PROJECT IN LIEU OF THESIS:

SPACE ODDITY

Written by Paul Burgermeister

University of Florida

Digital Worlds Institute

Monday, April 07, 2014
ACKNOWLEDGEMENTS

I would like to express the deepest appreciation to my committee chair, Professor Angelos Barmpoutis, who has provided me with the opportunity to conduct such an enjoyable study. I would like to thank my committee member, Marko Suvačdžić for his feedback and dedication to ensuring this project was a success. In addition, I also would like to give a special thank you to the two brilliant programmers, and friends, Ryan Roden-Corrent and Christian Brewer, who helped me make my vision come to life. Through their consistent hard work, support, and knowledge of programming, they have broadened my understanding of how truly grueling creating a video game from start to finish can be.
### TABLE OF CONTENTS

PROJECT IN LIEU OF THESIS: ................................................................. 1
SPACE ODDITY ....................................................................................... 1

**ACKNOWLEDGEMENTS** ................................................................. ERROR! BOOKMARK NOT DEFINED.

TABLE OF CONTENTS ........................................................................... 3
LIST OF FIGURES .................................................................................. 4
LIST OF ABBREVIATIONS .................................................................... 5

INTRODUCTION ....................................................................................... 7
INITIAL THOUGHTS ................................................................................ 7
RECRUITMENT .......................................................................................... 9

THE DESIGNER ..................................................................................... 10
  Genre ................................................................................................... 10
  Roguelike ............................................................................................ 10
  Arena Shooter .................................................................................... 11
  Game World ......................................................................................... 12
  Game Design ....................................................................................... 12
  Game Characters ................................................................................ 14
  User interface .................................................................................... 15
  Shop User Interface ........................................................................... 17
  Game Items ........................................................................................ 22
  Status Effects ..................................................................................... 25
  Specialization System ........................................................................ 25
  Item Upgrade System ......................................................................... 26
  Dynamic Level Content Generator .................................................. 27

THE ARTIST ........................................................................................... 30
  Art Style/Tools Used ........................................................................... 30
  Animation Solution ............................................................................ 31

THE SOUND ENGINEER ....................................................................... 32
  Musical Style ....................................................................................... 32
  Sound Engineer Tools Used .............................................................. 33

THE TESTER ........................................................................................... 36

THE PROGRAMMER ............................................................................. 39

PROJECT MANAGEMENT ..................................................................... 40
  Programs Used ................................................................................... 40

CONCLUSION ......................................................................................... 42

LIST OF REFERENCES ........................................................................... 43
# LIST OF FIGURES

<table>
<thead>
<tr>
<th>FIGURE</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>VISUAL OF THE UI WHILE IN ARENA MODE.</td>
</tr>
<tr>
<td>2</td>
<td>VISUAL OF THE UI WHILE IN FIRST PERSON MODE WHEN VISITING THE GAME SHOP.</td>
</tr>
<tr>
<td>3</td>
<td>VISUAL OF THE UI WHILE IN FIRST PERSON MODE WHEN VIEWING THE INVENTORY SCREEN.</td>
</tr>
<tr>
<td>4</td>
<td>VISUAL OF THE SCOREBOARD UI THAT IS DISPLAYED WHEN THE PLAYER SUCCESSFULLY COMPLETES A LEVEL.</td>
</tr>
<tr>
<td>5</td>
<td>VISUAL OF THE LOADING SCREEN UI THAT IS DISPLAYED WHEN THE GAME NEEDS TO LOAD CONTENT.</td>
</tr>
<tr>
<td>6</td>
<td>VISUAL OF THE ESC MENU UI THAT IS DISPLAYED WHEN THE PLAYER SUCCESSFULLY COMPLETES A LEVEL.</td>
</tr>
<tr>
<td>7</td>
<td>VISUAL OF THE DIFFERENT WEAPONS THE PLAYER HAS ACCESS TO IN THE GAME.</td>
</tr>
<tr>
<td>8</td>
<td>VISUAL OF THE DIFFERENT GADGETS THE PLAYER HAS ACCESS TO IN THE GAME.</td>
</tr>
<tr>
<td>9</td>
<td>VISUAL OF THE DIFFERENT CONSUMABLES THE PLAYER HAS ACCESS TO IN THE GAME.</td>
</tr>
<tr>
<td>10</td>
<td>THIS IS AN EXAMPLE OF HOW THE DYNAMIC LEVEL CONTENT GENERATOR SYSTEM WORKS.</td>
</tr>
<tr>
<td>11</td>
<td>IMAGE OF NANOLOOP'S UI FOR THE IPHONE.</td>
</tr>
<tr>
<td>12</td>
<td>IMAGE OF THE AS3SFXR UI.</td>
</tr>
</tbody>
</table>
# LIST OF ABBREVIATIONS

