay. These mechanics include:

- Walking

These mechanics are performed by the player with the peripheral device they are sing. This device may vary, as the game will support console controller like the Xbox One controller. However, the standard control layout using a keyboard and mouse will assign these controls to the mechanics:

- Walking – Up, Down, Left, Right arrows

Physics

The game will not use any advanced physics as seen in most modern games. As it is a simple, 2D maze game it will only use basic physics like colliding with physical obstacles and moving at realistic rates.

Section III – Story, Setting and Characters

Story/Narrative and Characters

Karl – Karl is the character you control and the focus of the game. When he goes to sleep at night, he finds himself in a lucid dream and cannot wake up until he has escaped the dream. He is a Swedish man of Viking descent and lives naturally in the Scandinavian mountain terrain. He is short and stocky and dresses in clothing made from animal products.

He has always believed that there is something more to his dreams as he is not aware of modern science and how lucid dreams work due to his traditional upbringing. As he is isolated in the mountains and surrounded by no one else but his family, no one is aware of his dreams. He is afraid to tell his family as they have very traditional beliefs and might think he is possessed.

Section IV – Levels

As Mind of Karl is based around different levels based upon different dreams, the levels are quite modular.

Game World

- ***Hell*** – Karl finds himself in hell. He must collect all the coins to escape through a portal while avoiding danger. It is dark and gloomy – much how you would imagine hell to be. Fire, red, evil. You will encounter fireflies (balls of fire) that will burn you on-contact.

- ***Alien Planet*** – Karl is placed within an alien environment where he must avoid capture and collect coins to escape. It will be a predominantly neon environment with alien terrain. You must collect the coins and reach the portal to return home. You will encounter alien orbs which patrol perimeters much like police would on earth. However, these aliens are used to other alien encounters and are ordered to attack with deadly force on-contact.

Section V – Interface

Heads up Display (HUD)

The HUD in Mind of Karl features important information for your survival and escape. At the top left of the display it will show your lives (how many times you can be become captured before you must restart the dream) and your score (how many coins you have collected to escape with).

Main Menu Screen

When Mind of Karl is first launched, players will be faced with a main menu screen. It acts as a start menu where players can enter the game. However, it will also let players navigate through the different screens available too. They can use an exit button to exit the game too.

Help Screen

This screen is accessible through the main menu screen and provides details on the game and how to play it, like control layout and FAQs.

Game over Screen

If you are unfortunate enough to be captured by any enemy within a dream, you will be faced with a game over screen with options to try again, return to the main menu, or exit the game. This indicates that you have been captured and you must start again.

Cheats and Easter Eggs

There is only one bonus feature in this game that acts as an Easter egg (as it is not clear what you will get from it until you pick it up). This is the special power up that is heavily guarded (mentioned earlier in the document).

Section VI - Artificial Intelligence

Friend & Foe (NPC's)

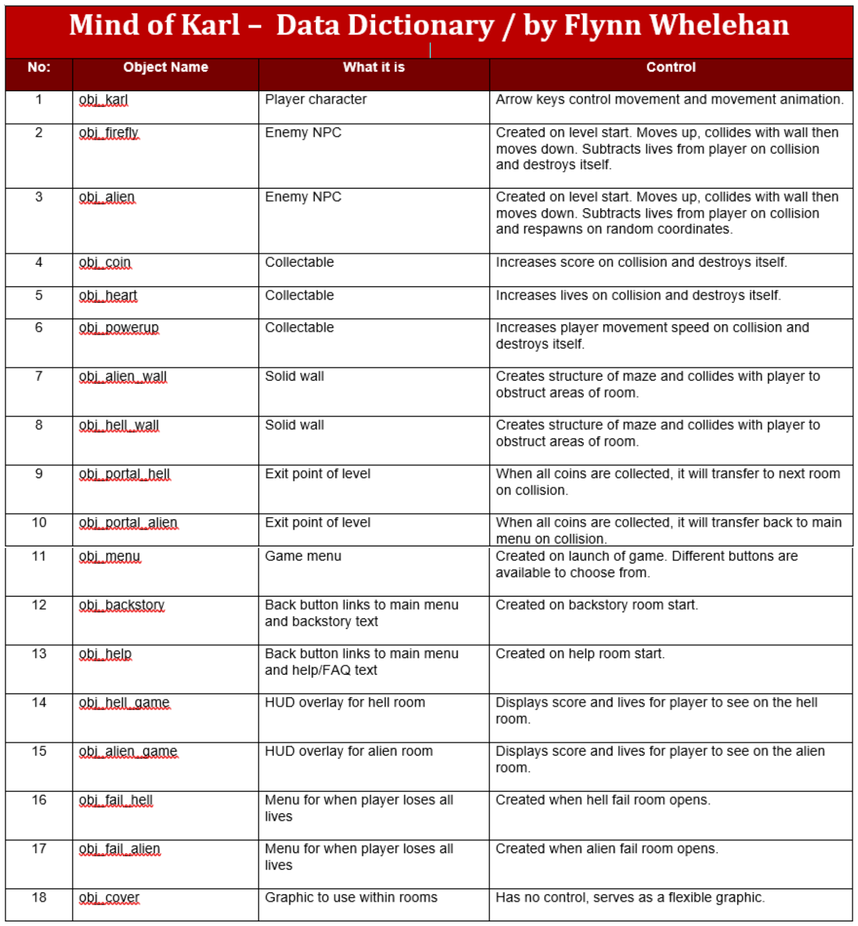
Enemy Characters

As the entire scenarios are created by Karl's brain, all the enemy characters are relevant to the game story through his imagination. Here are the enemy characters found within Mind of Karl:

- ***Fireflies*** – The fireflies are balls of fire that will patrol hell. They spawn naturally in hell. These fireflies will inflict burning damage to Karl if they touch him.
- ***Alien orbs*** – These alien orbs live on the alien planet of the second Mind of Karl level. They are green, territorial and are under orders to attack any alien life that comes to their planet. Avoid contact with them to avoid damage.

Section VIII – Data Dictionary, Code & Game Flow Chart

Data Dictionary



Mind of Karl – Data Dictionary / by Flynn Whelehan

No:	Object Name	What it is	Control
1	obj_karl	Player character	Arrow keys control movement and movement animation.
2	obj_firefly	Enemy NPC	Created on level start. Moves up, collides with wall then moves down. Subtracts lives from player on collision and destroys itself.
3	obj_alien	Enemy NPC	Created on level start. Moves up, collides with wall then moves down. Subtracts lives from player on collision and respawns on random coordinates.
4	obj_coin	Collectable	Increases score on collision and destroys itself.
5	obj_heart	Collectable	Increases lives on collision and destroys itself.
6	obj_powerup	Collectable	Increases player movement speed on collision and destroys itself.
7	obj_alien_wall	Solid wall	Creates structure of maze and collides with player to obstruct areas of room.
8	obj_hell_wall	Solid wall	Creates structure of maze and collides with player to obstruct areas of room.
9	obj_portal_hell	Exit point of level	When all coins are collected, it will transfer to next room on collision.
10	obj_portal_alien	Exit point of level	When all coins are collected, it will transfer back to main menu on collision.
11	obj_menu	Game menu	Created on launch of game. Different buttons are available to choose from.
12	obj_backstory	Back button links to main menu and backstory text	Created on backstory room start.
13	obj_help	Back button links to main menu and help/FAQ text	Created on help room start.
14	obj_hell_game	HUD overlay for hell room	Displays score and lives for player to see on the hell room.
15	obj_alien_game	HUD overlay for alien room	Displays score and lives for player to see on the alien room.
16	obj_fail_hell	Menu for when player loses all lives	Created when hell fail room opens.
17	obj_fail_alien	Menu for when player loses all lives	Created when alien fail room opens.
18	obj_cover	Graphic to use within rooms	Has no control, serves as a flexible graphic.

