2016

Andrew Davis

Movies and Shakers

Design Document

GAME REPORT – Andrew Davis

Contents

[1 Basic Concept 2](#_Toc445480907)

[2 Design Details 3](#_Toc445480908)

[3 Software and Hardware 4](#_Toc445480909)

[4 Graphics and Audio 6](#_Toc445480910)

[5 Code Structure 7](#_Toc445480911)

[6 Game Specific Features 8](#_Toc445480912)

[7 Time Management Plan 9](#_Toc445480913)

[8 Bibliography 10](#_Toc445480914)

**APPENDIX A - GRAPHICS:**

[1 Staff 11](#_Toc445480915)

[2 Customers 12](#_Toc445480916)

[3 Screens 12](#_Toc445480917)

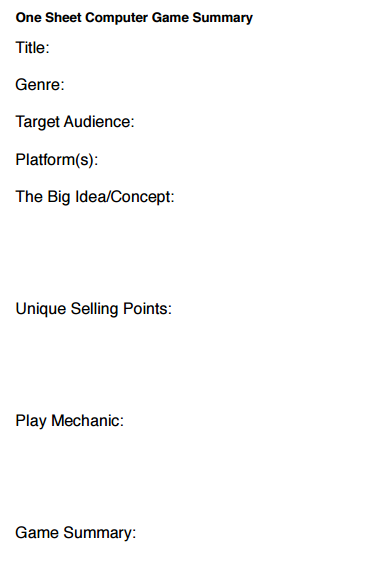
[4 Other Purchasable Items 13](#_Toc445480918)

[5 Buttons 13](#_Toc445480919)

[6 Currencies 13](#_Toc445480920)

# Basic Concept

My game is………



* TITLE: **“Movies and Shakers”**
* GENRE: Management and Simulation
* PLATFORM: IOS – iPhone, iPad, iPod Touch
* TARGET AUDIENCE: Any age – usually someone with a small amount of time to kill
* CONCEPT: Describe Game! \*see below\*
* UNIQUE SELLING POINTS: insight into how businesses grow? Customisability.
* PLOT: none
* SIMILAR GAMES: FarmVille, CityVille, The Simpsons – Tapped Out, Family Guy – TQFS

CONCEPT:

Develop as successful a cinema as possible.

Build and upgrade screens to entice more customers.

Move staff members around to speed up productivity / progress

Make your cinema as you want - customise

1 page

# Design Details

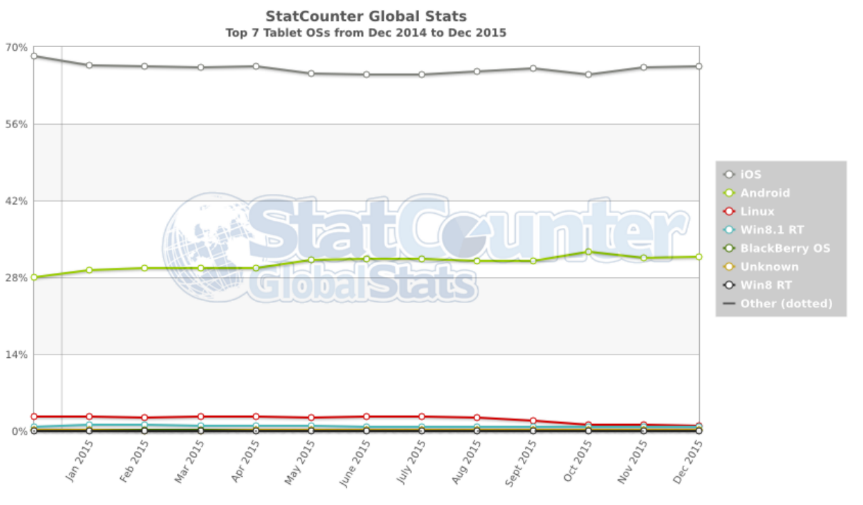
Not sure what goes here… Maybe further info on what the game is? How it will work – features