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Full Form</th>
</tr>
</thead>
<tbody>
<tr>
<td>RPG</td>
<td>Role Playing Game</td>
</tr>
<tr>
<td>FPS</td>
<td>First Person Shooter</td>
</tr>
<tr>
<td>GDC</td>
<td>Game Developers Conference</td>
</tr>
<tr>
<td>UI</td>
<td>User Interface</td>
</tr>
<tr>
<td>IDEA</td>
<td>Interactive Digital Entertainment Association</td>
</tr>
<tr>
<td>NPCs</td>
<td>Non-Playable Characters</td>
</tr>
<tr>
<td>2D</td>
<td>Two-Dimensional</td>
</tr>
<tr>
<td>3D</td>
<td>Three-Dimensional</td>
</tr>
<tr>
<td>M1</td>
<td>Left Mouse Button</td>
</tr>
<tr>
<td>M2</td>
<td>Right Mouse Button</td>
</tr>
<tr>
<td>SP</td>
<td>Specialization Points</td>
</tr>
</tbody>
</table>
Abstract of Project in Lieu of Thesis Presented to the College of Fine Arts of the University of Florida in Partial Fulfillment of the Requirements for the Degree of Master of Arts

SPACE ODDITY

By
Paul Burgermeister
April 2014

Chair: Angelos Barmoutis
Committee Members: Marko Suvajdzic
Major: Digital Arts and Sciences

The purpose of my project is to better understand the process a small indie game development company would undergo when they develop a game title from a concept to a finished shippable product. The best way I figured to understand and to learn about the process was to make a shippable game title throughout the duration of my graduate studies. With the help of two programmers, I was able to create an incredibly enjoyable game and understand the efforts required to make a video game; this was my process and what I found.
INTRODUCTION

A game can be an extremely powerful tool; it can be used to tap into many types of positive physical, mental and emotional states of engaged individuals. Achieving these positive states can be a challenge, one that tries incredibly hard to create a feeling of accomplishment and avoid causing boredom. Large companies that have a large supply of funds and a small nation of workers often masterfully achieve these attributes. Small parties rarely achieve strong attributes, but when they are able to portray these emotions on a high scale, the game content peaks my interest. With the growing desire for compact, lightweight games to be played on handheld devices, such as mobile phones, the rise of small indie game development companies comprised of very few workers positioned to create this type of content has recently seen an increase. In one survey, State of the Industry, taken at the Game Developers Conference (GDC) during March of 2013, one of the key findings showed that the number of independent game developers and small-scale game-development groups are steadily rising. The survey showed that 53% of the 2,500 contributors identified themselves as “indie developers” and of those, 51% had been an indie developer for less than two years. In addition, 46% of the respondents worked for companies that had ten or less people employed and only 24% of them worked with a publisher when releasing their last game. I wanted to experience indie game-development firsthand by creating a single-player role-playing game (RPG).

INITIAL THOUGHTS
I quickly realized that creating a game was an enormous task, so I broke the process up into a list of the necessities I needed to complete it. The list of job positions at a large scale game development company can reach into the hundreds; I did not have that luxury. I needed to design a game that would meet my desired level of enjoyment and do it with a small group of unpaid volunteers, not a very easy task. To minimize the amount of help I would need, I decided that I would fill many of the needed roles. The basic roles that go into making a game can be broken down into designer, artist, programmer, sound engineer and tester.

The designer is the person who designs the way the game plays; they conceive the entire structure of the game and the rules that define it. Often, a development team will have a lead designer who coordinates the work of other game designers. The designers are the primary visionaries of the game. One of the many roles of a designer is being a writer, who oftentimes creates the game’s overall narrative, commentary, journals, cut scene narrative, and dialogue. Writers also contribute to game mechanics, the user interface, and character development as needed.

The artist produces the art for the game. A lead artist, or art director, generally oversees the production of the games art, to ensure that the designer’s vision is followed. The artist’s job differs based on what type of visual dimension they will be working with. If the artist is a 2D artist, their tasks can include creating, concept art, sprites for animation, textures, environmental back drops or terrain images and the games user interface. 3D artist’s tasks can include creating 3D models, animations, 3D environments, lighting, rigging, and cinematics.
The programmer is a software engineer who develops the game or any software that is needed to create the game. The programmer is responsible for creating the game’s codebase. Oftentimes there are a number of programmers that are led by a lead programmer. The lead programmer helps implement the game’s initial codebase, oversees the remaining tasks to be completed and allocates projects to other programmers as needed until the game development is complete. Some of the different programming disciplines include game physics, AI, graphics, sound, gameplay, UI, network communications, scripting, game tools, and input processing.

The sound engineer is a technical professional who is responsible for all of the sound effects used within the game. Sound engineers also may be tasked with creating sound assets, overseeing voice acting, and music compositions.

The tester fills the role of quality assurance. They analyze the game and document the game’s defects so that they may be corrected by the other members of the development team. Testing can be very expensive and is primarily done near the completion of the project.

I utilized my personal skill set to fill the roles of designer, artist, sound engineer, and tester. I reached out to others to complete the programming for my game.

RECRUITMENT

When I decided to start this project, I was a member of the UF club IDEA (Interactive Digital Entertainment Association), a club focused on working as a team to create small video games. The club had many members with various backgrounds that
were all applicable to the creation process of game development. During my time with IDEA, I spoke with many UF students who were skilled in programming and eventually found two willing individuals who offered to help by taking on the task of the programmer.

THE DESIGNER

My first task as the designer was to come up with an achievable project for three people that wasn’t too overwhelming and would complement our individual skill sets.

Genre

I decided to experiment with two different game genres: a hybrid of a roguelike and an arena shooter that was displayed in a third person view from the side.