Pseudo Code

Selection

In order to stop the player from travelling to the next level without collecting all the coins (the game objective) I used selection code. The pseudo code for this is as follows:

```
If score equal to, or above 30
Then allow travel to room 2
```

Iteration

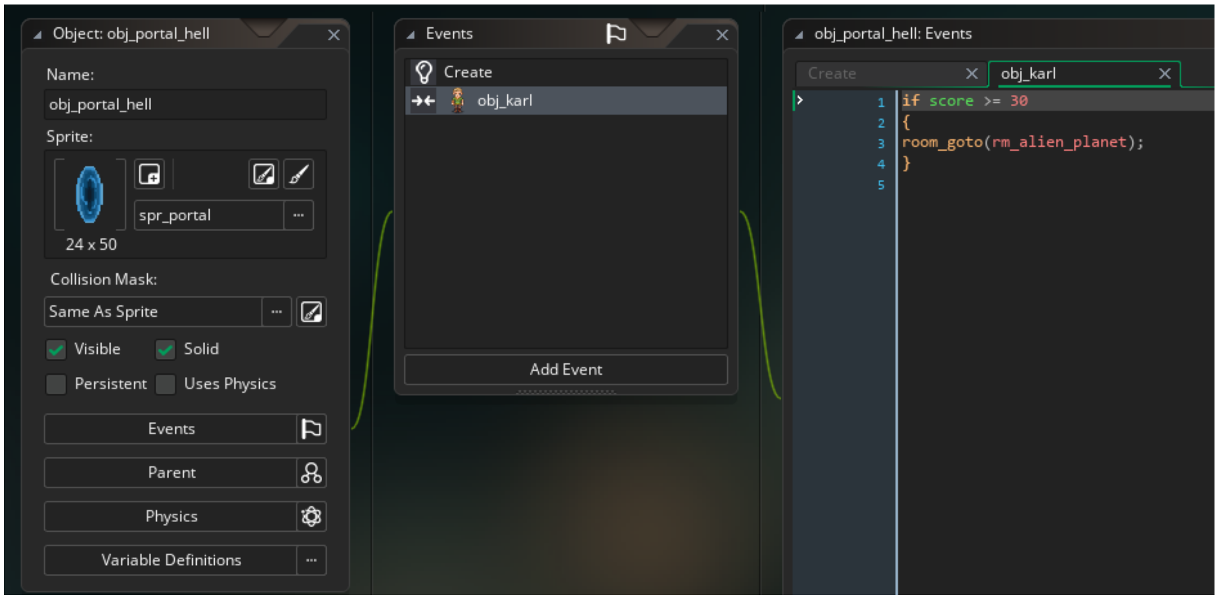
In order to make the second level (the alien planet) more difficult, I used iteration to respawn alien orbs after the collide with the player and inflict damage. Here is the pseudo code for this:

```
Randomise x and y coordinates
Until there are no collisions on the coordinates
Then destroy instance
Create instance of object at the x and y coordinates
```

Coding

Selection

Here is the code used for the portal unlock selection for level 1 (for level 2, it requires more score but uses the same code layout):



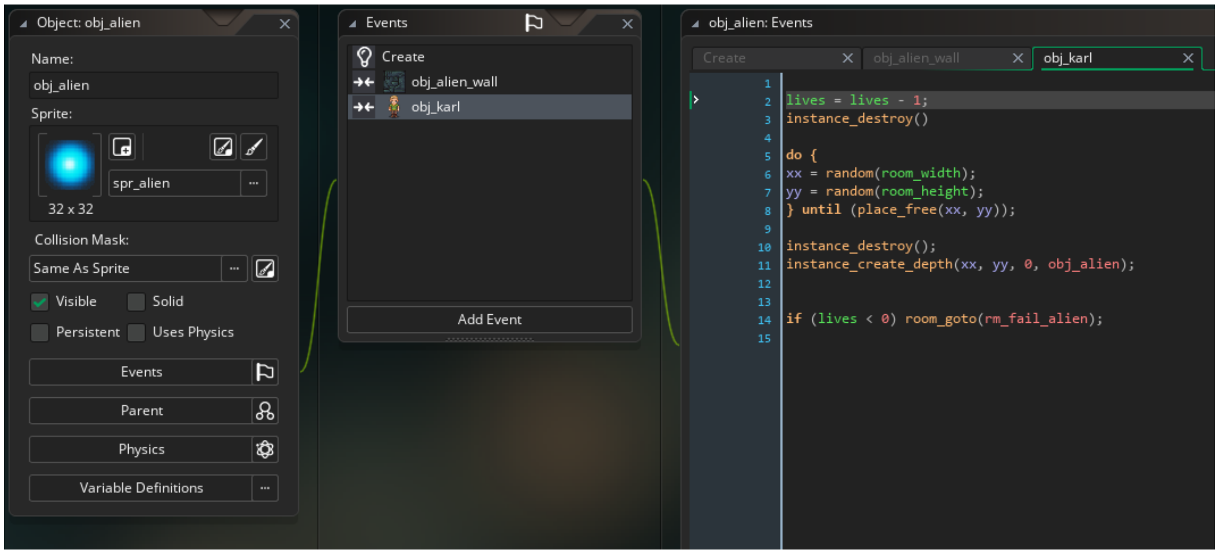
Name: obj_portal_hell
Sprite: spr_portal 24 x 50
Collision Mask: Same As Sprite
Visible ✓ Solid ✓ Persistent Uses Physics
Events: Create, obj_karl

obj_karl

```
if score >= 30
{
room_goto(rm_alien_planet);
}
```

Iteration

Here is the code used for the respawn iteration:



Name: obj_alien

Sprite: spr_alien 32 x 32

Collision Mask: Same As Sprite

✓ Visible ☐ Solid ☐ Persistent ☐ Uses Physics

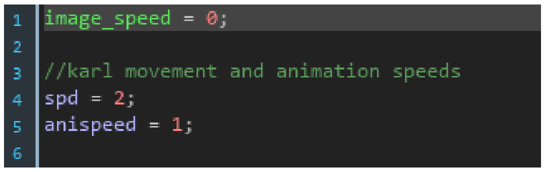
Events: Create, obj_alien_wall, obj_karl

Tabs: Create | obj_alien_wall | obj_karl

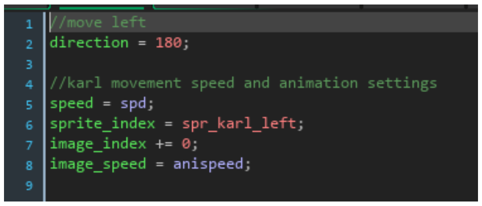
```

lives = lives - 1;
instance_destroy()

do {
xx = random(room_width);
yy = random(room_height);
} until (place_free(xx, yy));

instance_destroy();
instance_create_depth(xx, yy, 0, obj_alien);


if (lives < 0) room_goto(rm_fail_alien);

```

Rest of object code:

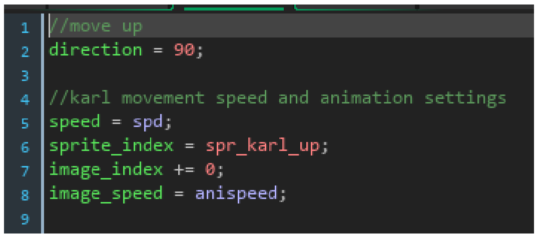
obj_karl



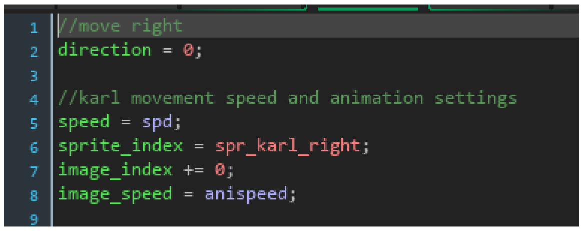
```
image_speed = 0;

//karl movement and animation speeds
spd = 2;
anispeed = 1;

```



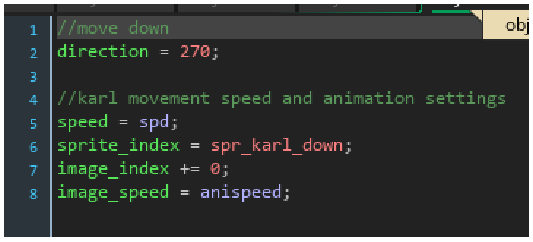
```
//move left
direction = 180;

//karl movement speed and animation settings
speed = spd;
sprite_index = spr_karl_left;
image_index += 0;
image_speed = anispeed;