2 pages

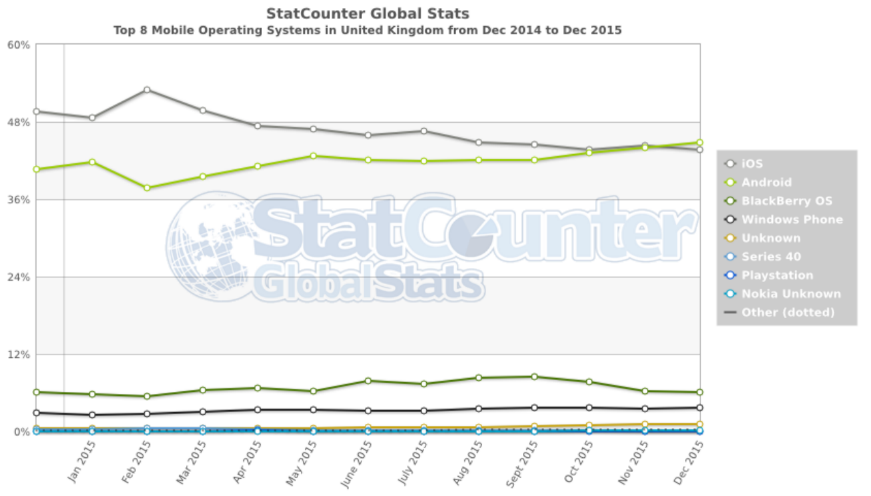
# Software and Hardware

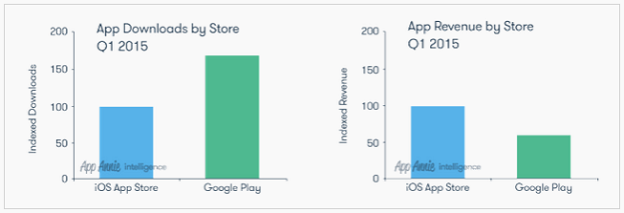
I decided to develop a Mobile game because it was an area which interested me. I am not a massive gamer myself but do play console and mobile games occasionally. However, when it came to deciding a platform, developing a console game did not appeal to me so I decided to go for a mobile game. The type of game I had planned to make (Management Simulation) is also heavily suited to a mobile device.

I was then faced with a choice which I’m sure many people have faced before: which mobile operating system to develop for? I decided to develop a game for iOS because it is one of the most popular platforms on the market. A study by *StatCounter* showed that iOS was (throughout the year of 2015) by far the most popular Operating System for Tablet devices, with over 65% of Tablet devices using it.



The graph shows that iOS is comfortably the most popular iOS for tablet worldwide and that the nearest rival (Android) is comfortably behind, with only around 30-35% of tablet devices using Android. And there is no sign that iOS will be replaced at the top any time soon.

In terms of mobile phone operating systems in the United Kingdom, there was somewhat of a shift in power in 2015. The same study from *StatCounter* found that at the beginning of the year, iOS was the most popular device. However, by the end of the year, Android had edged ahead in the rankings. However, I decided to target iOS for my game because overall (combining the figures for Tablet and Mobile), iOS is still more widely used. My game will not just be targeted towards the United Kingdom, but will be suitable for use in any country, as going to the cinema is not just a UK past time – almost every country in the world will have cinemas.

As well as the number of users that each platform had, I also did some research into how apps do on the respective markets for Android and iOS. I intend to release my game on the market after this project is complete so I was interested to see which store was more popular for downloading apps. A study from *App Annie Intelligence* found that, although more apps are downloaded Android’s Google Play store, apps on the iOS App Store tended to have a higher revenue. Since I am interested in putting my App onto the market, this is a figure which helped with my decision of choosing iOS as my platform.

Having chosen iOS, I then had to decide on what software I would use to develop my game. I decided that I would need an engine to make several aspects of the game, such as handling how the customers are created and move, a lot easier for myself, as well as making the implementation of a ‘Tile-Based’ floor easier.