Roguelike

Roguelike is a sub-genre of role playing games (RPG), that are generally characterized by random level generation and punishing game mechanics such as permanent death. Roguelikes descend from the game Rogue, created in 1980, that mimic the games, turn based gameplay that provides time for the player to plan their move, extreme fantasy settings, mechanical difficulty and sprite-based graphics. Many games that include all of these are oftentimes considered to be classical or Berlin versions of the genre. In more recent years, with more powerful computers and gaming systems, the genre has shifted away from tile-based movement and turn-based gameplay to instead have their game play supplemented with other genres such as
platformers and action games. There have been some recent very popular and famous games that use the modern rendition of roguelike such as the Diablo series and the Torchlight series. Both series have had a number of published sequels and copycat games.

**Arena Shooter**

The arena shooter is a sub-genre of the first person shooter (FPS) genre. The description for the arena shooter genre is hazy at best, but some of the more common trends found in the genre are a fixed first person view and some type of combat between the player and either other players or non-playable characters that is confined to a small enclosed area that the player cannot leave. These games are also often times based on a point system to tally scores, rounds of game play that are ended after a specific duration or a reached goal, and some limited amount of character customization in way of the players weaponry and abilities that they can use.

**Hybrid Design**

I decided to create a role playing game, set in a high fantasy setting, where each level played by the player is within an enclosed arena, viewed in third person at a profile angle. Within each level the player would fight waves of non-playable characters (NPCs) with a large variety of weapons they would choose from before entering each level. The way in which the player dispatched their enemies would reward varying points based on a dynamic point system that rewards more points for creativity, each round of gameplay ends when a specific amount of enemies have been destroyed.
Game World

For the game world, I wanted to create an environment that would allow for humorous game play and exciting encounters that were only enhanced by its goofy themes. The game story involves a rarely traversed sector of space, known as HB-12, that has recently shown some unusual gravitational activity. Void zones randomly appear throughout the sector and must be filled by the space traveler. The protagonist controlled by the player is an adventurous Space Meteorologist who sets out to explore the region of space to discover what is causing the strange anomalies. Upon arrival to HB-12, the player discovers that the region is not as barren as initially thought. HB-12 has become home and battlefield to many strange and foreign creatures. The new HB-12 inhabitants have mysteriously appeared in the region and they don’t seem to be from the Space Meteorologist plane of existence. The new “neighbors” are from an astral plane called The Fae Wild, a harsh and unforgiving environment. The Fae Wild is home to many exotic races and species such as the Elves, Dwarves, Goblins, Orcs, Ogres and many more. Their world is in constant chaos caused by many of the more prominent factions warring against each other over trivial reasons. The Space Meteorologist finds himself stuck in the middle of this incredible event and is determined to figure out why The Fae Wild is spilling into his realm and how to stop it.

Game Design

World Layout

The game is broken up into three different modes: the arena, first person and the Overworld.
**Arena**

The Arena is the portion of the game where the player fights the antagonists. The player is forced to survive against waves of enemies, neutral characters who may or may not be of help to the player, void zones that are constantly causing changes to the physics inside the arena, and random space junk. The arena is a closed environment (where) the player loads into upon starting a level and may not leave it until the void zone is filled. The void zone is filled by incapacitating enemies which causes them to float helplessly into the void zone, increasing its capacity. Once the void zone is filled, the level is over and the player is rewarded with in-game currency and a score for the level.

**First Person**

The First Person portion is a blanketing term for whenever the player is at a screen that is made to look like you are viewing the world from their eyes. For example, when a player goes to view their inventory or look at what's in stock at the nearby vendor, they are in First Person. There is no movement involved, only game options, such as equipping an item or buying a gadget.

**Overworld**

The Overworld is a large 2D environment in which the player may explore. The Overworld initially is covered by a layer of black fog that conceals areas of the Overworld. These areas of black fog are not uncovered until the player travels to them;
this is oftentimes referred to as fog of war. Uncovering the zone allows the player to see new locations of levels, cities, shop and quest giving non-playable characters (NPCs). The player may traverse the Overworld by following predetermined paths that lead from one area of interest to the next. These paths branch off in many directions giving the player options on where to travel.

**Game Characters**

There is only one playable character, the Space Meteorologist that is exploring and studying the region the game is set in. There are neutral and hostile non-playable characters. The majority of them are made of up the creatures from the Fae Wild, the plane of existence where they come from.

**Creating a Character**

When you start a new game, you may name your character and adjust the color of their space suit. Additional character customization comes from the different items the player has to pick from when deciding what to carrying into the next level. The player has a large arsenal of items to choose from when preparing for each level. The player is only allowed to carry a specific amount of items with them into a level, also known as the arena. Because of this limitation, the player is forced to pick and choose which items they would prefer to wield while in combat during the following encounter. These items may be bought with in-game currency or earned as rewards for completing missions and quests. Each item has a special property that helps the player in combat.
The items are separated into three different categories: Weapon, Gadget and Consumable. Character customization also comes in the form of a specialization system. The specialization system allows the player to spend their collected points earned after each arena battle. These points can be used to gain additional modifiers to their character. Within the specialization system, there is a myriad of different talents to choose from, all with different modifiers the player can benefit from. The player, allowing for further customization of their character, decides upon their specializations.