```



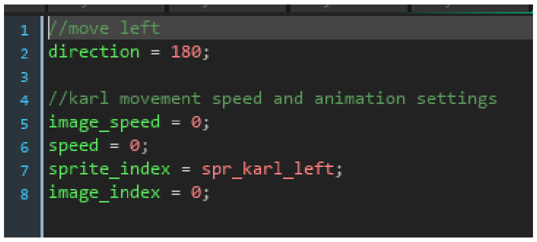
```
//move up
direction = 90;

//karl movement speed and animation settings
speed = spd;
sprite_index = spr_karl_up;
image_index += 0;
image_speed = anispeed;

```



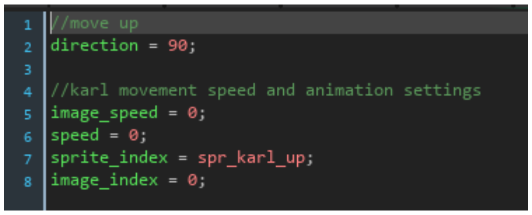
```
//move right
direction = 0;

//karl movement speed and animation settings
speed = spd;
sprite_index = spr_karl_right;
image_index += 0;
image_speed = anispeed;

```



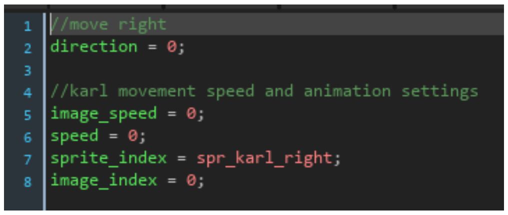
```
//move down
direction = 270;

//karl movement speed and animation settings
speed = spd;
sprite_index = spr_karl_down;
image_index += 0;
image_speed = anispeed;
```



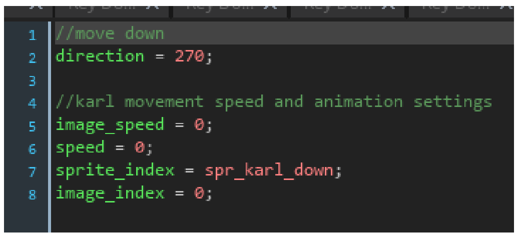
```
//move left
direction = 180;

//karl movement speed and animation settings
image_speed = 0;
speed = 0;
sprite_index = spr_karl_left;
image_index = 0;
```



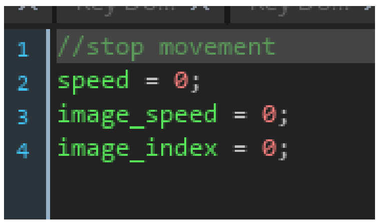
```
//move up
direction = 90;

//karl movement speed and animation settings
image_speed = 0;
speed = 0;
sprite_index = spr_karl_up;
image_index = 0;
```



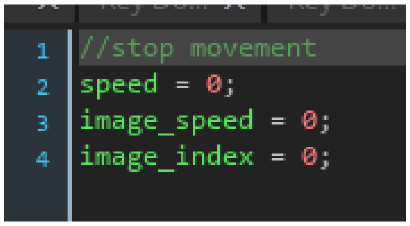
```
//move right
direction = 0;

//karl movement speed and animation settings
image_speed = 0;
speed = 0;
sprite_index = spr_karl_right;
image_index = 0;
```



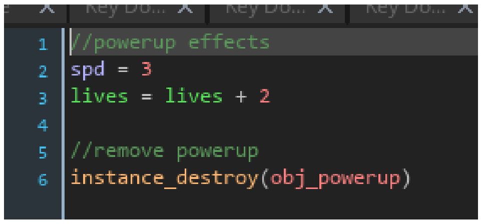
```
//move down
direction = 270;

//karl movement speed and animation settings
image_speed = 0;
speed = 0;
sprite_index = spr_karl_down;
image_index = 0;
```



```
//stop movement
speed = 0;
image_speed = 0;
image_index = 0;
```

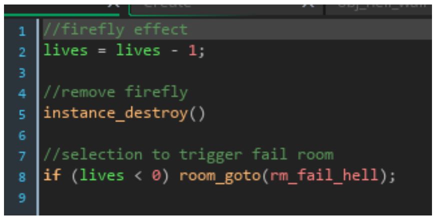


```
//stop movement
speed = 0;
image_speed = 0;
image_index = 0;
```

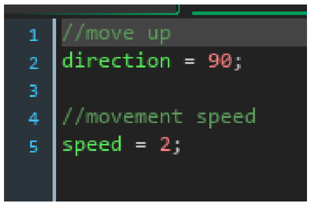


```
//powerup effects
spd = 3
lives = lives + 2

//remove powerup
instance_destroy(obj_powerup)
```

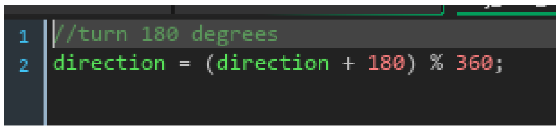
obj_firefly



```
//firefly effect
lives = lives - 1;

//remove firefly
instance_destroy()

//selection to trigger fail room
if (lives < 0) room_goto(rm_fail_hell);

```

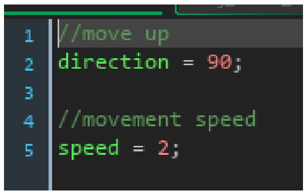


```
//move up
direction = 90;

//movement speed
speed = 2;
```

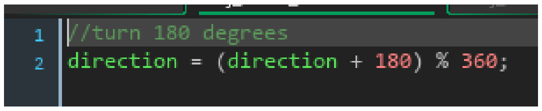


```
//turn 180 degrees
direction = (direction + 180) % 360;
```

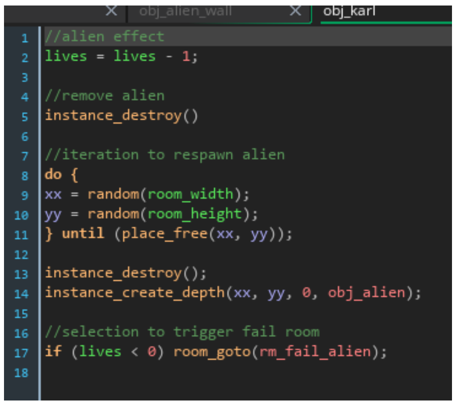
obj_alien



```
//move up
direction = 90;

//movement speed
speed = 2;
```

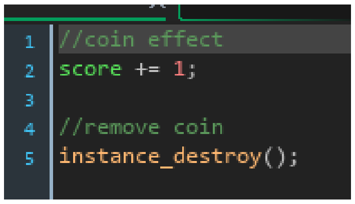


```
//turn 180 degrees
direction = (direction + 180) % 360;
```



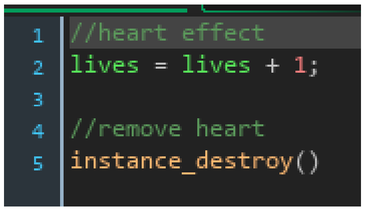
```
//alien effect
lives = lives - 1;

//remove alien
instance_destroy()

//iteration to respawn alien
do {
xx = random(room_width);
yy = random(room_height);
} until (place_free(xx, yy));

instance_destroy();
instance_create_depth(xx, yy, 0, obj_alien);

//selection to trigger fail room
if (lives < 0) room_goto(rm_fail_alien);