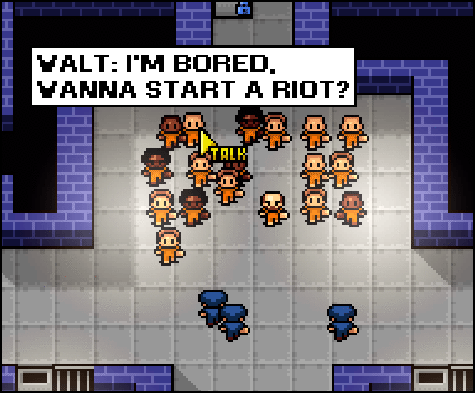
In the end, I decided to use the Unity engine.

One reason for choosing Unity is that the Scripts to control the game can be written in C# which is a language which I am comfortable in and have been successful with before. So I would not need to learn a new language and would be able to just dive straight into the coding when I was ready.

As well as this, I realised that I would have trouble with creating a game for iOS since I would primarily be developing it on my laptop which uses Windows – and Apple only allow apps for their devices to be developed on other Apple products, and also required the app to be written in a language called XCode. However, I quickly learnt that Unity has a feature which allows projects to be easily ported into XCode. I then was able to get everything that I needed installed onto one of the Macs in the Labs in the Queen Mother Building so that I could easily convert my Windows version of the game into an iOS version, and then get it on to my iPhone (which is the primary device that I will be using to test my game).

The various options for converting the Unity project into various platforms was key in my decision because if I do decide to release the app onto the market, then I should be able to expand into other platforms without too much hassle – Unity allows for porting to lots of different platforms, including Android, WebGL, and Xbox.

I also looked into other games which had been created using Unity to see what it was possible to create and how successful Unity-based games can be. I found that there were several games using Unity that had gone on to be successful – two of which stood out for me. The first was a game called “*The Universim*”. This game is a simulation game so I was interested to learn that Simulation/Management games are often made in Unity. The base functionality of *“The Universim”* is the same as what I will be implementing. By this, I mean that the player has different people (in my case they will be ‘staff’) who can be assigned to do different jobs which then impact on how successful the world (in my case, the Cinema) is.



While this was interesting to learn, I was still not convinced that Unity was the right choice as a lot of the games that I saw had very complex graphics and looked very stylish – which is not suited to my game at all! Then I came across another game, called *“The Escapsists”*. This game uses very basic, pixel-art, graphics which were much more similar to my style of graphics. It uses a top-down view similar to the graphics I have produced.

Looking into these two games have convinced me that Unity was the right choice for my game. The fact that both of these games have gone on to be successful is also very encouraging for me as I develop my own game.

The graphics style used in *“The Escapists”* is similar to the graphics style that I am using.

# Graphics and Audio

Graphics

The graphics for my game will be rather simplistic and cartoon-like. This style of graphics is often used for the Management Simulation genre of game, as they are usually just used to represent actions taking place or progress updates. In these types of games, the player does not often have direct control over the objects represented by the graphics (customers, colonists, town members etc.) – they are simply used to show the current state of the Game. As well as this, the graphics look quite fun and light-hearted – which matches the type of game I hope to create.

In my case, there will be graphics to represent the Customers of the Cinema. They will be animated to show how they progress or ‘flow’ through the cinema – from buying a ticket, to going to their screen. There will be animations for being walking in each direction (up, down, left and right), being idle, and for queuing.

Staff members will also have graphics to represent them. These graphics will be similar to those for the customers although there will be different animations. There will simply be two animations - for the staff being idle, and the other for when the player drags the staff member to the desired workstation / post. If I have time before the end of the project, I will add more animations to the staff to show them carrying out their tasks – to show the player the state of each staff member.

As well as graphics for the customers and staff, there will also be some to represent the screens and other purchasable items that the player game buy in the game. These graphics will be repositionable by the player to allow them to set up their cinema in any way that they choose. Again, these graphics will be very simple in accordance with the style of the game.