**Non-Playable Characters (NPCs)**

Non-playable characters are the enemies, monsters, and creatures found in the game. Every character in the game, aside from the character the player controls, is considered a non-playable character. The non-playable characters are from a different plane of existence, known as the Fae Wild. Both friendly and evil characters, collectively known as Fae creatures, inhabit the Fae Wild. The Fae creatures are mysteriously appearing in the players’ universe and causing all sorts of chaos in the region they’re appearing. They are separated into multiple, currently warring factions, the elves, goblins, orcs, dwarves and a few other smaller groups. Each faction has multiple different units all with their own unique abilities, attributes and art.

**User interface**

The overall theme of the user interface is to look like a hodgepodge of scrap space parts and holographic displays. The user interface changes depending on the game setting the player is in.
**Arena User Interface**

The user interface (UI) for the arena is intended to be clean and minimal. It is designed this way to maximize the view the player has of the current level to helping them to see content as soon as it appears, allowing them more time to react. Many of the user interface assets are designed to have some level of transparency to them to enhance the players overall awareness of the level. Some user interface options will remain hidden until the player needs access to that portion of the interface, for example, the consumable item buttons.

![Visual of the UI while in arena mode.](image)

**Overworld User Interface**
The user interface for the Overworld is also straightforward and simplistic. The reason for this design is to provide a smooth transition for the player when navigating from location to location throughout the Overworld.

**Shop User Interface**

The design for shop user interface is to be comical and engaging. The shop is set within an old dingy food truck that has been repurposed to sell items that the player may need. There is an area in the shop user interface that displays the objects for sale. When an item is selected, a portion of the user interface displays the specific attributes that belong to that item. A player can browse the shops inventory, which fluctuates after a specific duration of time based on an in-game timer.

Figure 2 Visual of the UI while in first person mode when visiting the game shop.
Inventory User Interface

The user interface for the inventory screen displays the player’s current character in the center, along with different menus that are brought up by selecting tabs that are labeled and correspond with the different classes of items the character can equip: Weapons, Gadgets and Consumables. The player may browse through a list of items they have obtained and inspect the item and its attributes by clicking on the specific item. The player may drag the icon of an item they want to use from their list of items onto a portion of the user interface that depicts the items the character is currently using, called character slots. The character slots can only hold two weapons, two gadgets and six consumables at a time. The items the player has placed into their character slots when entering an arena are the only items they have access to for the duration of the arena. If the player wishes to make adjustments to the items they have in their character slots, they must do it to the inventory user interface which is only accessible through the Overworld screen.
Scoreboard User Interface

The user interface for the Scoreboard appears at the end of a completed arena. It displays a large amount of statistics based on the previous arena completed. The Scoreboard always displays the number of enemies the player destroyed, the amount of coins the player collected during the arena, the new current total amount of coins the player has, and the total score the player earned. There is a portion of the Scoreboard that displays Fun Random Stats, the subject of the stat could be something like the number of shots fired that didn’t hit anything and the number of enemies frozen. The subject is randomly pulled from a large table of options and only six of these options are displayed on the Scoreboard at a time. The particular statistics for the subjects aren’t random; they are continually tracked and recorded by the game and then displayed if
the subject of the stat is randomly chosen to be displayed on the Scoreboard once the arena is completed.

Figure 4 Visual of the Scoreboard UI that is displayed when the player successfully completes a level.

**Load Screen**

The user interface for the load screen is an image that overlays the previous background and displays a load bar with a helpful tip for gameplay and a humorous reason for why the loading bar is replenishing.
Figure 5 Visual of the Loading Screen UI that is displayed when the game needs to load content.

ESC Menu

The ESC menu is an image that overlays the previous background and displays a small menu with options for the player to choose from. This menu is accessible at any point in the game by pressing the ESC key on the keyboard. While this menu is active, the game progress is halted until the escape menu is turned off. To turn off the escape menu, the player only has to press the ESC key while the ESC menu is active or press the resume button on the ESC menu.
Figure 6 Visual of the ESC Menu UI that is displayed when the player successfully completes a level.

**Game Items**

Within the game, the player will collect and use a number of different items to help advance their progress in the game. Each item is broken up into separate categories: weapons, gadgets and consumables.

**Weapons**

Inside the arena, the player may use the weapons they have equipped in their character slots. The character may wield two weapons at once, one in each arm. The player may choose to carry two of the same weapons. To shoot a held weapon while the player is in the arena, the player will use the left mouse button (M1) and the right mouse button (M2) to use the corresponding weapon, which is determined by the
placement of the weapon in the character slots on the inventory user interface screen. Only one weapon may be used at a time. The aiming of the weapon is based on the coordinates of the player’s mouse cursor and where the player controlled character is while they are using the weapon. Each weapon has a separate set of attributes: Accuracy, Damage, Range, Rate of Fire and Special Abilities. All weapons can be found at the shop, however, not all of them may be in stock while the player is there.

![Visual of weapons](image)

Figure 7 Visual of the different weapons the player has access to in the game.