```

obj_coin



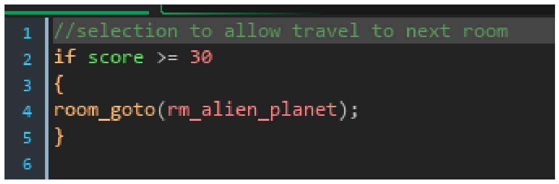
```
//coin effect
score += 1;

//remove coin
instance_destroy();
```

obj_heart



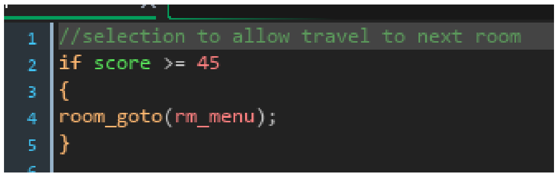
```
//heart effect
lives = lives + 1;

//remove heart
instance_destroy()
```

obj_portal_hell



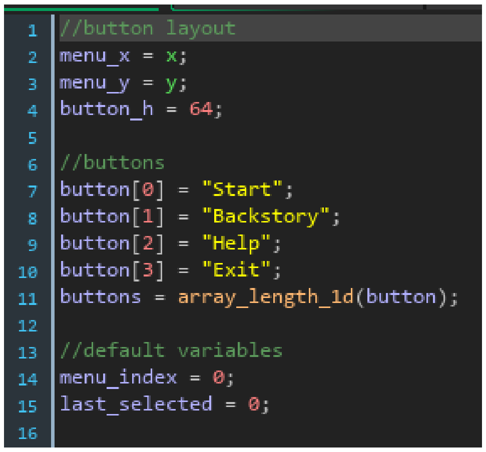
```
//selection to allow travel to next room
if score >= 30
{
room_goto(rm_alien_planet);
}

```

obj_portal_alien

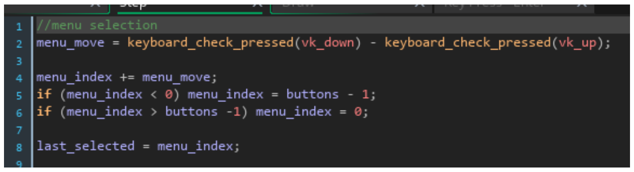


```
//selection to allow travel to next room
if score >= 45
{
room_goto(rm_menu);
}
```

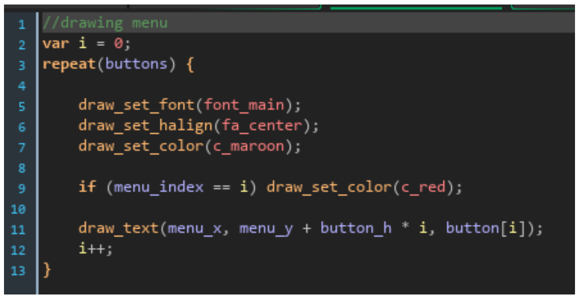
obj_menu



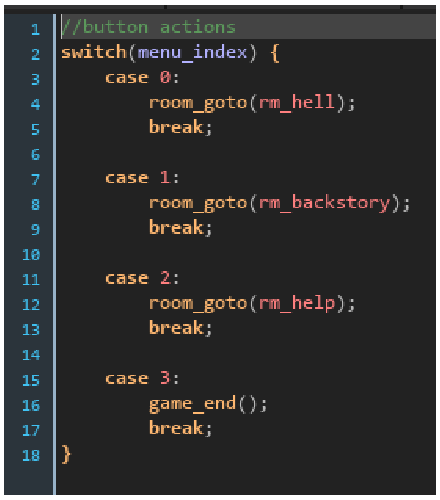
```
//button layout
menu_x = x;
menu_y = y;
button_h = 64;

//buttons
button[0] = "Start";
button[1] = "Backstory";
button[2] = "Help";
button[3] = "Exit";
buttons = array_length_1d(button);

//default variables
menu_index = 0;
last_selected = 0;

```



```
//menu selection
menu_move = keyboard_check_pressed(vk_down) - keyboard_check_pressed(vk_up);

menu_index += menu_move;
if (menu_index < 0) menu_index = buttons - 1;
if (menu_index > buttons -1) menu_index = 0;

last_selected = menu_index;
```

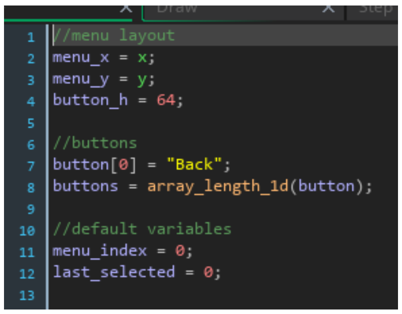


```
//drawing menu
var i = 0;
repeat(buttons) {

    draw_set_font(font_main);
    draw_set_halign(fa_center);
    draw_set_color(c_maroon);

    if (menu_index == i) draw_set_color(c_red);

    draw_text(menu_x, menu_y + button_h * i, button[i]);
    i++;
}
```

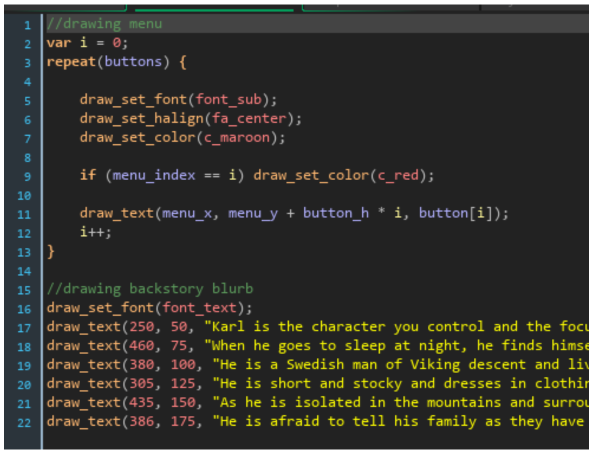


```
//button actions
switch(menu_index) {
    case 0:
        room_goto(rm_hell);
        break;

    case 1:
        room_goto(rm_backstory);
        break;

    case 2:
        room_goto(rm_help);
        break;

    case 3:
        game_end();
        break;
}
```

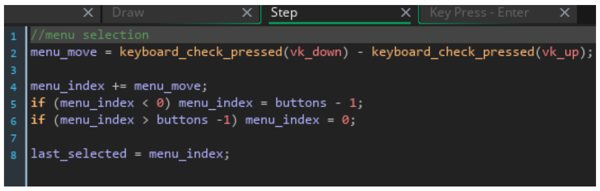
obj_backstory



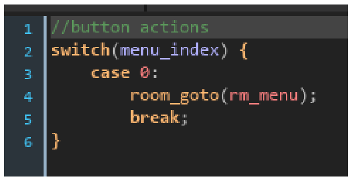
```
//menu layout
menu_x = x;
menu_y = y;
button_h = 64;

//buttons
button[0] = "Back";
buttons = array_length_1d(button);

//default variables
menu_index = 0;
last_selected = 0;

```



```
//drawing menu
var i = 0;
repeat(buttons) {

    draw_set_font(font_sub);
    draw_set_halign(fa_center);
    draw_set_color(c_maroon);

    if (menu_index == i) draw_set_color(c_red);

    draw_text(menu_x, menu_y + button_h * i, button[i]);
    i++;
}

//drawing backstory blurb
draw_set_font(font_text);
draw_text(250, 50, "Karl is the character you control and the focu
draw_text(460, 75, "When he goes to sleep at night, he finds himse
draw_text(380, 100, "He is a Swedish man of Viking descent and liv
draw_text(305, 125, "He is short and stocky and dresses in clothir
draw_text(435, 150, "As he is isolated in the mountains and surrou
draw_text(386, 175, "He is afraid to tell his family as they have
```

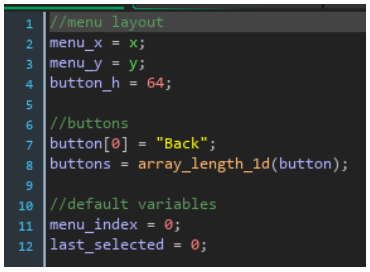


```
//menu selection
menu_move = keyboard_check_pressed(vk_down) - keyboard_check_pressed(vk_up);

menu_index += menu_move;
if (menu_index < 0) menu_index = buttons - 1;
if (menu_index > buttons -1) menu_index = 0;

last_selected = menu_index;
```

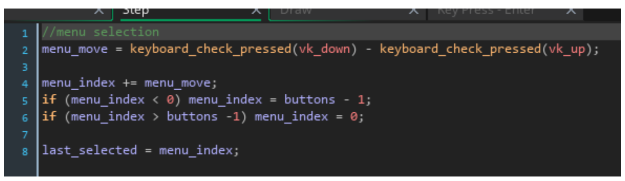


```
//button actions
switch(menu_index) {
    case 0:
        room_goto(rm_menu);
        break;
}
```