All of the graphics for this game have been, and will continue to be, created by myself. Since they are simplistic graphics, I am able to use a basic graphics editor to create them – namely, Paint.net. This software allows for basic drawing capabilities which are all that I require for my graphics.

Audio

There will be minimal audio involved in my game. There will be some background music to further convey the tone of the game. This music will be lighthearted and fun – in keeping with the style of the game. This music has been created for me by one of my fellow course mates. I have his permission to use this music in any way I like for this game.

As well as the background music, I also hope to add sound clips to each of the customers. These sound clips will be incomprehensible talking sounds that I hope will help show the success (or lack of success!) of the player’s Cinema. To explain, a busy cinema, i.e. lots of customers, will have a lot of sound coming from it as more customers means more sounds – these sounds will convey to the player that their cinema is busy and successful. On the other hand, if the cinema is less busy, there will be fewer customers and hence fewer sounds.

I hope that, by attaching sounds to the customers, I will be able to convey the success / busyness of the Cinema.

These ‘Customer Sounds’ will likely be recorded by myself.

Maybe add a page – mention the GUI – i.e. the world

# Code Structure

UML Diagrams…………………………………………………………………….

Behaviour Diagram…………..

Object Pool

I decided to use the ‘Object Pool’ design pattern in my game. The purpose of using an Object Pool is to reduce the number of resources used by the game by storing a list of all the available resources and then adding them back to the Object Pool once they are no longer in use. By reducing the number of times a new object is created (using the ‘Instantiate’ command in Unity), it frees up processor time and allows for the game to run a lot more smoothly.

I will the Object Pool to store the customer objects. I made this decision because there will be a high turn-over of customers in my game - when one customer is finished visiting, its object will be returned to the pool and can then be used by the next customer to arrive – meaning that there is no need to Instantiate a new object every time a new customer arrives. However, this raises a problem – the Object Pool needs to be set with a size of how many objects to store. However, as my game progresses and the player’s cinema becomes more successful, more and more customers will arrive every day. This will make setting the size of the queue very difficult/impossible. I hope to solve this problem by making the Pool grow-able – i.e. if there is not an object available in the pool, create another one. This means that there will always be an object available when it is needed.

This then raises another problem – if the pool grows in size during the ‘peak time’, then when the cinema becomes quiet again, we are left with too many objects in the pool – so some objects are going unused.

I hope to solve this problem by writing a piece of code that will detect if a customer object has been out of use for an extended period of time, and remove it from the pool if it has.

As well as Object Pool, I will also use the Update Design Pattern. This involves

MVC

Other 2 design or games patterns

* Where used
* How implemented
* Why chosen

Update:

* Moves the customers across the screen
  + [MOVEMENT\_SCRIPT]
  + Smooth movement across screen - DeltaTime
* Updates the time count
  + [CONTROLLER]
  + Consistent count using fixedUpdate() or using deltaTime
* Moving objects around – customizing screens
  + [MOVEMENT\_SCRIPT]
  + Checks for input – speed is not massively essential
* Moving the camera
  + [CAMERA\_MOVEMENT]
  + Gets input
* MENTION Encapsulation

Object Pool:

* Storing the GameObjects for the customers
  + [OBJECT\_POOL] / [CONTROLLER]
  + Cuts down on resources required – only add a new object if needed. Since not all customers will be on screen at same time, do not need one object per customer – can simply reuse them

4 pages

# Game Specific Features

AI – Pathfinding for customers – custom algorithm similar to A\*. Explain (Variation) – diagram (Paint)?

Concurrency – multiple customers walking on at once at once. UPDATE PATTERN

Online game play – Facebook Login???