**Gadgets**

While in the arena the player may use the gadgets they have equipped in their character slots. To use a gadget while the player is in the arena, the player will use the left shift button (L Shift) and the space bar (Space Bar) to use the corresponding gadgets, which is determined by the placement of the gadget in the character slots on the inventory user interface screen. Only one gadget may be used at a time. The aiming of the gadget, if needed, is based on the coordinates of the player’s mouse cursor and where the player controlled character is while they are using the gadget. Each gadget
has an additional attribute, this attribute is called Energy Consumption. Each gadget has a pool of energy assigned to it which is represented on the arena user interface by a bright blue bar. Each pool of energy may store up to 100 points of energy. On use of a gadget, the pool of energy that is linked with the gadget is reduced by a specific cost, unique to the gadget. The amount that the pool of energy is reduced by for each use of the gadget is based on the Energy Consumption attribute. A gadget may be used until using the gadget would reduce the energy pool to less than 0. The energy in the energy pools replenishes over time, at a rate of 2 energy points per second. All gadgets can be found at the shop, however not all of them may be in stock while the player is there.

Figure 8 Visual of the different gadgets the player has access to in the game.

**Consumables**
In the game, the player has access to items that they may use that have a one-time use ability on them that causes them to vanish after each use.

Figure 9 Visual of the different consumables the player has access to in the game.

**Status Effects**

Units can be affected by different status effects, these status effects aren’t just binary but some of them have varying degrees of severity represented on a scale of 0 to 100. The higher the severity the more pronounced the effects are. The severity of the effect decreases over time, based on the unit’s resistance to the specific effect.

**Specialization System**

The Specialization System is another way to personalize your character. The reason for the Specialization system is so that every player may play their character
slightly different from another person and to build their character to fit their specific play style. At the end of every completed Arena, the player is rewarded a certain amount of Specialization Points (SP) after the scoreboard is done calculating their score for the level. The SP rewarded correlates directly with the score earned for the level. On the Specialization System user interface, the player may spend their collected SP to unlock new abilities and attributes called Specialties, for their character. Some Specialties are available to purchase with SP immediately, and some are grayed out. This is because the player must first unlock a specific Specialty that is linked to the desired Specialty before being able to unlock it. When a player dies and is forced to start the game over, any previous purchased Specialties remain unlocked, but any unspent SP is lost.

**Item Upgrade System**

At any point in time, a player has the option to modify the weapons and gadgets they are carrying. When a weapon or gadget is dropped in the arena or generated randomly at the in-game store to be sold, the weapon or gadget may come with 0 to 2 sockets on it. The sockets are used for modifying the properties, attributes and functions of the item. The specific modification that is placed into the weapons and gadgets to modify them is called a Mod. Mods come in various types. Each type of Mod affects each item differently. A specific user interface is dedicated to modifying items and is the only place where item modifying may be done. This user interface is accessible through the inventory screen. Once a Mod has been into an item, it cannot be removed. A Mod may be placed in a socket that has a preexisting Mod in it, the new Mod replaces the old Mod and the old Mod is lost forever. Only one of each type may be placed into an
item. There are 8 types of Mods, each represented by a different color, red, orange, yellow, green, blue, purple, black and white.

**Dynamic Level Content Generator**

Each level in the game has content for it generated by a system designed to provide a new and exciting experience that allows the designer to introduce events to the player at any specific time of their choosing. The system is called a Dynamic Level Content Generator. Each level has a number of waves, each wave brings new enemies to the arena and each wave appears in the game at a specific defined interval. Each wave has two different attributes that it references when spawning its content, threat level and static enemies. Each enemy in the game has a certain threat level assigned to it based off of its intended difficulty. When creating the content of the waves for the level, the designer may assign a number to the wave’s threat level. A wave’s threat level is a number that represents the amount of points total that may be randomly spent on enemies chosen from a preset table of possible enemies it may spawn. This preset table that may be chosen from can be adjusted per wave to have different content each time. The preset table allows the designer to define the different types of enemies they want to have the computer spawn during a wave but maintain the difficulty of the wave each play through of the game. In addition to a defined threat level number the designer may also set a static number and type of enemy to be spawned at a specific wave every play through in case they want to introduce something to the player at a very specific moment in the game. An example of this system would be:
Wave 5 is given the allowance of 10 points to spend on enemies to spawn and 1 static enemy. The table of enemies that the 10 points may be spent on is set to only have the Goblin Melee, Goblin Elite and Goblin Ogre. The Goblin Melee costs 1 point, the Goblin Elite costs 5 points and the Ogre costs 8 points. The static enemy is set to be an Ogre. When the wave spawns it randomizes the points spent on enemies that will be spawned, the points spent must equal the total allowance set, which is 10. In addition to randomized enemies that will spawn, an Ogre will spawn as well, because it was designated as a static spawn for this wave. This gives the outcome of the wave 4 different possibilities. These 4 different possibilities can be seen in the example image below.
Dynamic Level Content Generator
Wave Content Attributes

Example Wave Setup

**Threat Level**

<table>
<thead>
<tr>
<th>Enemy Values:</th>
<th>1</th>
<th>5</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>![Goblin]</td>
<td>![Goblin]</td>
<td>![Ogre]</td>
</tr>
</tbody>
</table>

**Static**

Possible wave outcome with an allowance of 10

1#

- 10 Goblin Melee
- 1 Ogre

2#

- 5 Goblin Melee
- 1 Goblin Elite
- 1 Ogre

3#

- 2 Goblin Melee
- 2 Ogre

4#

- 2 Goblin Elite
- 1 Ogre

---

Figure 10 This is an example of how the dynamic level content generator system works
The randomness that this system provides continues to hold true to the roguelike
genre and to keep each play through of the game new and fresh.