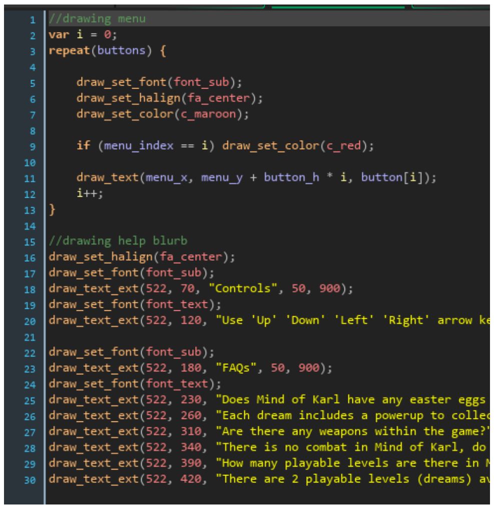
obj_help



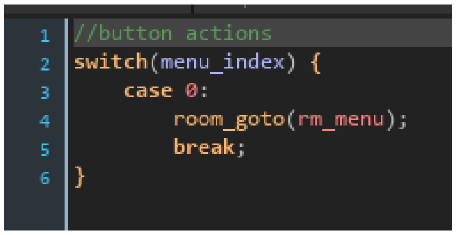
```
//menu layout
menu_x = x;
menu_y = y;
button_h = 64;

//buttons
button[0] = "Back";
buttons = array_length_1d(button);

//default variables
menu_index = 0;
last_selected = 0;
```



```
//menu selection
menu_move = keyboard_check_pressed(vk_down) - keyboard_check_pressed(vk_up);

menu_index += menu_move;
if (menu_index < 0) menu_index = buttons - 1;
if (menu_index > buttons -1) menu_index = 0;

last_selected = menu_index;
```

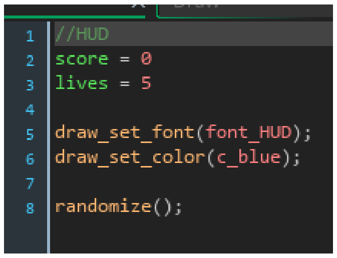


```
//drawing menu
var i = 0;
repeat(buttons) {

    draw_set_font(font_sub);
    draw_set_halign(fa_center);
    draw_set_color(c_maroon);

    if (menu_index == i) draw_set_color(c_red);

    draw_text(menu_x, menu_y + button_h * i, button[i]);
    i++;
}

//drawing help blurb
draw_set_halign(fa_center);
draw_set_font(font_sub);
draw_text_ext(522, 70, "Controls", 50, 900);
draw_set_font(font_text);
draw_text_ext(522, 120, "Use 'Up' 'Down' 'Left' 'Right' arrow ke

draw_set_font(font_sub);
draw_text_ext(522, 180, "FAQs", 50, 900);
draw_set_font(font_text);
draw_text_ext(522, 230, "Does Mind of Karl have any easter eggs
draw_text_ext(522, 260, "Each dream includes a powerup to collec
draw_text_ext(522, 310, "Are there any weapons within the game?"
draw_text_ext(522, 340, "There is no combat in Mind of Karl, do
draw_text_ext(522, 390, "How many playable levels are there in M
draw_text_ext(522, 420, "There are 2 playable levels (dreams) av
```

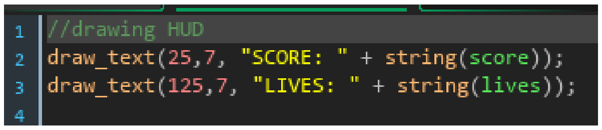


```
//button actions
switch(menu_index) {
    case 0:
        room_goto(rm_menu);
        break;
}
```

obj_hell_game

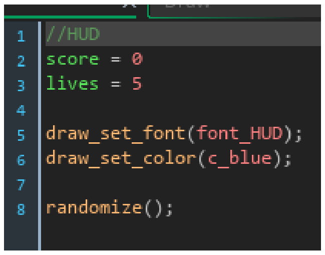


```
//HUD
score = 0
lives = 5

draw_set_font(font_HUD);
draw_set_color(c_blue);

randomize();
```

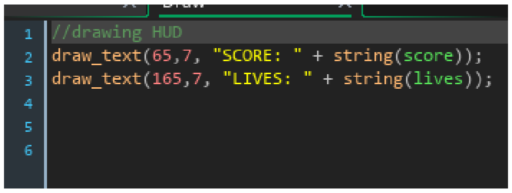


```
//drawing HUD
draw_text(25,7, "SCORE: " + string(score));
draw_text(125,7, "LIVES: " + string(lives));
```

obj_alien_game

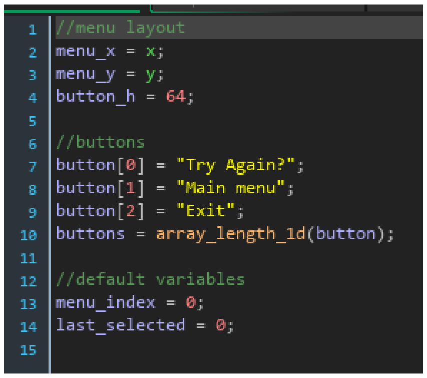


```
//HUD
score = 0
lives = 5

draw_set_font(font_HUD);
draw_set_color(c_blue);

randomize();
```

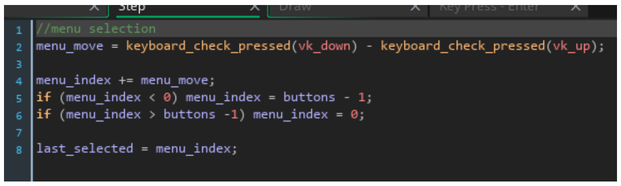


```
//drawing HUD
draw_text(65,7, "SCORE: " + string(score));
draw_text(165,7, "LIVES: " + string(lives));
```

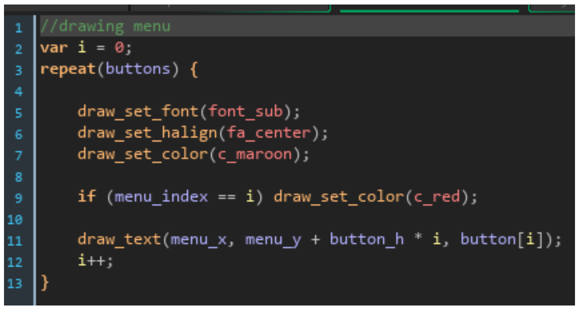
obj_fail_hell



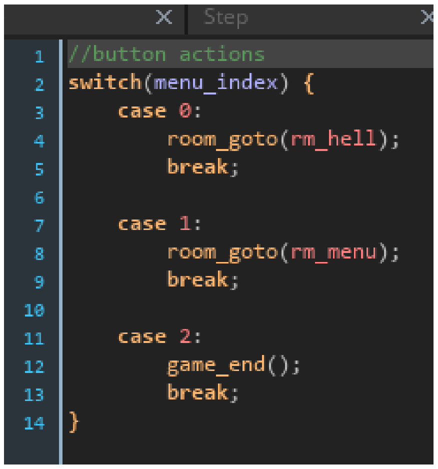
```
//menu layout
menu_x = x;
menu_y = y;
button_h = 64;

//buttons
button[0] = "Try Again?";
button[1] = "Main menu";
button[2] = "Exit";
buttons = array_length_1d(button);

//default variables
menu_index = 0;
last_selected = 0;
```



```
//menu selection
menu_move = keyboard_check_pressed(vk_down) - keyboard_check_pressed(vk_up);

menu_index += menu_move;
if (menu_index < 0) menu_index = buttons - 1;
if (menu_index > buttons -1) menu_index = 0;

last_selected = menu_index;
```