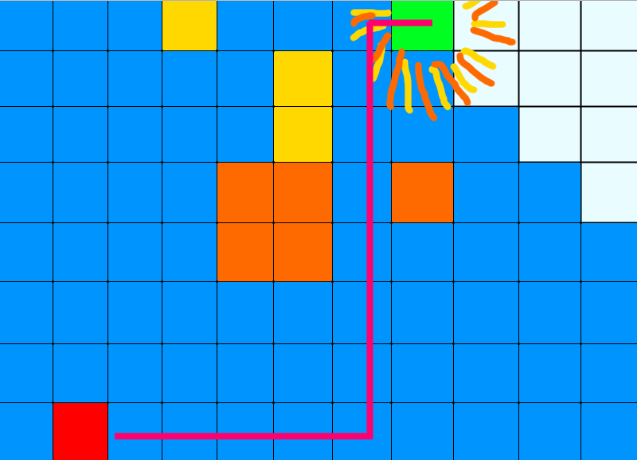
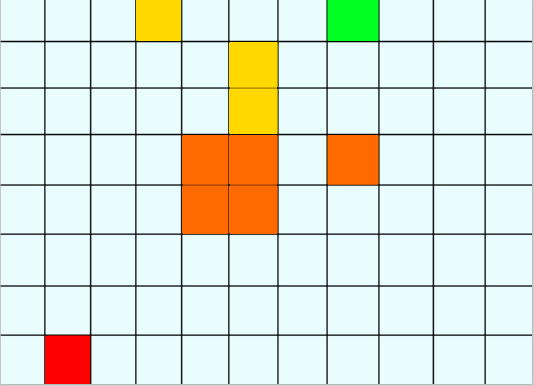
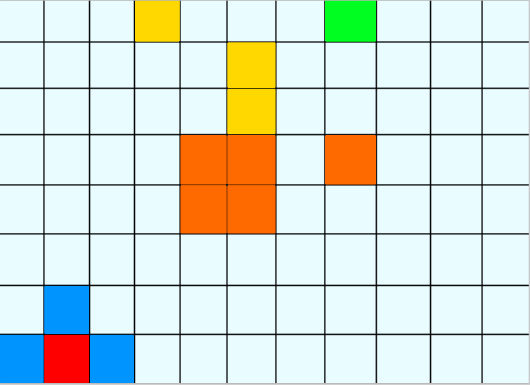
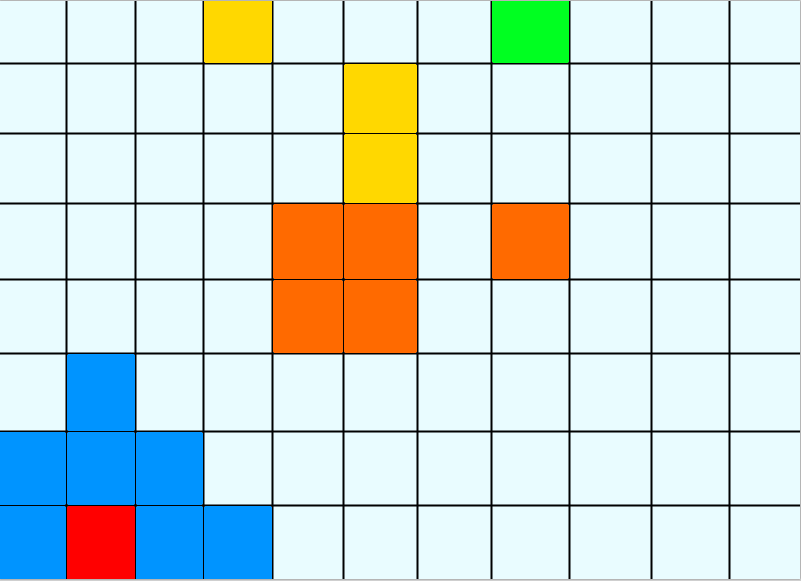
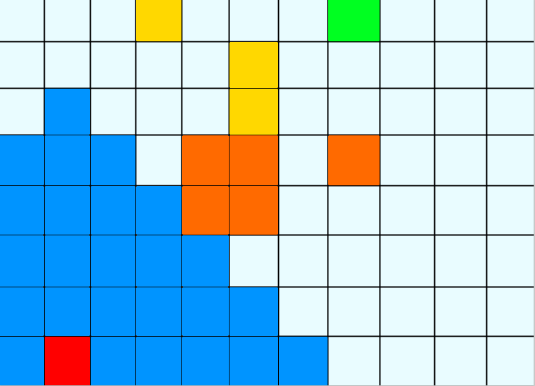
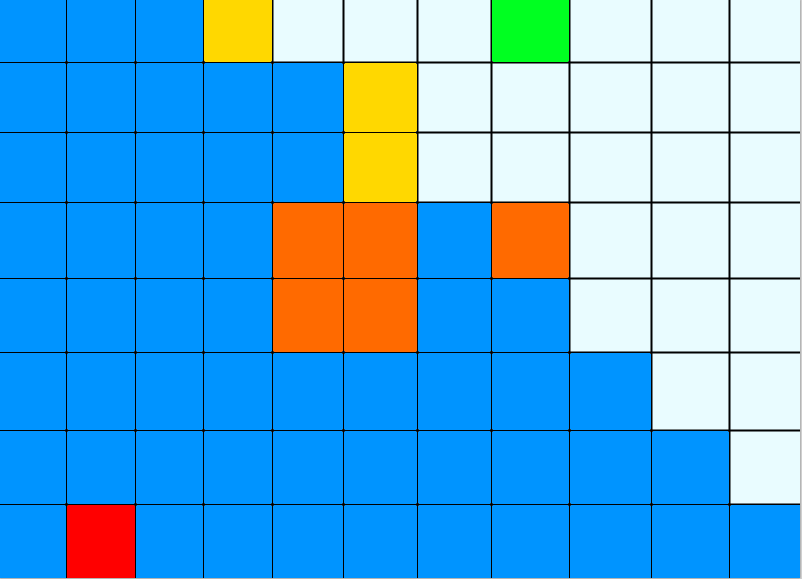
1 – 1.5 pages