THE ARTIST

Art Style

When choosing the direction of the art style for the game, I had to take into
account the abilities of the artist, myself. I being the artist for the project made the
decision very easy. The majority of my artistic abilities are in 2D, so that is what the
game is in. I really enjoy the 2D art work done by Dan Paladin for the games Alien
Hominid, Castle Crashers and Battle Block Theater so that is where I drew my
inspiration from for the style of the game. The art work created for these games are very
iconic and recognizable, they focus on a cartoony theme. This seemed like a great fit for
my specific skillset, it would mesh well with the fantasy setting and the humor that I
wanted to be displayed in the game. I would use thick outlines for the images created to
give an illustrated feeling along with sold shapes of color for shading and showing
depth. The use of bold colors and avoiding using predominantly dark colors would help
the objects stand out from the dark background the player will oftentimes be playing in.

Artist Tools Used
Due to the enormous amount of work needed to create this game, I wanted to use a program that I was very skilled with, something that I could quickly create visuals and something that I didn’t have to take the time to learn.

**Illustrator**

I decided to create all of my art assets in a program by Adobe Systems called Adobe Illustrator. Adobe Illustrator is a vector based graphics editor program which allows me to scale any image I make within the program to any size and avoid any distortion that may happen when using a program that is pixel based. This will help cut down production time spent on any image size editing if testing shows that different scale images would be optimal.

**Spriter**

Spriter is program that enables the user to take a “modular” approach to animating where, instead of each frame being a single complete image, it is constructed from many small, re-useable images (such as body parts). Each of these images that are used to construct a full frame can be scaled and rotated to further increase the use an artist gains from the assets created. This modular method of animating offers many benefits for several aspects of game development, such as time conservation, iteration edits, saving file space, and it helps tremendously for character variations.

**Animation Solution**

I designed a system for attaching limbs to specified anchor points on sprites that can rotate within certain bounds based on the velocity of the sprite that gives the
appearance of animation. This system was implemented to save time when creating art assets. Traditionally artists would have to create separate images for every motion a sprite made. With this system it allows us to make one sprite and still have the illusion of animation. The specific limbs on a character that hold weapons rotate towards the direction that the character is aiming and have anchor points for the weapon located at the hand. The hand that the weapon will be joined to is connected to the limb of the character through anchor points, this is to make it look like the weapon is being pointed in the direction that the character is aiming.

THE SOUND ENGINEER

Musical Style

I had no previous musical experience, so I needed to come up with a style of music that would be easy for me to emulate and wouldn’t require an extreme amount of work. I chose to use the musical genre Chiptune after seeing the success of the Bit.Trip game series that features an amazing mix of 80s aesthetics and modern game design. The game series is primarily known for its great chiptune inspired soundtrack.

Chiptune

Chiptune, also known as 8-bit music, is synthesized electronic music that is created or emulated through the use of sound chips of vintage computers, arcade machines and video game consoles. The game technologies initially used in chiptune music were created for consumers in the 1980s and 90s. Some of the more popular gaming consoles that used such technology were the Nintendo Entertainment System,
Sega Genesis, and Game Boy. After many years of only being seen in retro gaming consoles, the genre is making a comeback with the release of many different chiptune music creators including LittleSoundDJ for the Game Boy and Milky Tracker for Windows.

**Sound Engineer Tools Used**

To create the different types of sounds I would need for the game, I settled on two different programs—one for soundtrack creation and another for sound effect creation.

**Soundtrack**

For soundtrack creation I chose the program Nanoloop. Nanoloop 1.6 is a sequencer for the Game Boy. Sound that is generated by Nanoloop 1.6 uses the Game Boy’s analog sound chip which is restricted to raw rectangular waves, noise and a 4-bit wave form. Nanoloop 2.5 is a version for the Game Boy Advance, which can produce filtered waves, filtered noise and simple FM. The iPhone version of Nanoloop, the version I used for this project shares the same sound engine, file format and a similar interface that Nanoloop 2.5 uses. It combines an enhanced version of Nanoloop 2.5’s synthesizer with a simple sampler. It also allows the user to record audio onto their mobile device and then may import it to a PC.

Nanoloop uses a sequencer that displays its pattern as a grid of 4 x 4 rectangles. Notes can be set, removed, and edited just by tapping and swiping. This editing without mode allows for changes on the fly coupled with its gorgeous layout provide a great
environment for a smooth workflow. Unlike the Game Boy versions, Nanoloop for
iPhone only allows the user to set pitch and one additional value step in the sequencer.
All other sound parameters use controlled channels through the synth panel.

The simple but powerful synthesizer allows the user to create a great bandwidth
of sounds, including beats, noises, basses and pads. Available synthesis types are:
rectangular wave with filter, FM, and LFSR noise generator. Rectangular wave and
LFSR sound similar to the Game Boy’s and other console’s soundchips but offer more
fine control and additional effects. The FM synth is the simple type with two sine wave
oscillators, with fixed base frequency and variable modulator frequency. An envelope /
LFO can be applied to modulation amplitude or frequency. For a sweeping spatial
effect, the modulator can be slightly detuned, with inverted phase for left/right. Each
synth channel is two-voice polyphonic and a stereo effect can be applied. Samples of
one second length @ 44 kHz, mono cannot just be used in a drum-machine style, but
also be pitched and played as notes.