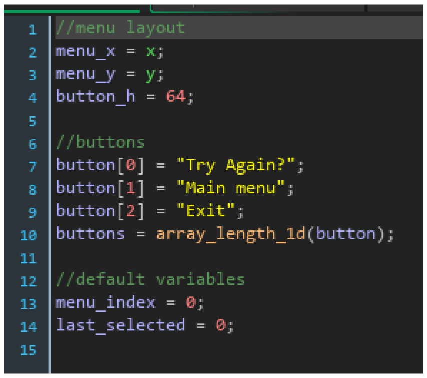


```
//drawing menu
var i = 0;
repeat(buttons) {

    draw_set_font(font_sub);
    draw_set_halign(fa_center);
    draw_set_color(c_maroon);

    if (menu_index == i) draw_set_color(c_red);

    draw_text(menu_x, menu_y + button_h * i, button[i]);
    i++;
}
```

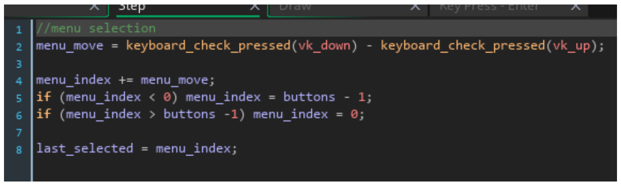


```
//button actions
switch(menu_index) {
    case 0:
        room_goto(rm_hell);
        break;

    case 1:
        room_goto(rm_menu);
        break;

    case 2:
        game_end();
        break;
}
```

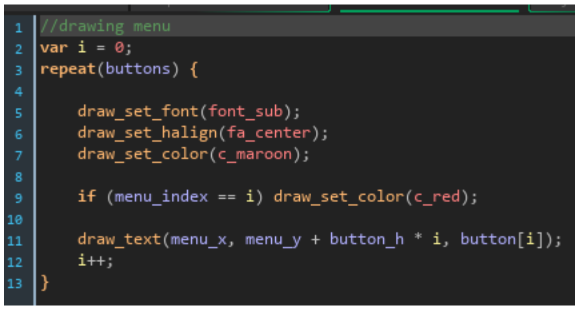
obj_fail_alien



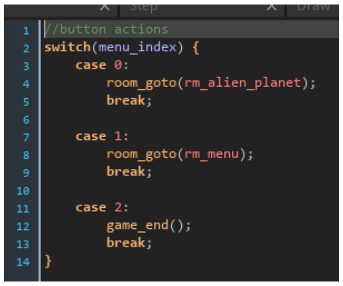
```
//menu layout
menu_x = x;
menu_y = y;
button_h = 64;

//buttons
button[0] = "Try Again?";
button[1] = "Main menu";
button[2] = "Exit";
buttons = array_length_1d(button);

//default variables
menu_index = 0;
last_selected = 0;
```



```
//menu selection
menu_move = keyboard_check_pressed(vk_down) - keyboard_check_pressed(vk_up);

menu_index += menu_move;
if (menu_index < 0) menu_index = buttons - 1;
if (menu_index > buttons -1) menu_index = 0;

last_selected = menu_index;
```



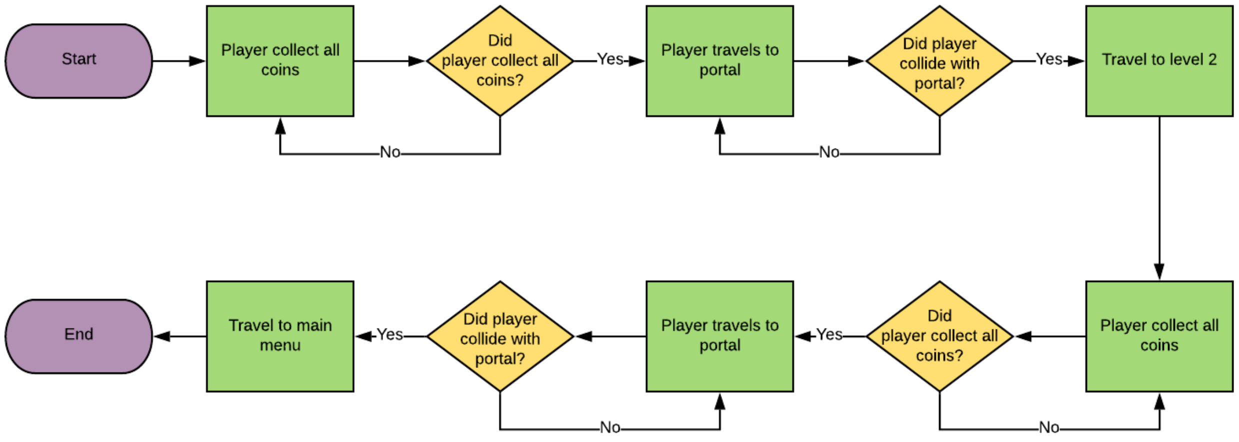
```
//drawing menu
var i = 0;
repeat(buttons) {

    draw_set_font(font_sub);
    draw_set_halign(fa_center);
    draw_set_color(c_maroon);

    if (menu_index == i) draw_set_color(c_red);

    draw_text(menu_x, menu_y + button_h * i, button[i]);
    i++;
}
```



```
//button actions
switch(menu_index) {
    case 0:
        room_goto(rm_alien_planet);
        break;

    case 1:
        room_goto(rm_menu);
        break;

    case 2:
        game_end();
        break;
}
```

Game Flow Chart

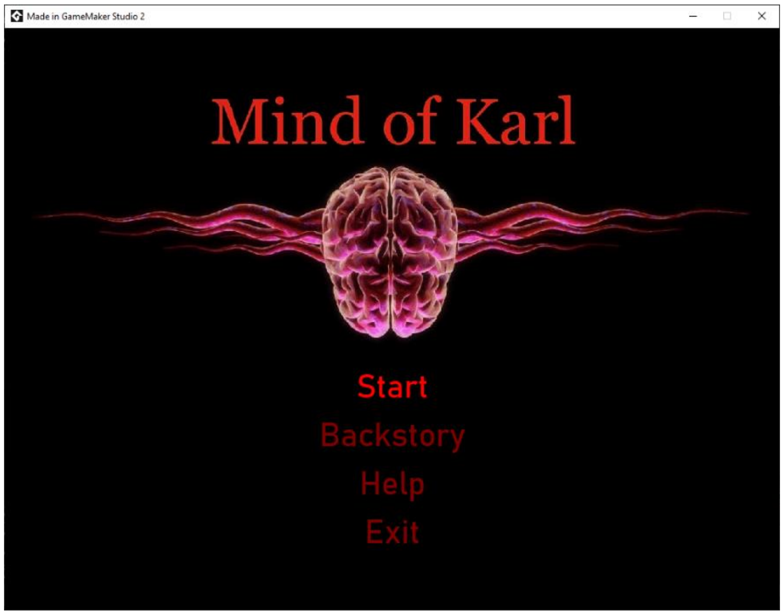
This diagram shows the flow of the Mind of Karl game:



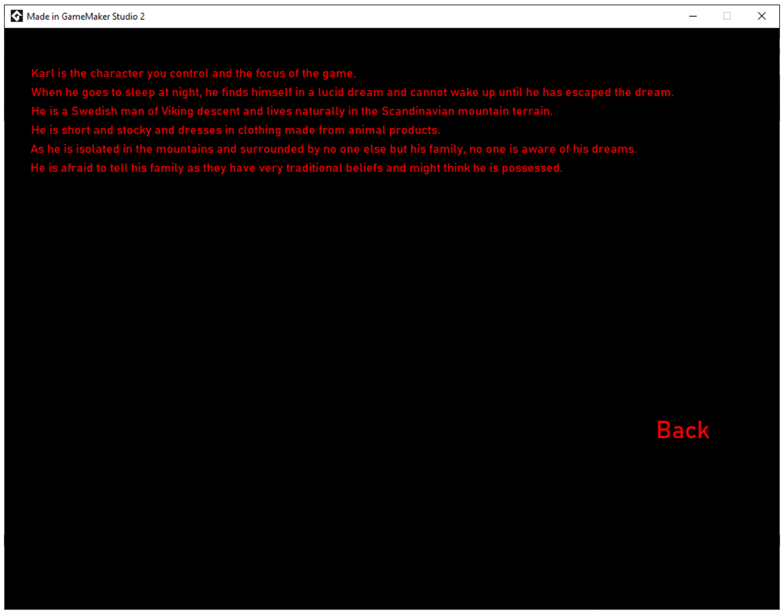
Section VIIII - Completed Game

Screen shots of the Game

This is the room you land on when you start the game. It is the main menu where you can start playing from, access other sub-menus, and exit.