Concurrency will be heavily present in my game. All of the customers who visit the cinema will have a ‘movement’ script attached to them – allowing them to all run independently of each other. [ASKED KAREN ABOUT THIS] - SHIT

My game will also have a small amount of Artificial Intelligence in it. This will be pathfinding for the customers to navigate a path through all of the obstacles or objects in the cinema to get to the various places that they need to get to (e.g. their selected Screen or the ticket booth etc.). The algorithm I will use for this is similar to the A\* algorithm but slightly different in places. In my algorithm, there is no ‘distance’ to the target stored – it simply stores whether or not a tile on the floor is in use. The algorithm will then move in all directions (up, down, left and right) until it finds the target tile.

My algorithm is not as efficient as A\* but for a limited number of tiles (i.e. the max size of my ‘World’ is 40 by 80 tiles), it is more than efficient enough to serve my needs.

There is also a problem with when to do the pathfinding. If it is done only when the path is required (for example, the customer is done with the ticket booth and now wants to find their screen), this will result in a small delay while a path is found. I plan to counteract this problem by loading all the paths that will be required at the start of a new day, then simply selecting the correct path from the list when it is needed. This is an appropriate method to use as the objects that need to be navigated to/around will be stationary and will not move throughout the day – so the paths will not need to change or be updated.



It can be seen from the diagrams alone that the algorithm I use is not the most efficient! In the diagrams, the red tile represents the start point, and the end point is the green tile. The orange / yellow tiles represent the obstacles in the way – i.e. what to avoid. The blue tiles show the list of tiles which have been explored (each tile is only explored once, so if it has been explored already, it will be ignored). From the diagrams, it is clear that a lot of tiles are explored that would not be explored (certainly not as soon) if I was using the A\* algorithm. If I have time, I may tweak my algorithm so add a distance check into it to optimise efficiency. Although, as I stated earlier, the algorithm is only run at the start of a new day – so efficiency is not a massive problem at that point.

# Time Management Plan

I do stuff when I want.

Maybe 2 pages

# Bibliography

Sources of where I got information from...