Nanoloop allows for multiple recording sources for samples such as the built-in
microphone (iPhone only), a headset microphone (iPhone, iPod touch 2nd gen only)
and Nanoloop’s own sound output. The latter allows the user to create new samples
from scratch, using up to five channels to create one single sound.
Figure 11 Image of Nanoloop's UI for the iPhone

**Sound Effects**

For sound effect creation I chose the program As3sfxr. As3sfxr is a modified version of Sfxr. Usage of As3sfxr involves pressing a randomize button or manipulating any of the other numerous sliders that the user may adjust to manipulate sound attributes to create a specific sound. The sounds created with this application have the same chiptune style that is desired for the games over all sounds aesthetics. When a desired sound is created, the user may save the sound and export it to be used elsewhere.
While the game development process is underway, every 2 weeks we update our current build of the game with the new content we have created. We do this so that I can play the game for a short time to test the newly applied content before working on next week’s sprint. During these test phases I record any found issues, if I find an issue with the programming I report it to the programmers to correct, any other found issues are my problem to solve. In addition to our own personal beta testing I reached out to
the University of Florida’s club Game Makers Guild to have a few of their members test our game and give me feedback.

The alpha testing of Space Oddity with the Game Makers Guild took place on April 1st, 2014. Nine guild members volunteered to test the game and to fill out a small survey afterwards about their experience. These are the questions they were asked:

1) In a single week how many hours do you spend playing video games? _____

2) If this game was offered on a tablet, circle the likeliness you would want to play it on a tablet versus on a desktop computer
   (Strongly disinterested) (Disinterested) (Neither) (Interested) (Strongly Interested)

3) If this game could be 4 player, circle how interested you would be in this feature
   (Strongly disinterested) (Disinterested) (Neither) (Interested) (Strongly Interested)

4) Circle the game’s controls ease of use
   (Very Difficult) (Difficult) (Mild) (Easy) (Very Easy)

5) Circle the difficulty of play
   (Very Difficult) (Difficult) (Mild) (Easy) (Very Easy)

6) Circle the enjoyment you had playing the game
   (Very Unenjoyable) (Unenjoyable) (Neutral) (Enjoyable) (Very Enjoyable)

7) Circle how engaging the game was
   (Very Unengaging) (Unengaging) (Neutral) (Engaging) (Very Engaging)

8) Circle how you felt about the graphics
   (Very Unenjoyable) (Unenjoyable) (Neutral) (Enjoyable) (Very Enjoyable)

9) Circle how you felt the speed of play was
   (Too Slow) (Slow) (Just Right) (Fast) (Very Fast)

10) Circle how you felt about the soundtrack
    (Very Unenjoyable) (Unenjoyable) (Neutral) (Enjoyable) (Very Enjoyable)
11) Rate the chances of you wanting to play this game again
(Strong disinterested) (Disinterested) (Neither) (Interested) (Strongly Interested)

Here are my findings:

<table>
<thead>
<tr>
<th>Question #</th>
<th>Mean: 12.8</th>
<th>Median: 12.5</th>
<th>Mode: N/A</th>
<th>Highest: 30</th>
<th>Lowest: 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>1)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2)</td>
<td>Strong Disinterest: 1</td>
<td>Neither: 0</td>
<td>Interested: 4</td>
<td>Strongly Interested: 1</td>
<td></td>
</tr>
<tr>
<td>3)</td>
<td>Strong Disinterest: 1</td>
<td>Neither: 0</td>
<td>Interested: 3</td>
<td>Strongly Interested: 4</td>
<td></td>
</tr>
<tr>
<td>4)</td>
<td>Very Difficult: 0</td>
<td>Difficult: 2</td>
<td>Mild: 3</td>
<td>Easy: 3</td>
<td>Very Easy: 1</td>
</tr>
<tr>
<td>5)</td>
<td>Very Difficult: 0</td>
<td>Difficult: 6</td>
<td>Mild: 2</td>
<td>Easy: 1</td>
<td>Very Easy: 0</td>
</tr>
<tr>
<td>6)</td>
<td>Very Unenjoyable: 0</td>
<td>Unenjoyable: 0</td>
<td>Neutral: 0</td>
<td>Enjoyable: 4</td>
<td>Very Enjoyable: 5</td>
</tr>
<tr>
<td>7)</td>
<td>Very Unengaging: 0</td>
<td>Unengaging: 1</td>
<td>Neutral: 1</td>
<td>Engaging: 7</td>
<td>Very Engaging: 0</td>
</tr>
<tr>
<td>8)</td>
<td>Very Unenjoyable: 0</td>
<td>Unenjoyable: 0</td>
<td>Neutral: 1</td>
<td>Enjoyable: 5</td>
<td>Very Enjoyable: 3</td>
</tr>
<tr>
<td>9)</td>
<td>Too Slow: 0</td>
<td>Slow: 0</td>
<td>Just Right: 3</td>
<td>Fast: 4</td>
<td>Very Fast: 2</td>
</tr>
<tr>
<td>10)</td>
<td>Very Unenjoyable: 0</td>
<td>Unenjoyable: 0</td>
<td>Neutral: 2</td>
<td>Enjoyable: 3</td>
<td>Very Enjoyable: 1</td>
</tr>
<tr>
<td>11)</td>
<td>Strongly Disinterested: 0</td>
<td>Disinterested: 0</td>
<td>Neither: 0</td>
<td>Interested: 8</td>
<td>Strongly Interested: 1</td>
</tr>
</tbody>
</table>
Based on the responses gathered for survey question 7, 7 out of 9 respondents stated that they found the game engaging. 8 out of 9 surveyors are interested in playing the game again. 6 of 9 players felt the game was difficult. Based on the information gathered by the survey responses and the feedback given during the testing of the game it would appear that the game is enjoyable and meeting all of the desired characteristics. We strove to create a fun and exciting but difficult game, it would seem like we are on the right path to achieve this.