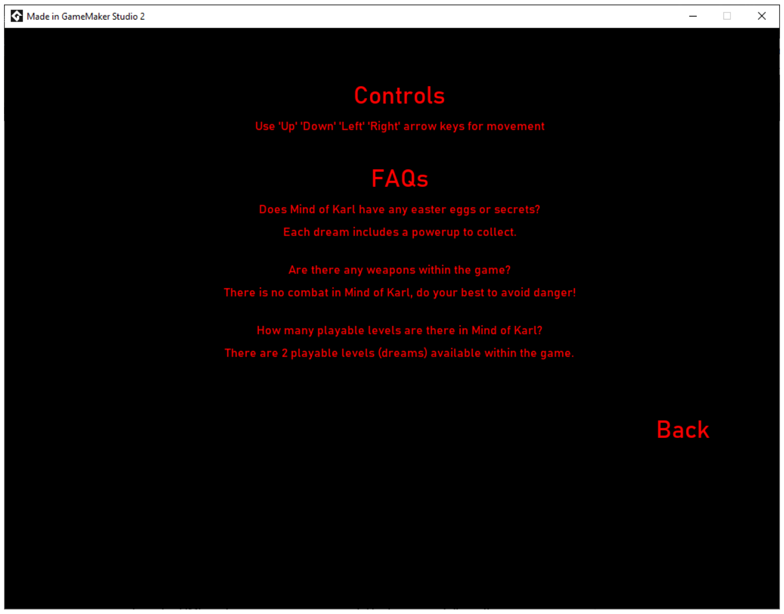
This is the Backstory room which includes a short description of Karl and why he is placed into these mazes. This helps give the player some context.


Made in GameMaker Studio 2

Karl is the character you control and the focus of the game.
When he goes to sleep at night, he finds himself in a lucid dream and cannot wake up until he has escaped the dream.
He is a Swedish man of Viking descent and lives naturally in the Scandinavian mountain terrain.
He is short and stocky and dresses in clothing made from animal products.
As he is isolated in the mountains and surrounded by no one else but his family, no one is aware of his dreams.
He is afraid to tell his family as they have very traditional beliefs and might think he is possessed.

Back

This is the Help room. It lists the basic controls and answers some frequently asked questions the players might have too.



Made in GameMaker Studio 2

Controls

Use 'Up' 'Down' 'Left' 'Right' arrow keys for movement

FAQs

Does Mind of Karl have any easter eggs or secrets?

Each dream includes a powerup to collect.

Are there any weapons within the game?

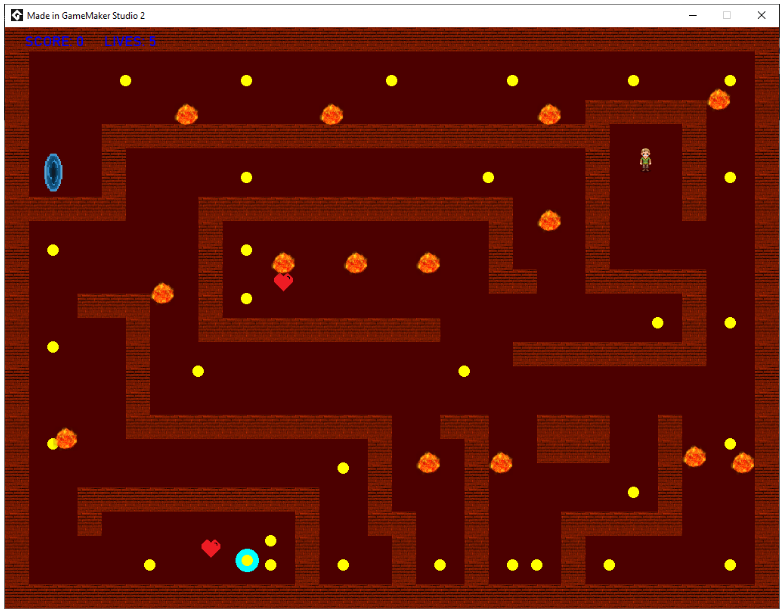
There is no combat in Mind of Karl, do your best to avoid danger!

How many playable levels are there in Mind of Karl?

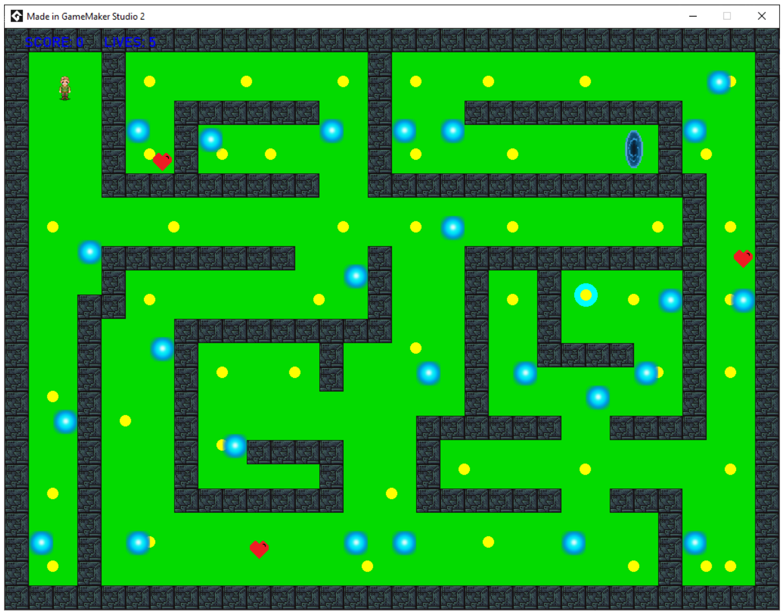
There are 2 playable levels (dreams) available within the game.

Back

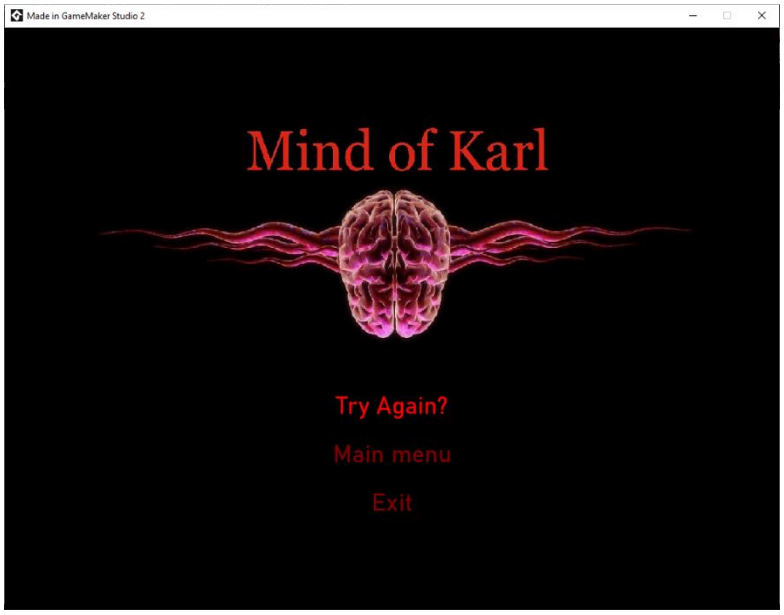
This is the first dream, the Hell room. It includes coins, hearts and a powerup to collect, as well as enemy fireflies to avoid. It has a portal which you must reach after collecting all the coins in order to go to the next level.



This is the second dream, the Alien room. It includes coins, hearts and a powerup to collect, as well as enemy alien orbs to avoid. It has a portal which you must reach after collecting all the coins in order to finish the game.



This is the fail room. This is where players are taken to after depleting all their lives. It gives them the option to try again, go to the main menu, or exit the game.