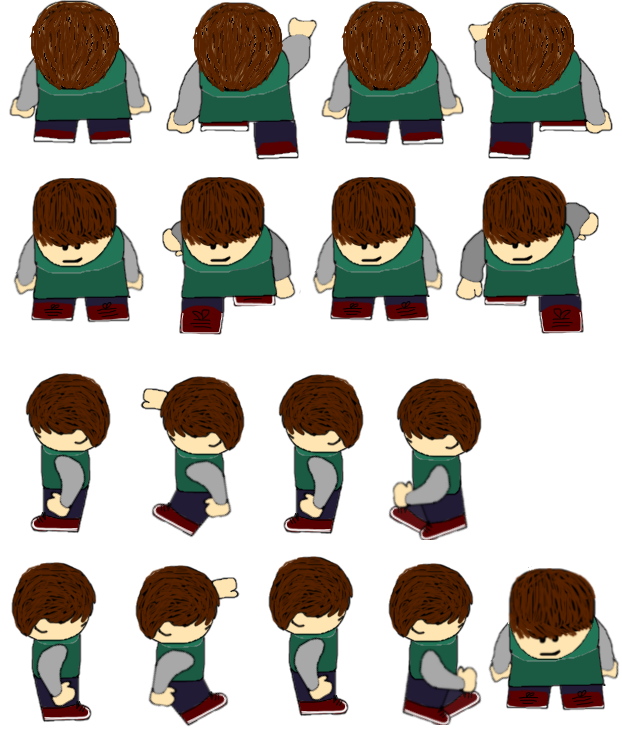
* Unity official Tutorials
* YouTube tutorials
* Design Pattern Tutorials from classmates.
* Discussion and paired learning with course mates
* OBJECT POOL: https://unity3d.com/learn/tutorials/modules/beginner/live-training-archive/object-pooling

APPENDIX A - GRAPHICS

# Staff

* There will be various staff member appearances:
* Each will have a ‘Floating’ animation Sprite sheet for when they are getting dragged by the user:

# Customers

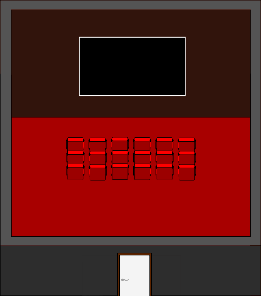
* There will be multiple customer appearances. Each will have a movement Sprite sheet:

# Screens

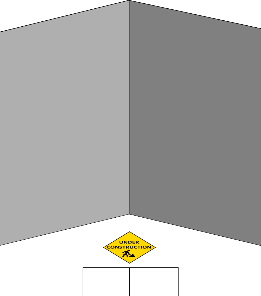
* There will be multiple images for the Screen – for the different upgrade levels:



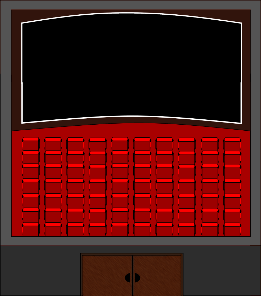
Level 2



Level 1 (Starting Level)



Upgrade in progress



Level 4 (Maximum)



Level 3

# Other Purchasable Items

* Various Different shapes and sizes:



Vending Machine



Plant



Bust/Statue of Game Creator)

# Buttons

* Basic Buttons:



Move Object



Warning



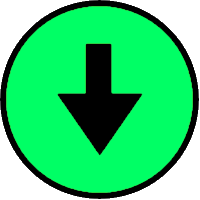
View Info



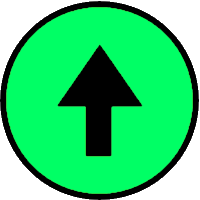
Confirm



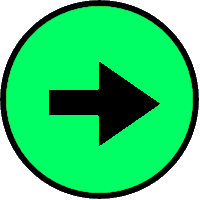
Cancel



Move Down



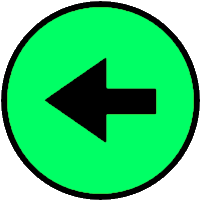
Move Up



Move Right



Upgrade



Move Left

# Currencies

* Two currencies:



Popcorn



Coins

­­­