THE PROGRAMMER

The game engine used to create Space Oddity is MonoGame. MonoGame is an open source implementation of the Microsoft XNA 4.x Framework. Using MonoGame will allow us to port to multiple platforms such as Xbox 360, Windows & Windows Phone to port their games to the iOS, Android, Mac OS X, Linux and Windows 8 Metro. Although I did have two programmers create the code for the game, I did help with the coding as much as I could. Despite not knowing how to program, I was able to help by creating what’s called pseudocode. Pseudocode is an informal high level description of an operating principle of a computer program or algorithm. It uses the typical conventional structure of programming languages but is created for the purpose of reading rather than to be interpreted by a machine. An example of this would be what happens when the player uses the in game gadget, Personal Shield Generator:

On use: Item creates a force field static to players position that prevents (x damage) to player for duration based off a static decay rate of (x per second) and
for every (x damage) prevented (x energy) is consumed. If player’s energy pool reaches 0 the force field is turned off.

Other examples of pseudocode I used can be found in the Game Design Document: Space Oddity.

PROJECT MANAGEMENT

Organizing the creation of an entire game is quite a daunting task, especially if portions of the game creation process are divided up among multiple people who don’t regularly meet in an office. The difficulty of organizing the production of this project required me to use a few programs to help facilitated the development process of the game.

Programs Used

The programs I used to help maintain the development process of the game are web based, this was intentional, it made monitoring development easy and allowed for quick updates.

Trello

Trello is a free web based project management application made by Fog Creek Software. Trello uses a very specific pattern for managing project that’s known as kanban, which means signboard or billboard. Projects on Trello are represented by boards that contain task lists called lists for short, these lists contain cards. On these cards project goals or milestones can by written and they are intended to be moved from list to list, displaying their current progression. Users can be assigned to specific
cards along with due dates and checklists for them. I used this to help organize my personal weekly goals and group sprint sessions. A sprint is a term we used to define a set period of time during which specific work has to be completed to stay on track with our project deadline. Sprint durations usually spanned two weeks.

**Dropbox**

Dropbox is a file hosting service that is operated by Dropbox Inc. Dropbox offers cloud storage and file synchronization for its users. Dropbox allows users to install their program and with it create special folders on their computers; these special folders then will synchronize via the internet. This means anything that goes into this special folder or is adjusted while in the folder is updated to have the same changes made to it across any other computer that the Dropbox account is linked to. Any file placed in the folders can also be accessed from the Dropbox website or with the mobile application. Access to these folders can be shared among users based on the owner's preference, allowing multiple people to modify the contents of the folders. I used this application to share all of the art and sound assets I created for the game with the programmers to implement into the game.

**Google Docs**

Google Docs is a free web based office suite offered by Google. It allows users to create and edit documents online while collaborating in real time with other users. Google Docs includes features similar to word processing programs and spreadsheet programs. I used this program for its online spreadsheet capabilities. I used the spreadsheet program to create and share a working game design document with the
programmers. It was used to house tables of game world assets joined with pseudocode to help define their attributes and design.

**Facebook**

Facebook is an online social networking service. Facebook allows users to message one another directly in real time. Users can create messages to be sent out to entire groups of people. I used the messaging system on Facebook to create a group message for the project to allow constant communication between myself and the programmers, allowing any discrepancy or question to be resolved or responded to immediately. It was also used to schedule video chat meeting sessions that we used for group brainstorming or game based discussions. Facebook conveniently allows video calling services as well through the technology partner Skype which we took advantage of to maximize our time and to receive faster feedback.

**CONCLUSION**

This project was immensely involved and extremely time consuming, but I enjoyed every bit of it. I have gained an invaluable wealth of knowledge in regards to creating a game with a small team. I better understand the tasks that come with the divisions of labor when working on a project like a game. From this experience I am able to apply many of the organizational skills and tools I used to better myself and other future projects. I now have a better understanding of the amount of time and effort that goes into the many phases a game undertakes throughout its development. This project has better prepared me both in skill and understanding for the work I would like to do in the future. The overwhelmingly positive feedback I received in regards to the
design, soundtrack, and art style I developed for the game has not only encouraged me to peruse a possible carrier in game development, but has left me wanting to hone these skills even more.

LIST OF REFERENCES