Section X - Game Maintenance

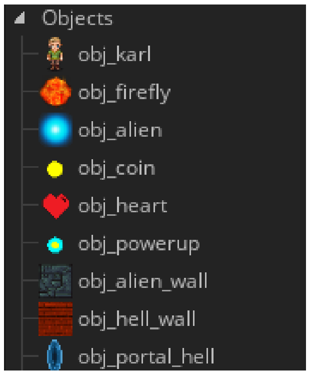
Games will often receive updates and maintenance throughout its lifetime. Aside from bugs, most games will be updated to improve balance. This could be things like modifying enemy abilities, level layouts, or weapon accuracy. These changes will often be in response to observations made while seeing the game played at scale. Developers will play and test their games thoroughly throughout development, but many changes only become apparent after the game is exposed to real players. A lot of developers will face criticism and players will demand changes. For example, in Overwatch (a popular arcade-style game), players will often influence change by criticising a character's abilities. Therefore, developers will use techniques that make their games easy to maintain/update.

I have used various techniques to ensure ease of maintenance in Mind of Karl should any problems or opportunities arise.

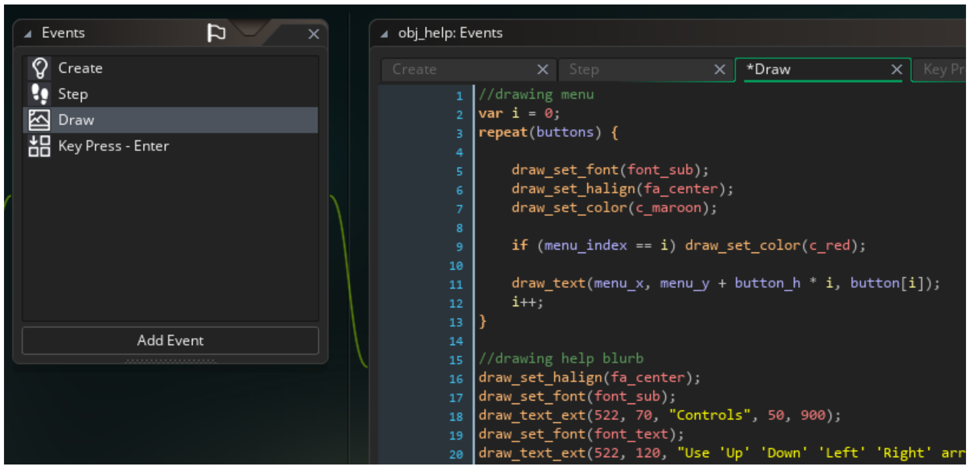
To maintain consistency throughout my sprites, objects and rooms and make maintenance much easier if required in the future, I used the snake-casing naming convention for all their names. This, along with camel-casing are two commonly used naming conventions for GML and programming in general. Snake-casing follows the format of word, then underscore, then word etc (all in lower case). This forms a snake-like format (this is where it gets its name from, of course).

Not only does maintaining a consistent naming convention mean that updating and adding assets names will be easy, but referring to them well also be easier as you know what format their name is in. If you start mixing different naming conventions, then you will often forget asset names. For example, you might think that your sprite is called 'SprEnemy' not 'spr_enemy'. Mixing up asset names is resource-consuming and tedious when trying to update/maintain a game.

You can see here that I used a consistent naming technique:


Objects
obj_karl
obj_firefly
obj_alien
obj_coin
obj_heart
obj_powerup
obj_alien_wall
obj_hell_wall
obj_portal_hell

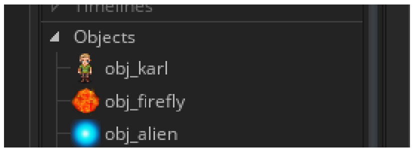
With all my object code, I made sure that it was all modular in that code was split up (by line spaces) dependent on its relevance. So, one block of code will be joined together that is associated with drawing a menu, while another block of code will surround drawing a blurb. GameMaker Studio 2 is useful for maintaining a modular approach as it separates events into different types (e.g. collision, step etc). This means you are not left with one long page of code with different actions; rather, lots of different events that are easily accessible and contain a reasonable amount of code. Moreover, I made use of the comment system to outline the blocks of code within each event. This means that you can easily access each event and distinguish the individual blocks of code easily by their relevance and by the comment at the head of each block. You can see what this looks like here:



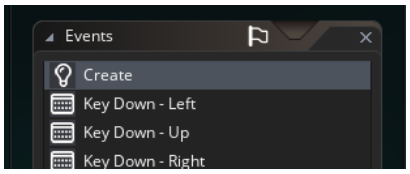
```
//drawing menu
var i = 0;
repeat(buttons) {

    draw_set_font(font_sub);
    draw_set_halign(fa_center);
    draw_set_color(c_maroon);

    if (menu_index == i) draw_set_color(c_red);

    draw_text(menu_x, menu_y + button_h * i, button[i]);
    i++;
}

//drawing help blurb
draw_set_halign(fa_center);
draw_set_font(font_sub);
draw_text_ext(522, 70, "Controls", 50, 900);
draw_set_font(font_text);
draw_text_ext(522, 120, "Use 'Up' 'Down' 'Left' 'Right' arr
```

Once you have located a block of code, custom variables will be relevantly named, appropriate indentation is applied, and line spacing is used to separate lines with slight changes in relevancy. The code is easy to read and follows a neat, industry-standard approach. Therefore, if I wanted to modify Karl's animation speed if he appears to move too slow, for example, here is how I would locate the appropriate code:

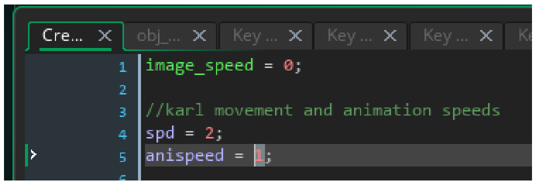
1. Open Karl's object by looking for the relevant object name.



2. Open the create event (where variables are stored, and general code is found – therefore I have made it the first event available).



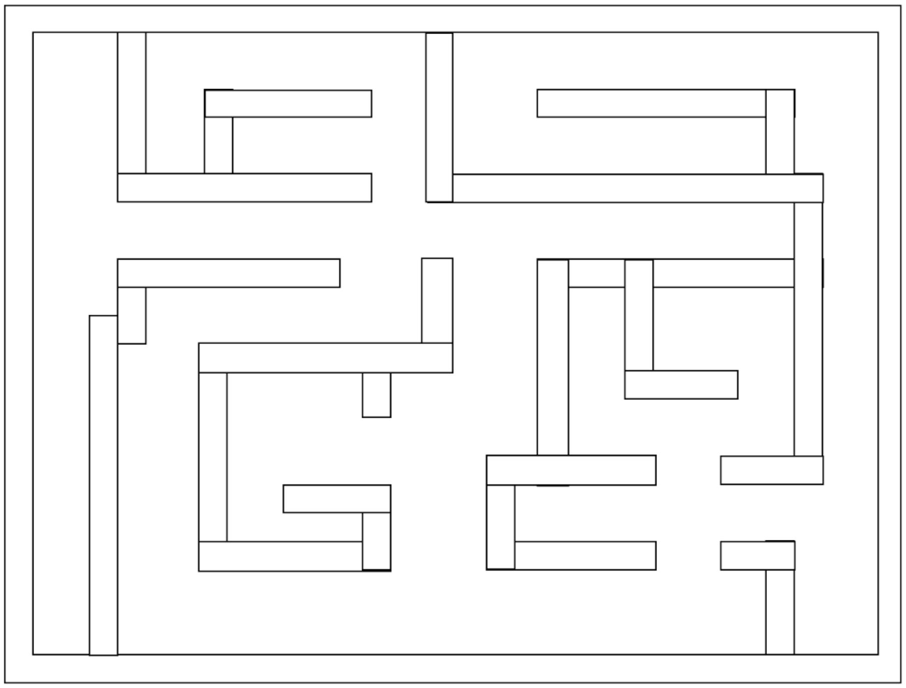
3. You can easily confirm that this is where Karl's movement code is found by the comment. Now you can just modify the animation speed variable (anispeed) as required.



```
image_speed = 0;

//karl movement and animation speeds
spd = 2;
anispeed = 1;
```

Appendix A – Game Level Designs

Level 1 (Alien Planet)



Level 2 (Hell)