Arctic Rush

Game Development In Unity

created

Spring 2015

by

Alex Odegaard
University of Alaska Anchorage

in association with

Scott Wheeler
Colorado State University

under the supervision of

Dr. Richard Webb
Kenai Peninsula College

and instruction of

Dr. Adriano Cavalcanti
University of Alaska Anchorage

This work is dedicated to those who have had ideas of creating something, but stopped before their vision could become a reality. May this project inspire you to go beyond the dreaming stage.

"Nothing starts until it begins."

Alex Odegaard

aodegaard@alaska.edu

Abstract

The purpose of this paper is leave an account of the three months spent developing my senior project at the University of Alaska Anchorage as a computer science major and the challenges that were encountered. The paper starts by mentioning my interest in computer science and videogames and how to combine those interests into a formidable capstone project. A large driving force behind the project is to take the reins as the project lead and interact with every aspect of the project to distance myself from becoming an “idea guy” -- a problem I’ve noticed with many people, including incoming computer science students.

The project began with the designing phase in early February where I would meet Scott Wheeler and Ryan Bergerson a few times each week over a combination of Skype and Google Docs to discuss a game concept and implementation strategy. In early March, Scott joined the team and coding began for what would be known as “Arctic Rush” -- a game based on the concept of the Oregon Trail, and inspired by the diphtheria serum run. In late March, we added my grandmother, and Ryan to the team to create original artwork for the game. In the span of three months, my team wrote 3440 lines of code and produced 14 pieces of original artwork with the prospect of a beta release within the next two months..

Acknowledgements

If there is anything I have learned while managing the development of Arctic Rush as part as of my senior capstone, is that it takes the support of many to ensure the success of an undertaking of this scale. The level of interest and feedback I received in my project from friends, family, acquaintances, and strangers was unprecedented. This support has led me to treat this project as more than just a school project, and as a result it will be completed with the consideration and polish that all of you deserve to see it in. I am taking every impression, suggestion, and criticism seriously to develop Arctic Rush the way we all believe is best.

Special Thanks

Ryan Bergerson - You had been pushing me for years to bring this project to life and when I finally decided to, you immediately offered whatever assistance you were capable of, and then some.

Scott Wheeler - There is absolutely no way I would have been able to implement all of the features that the project has now without your assistance on the project. Having you work on the coding aspect with me was great for productivity and motivation.

Carol Huber - You suddenly made the project's greatest weakness, it's greatest strength. Your artwork went above and beyond what I asked for every time and I truly appreciate the time you put in on my behalf.

Richard Webb - As I add the finishing touches for my capstone presentation, I am reminded how grateful I am that you responded to my frantic email about joining your class for a necessary amount of credits to graduate, mere hours before registration for classes was closed. I had no idea how central your class would become in the development of my project.

Table Of Contents

List Of Figures

Introduction

1.1 Background

Figure 1.1.1: Computer Science at UAA.

I started attending the University of Alaska Anchorage in the Fall of 2010 with the plan of majoring in electrical engineering, but switched to computer science the following semester after taking my first programming course, CS 201. I was fascinated at the interaction of programming languages as input code to produce an output to the computer. With the hardware of today at such an advanced level, the limit to what we can produce is directly related to our understanding of its capabilities and our own creativity.

This project initially began in the Spring of 2012, but never made it past the idea stage. When I considered the final product after all of brainstorming, what I saw was a colossal undertaking that went beyond the skills I had available. It wouldn't be until the Fall of 2014 that I would become acquainted with the Agile methodology through a software development class (CSCE A401), and start to think of how my project could be divided up into workable pieces. I had always been aware of divide and conquer strategies, but the pace and accountability that came as part of the Agile process would help lay the foundation for my revitalization of this project. In

the Spring of 2015, capstone was brought to the table and I decided to pursue my plan from three years ago to develop a game to showcase the skills I had developed during my time at UAA.

1.2 Getting Started

During the first few weeks of February, I was wavering between ideas as to which project to adopt as my final entry as a senior at UAA. I took into consideration how much I could accomplish in the short timespan given, and whether or not the topic chosen would be interesting enough to keep an audience at the concluding presentation. Though it would be perfectly acceptable to not finish a project if it was ambitious enough, my goal was to have a working concept and end my chapter at UAA on a high note.

Having just come from software engineering (CSCE A401) as a part of a five man team tasked with creating a game that utilized eye-tracking for the purpose of enticing prospective computer science majors, I was familiar with the amount of work that went into creating a game from scratch. I knew it would be a lot of work and that I would never reach the level I was planning to without another hand. As a result, I reached out to my long-time friend, Scott Wheeler, a fellow computer science major studying at Colorado State University with the idea of collaborating on the project together. Despite his own obligations as a student at CSU, and a member of its ultimate frisbee team, he agreed to collaborate on the project with the hope that it could turn into a highlight on his resume.

We decided to create a game based off The Oregon Trail video game that was published by MECC in the early 1970s. Despite its age and simplicity, it has a charm which has caused it to become a legend in the world of video games. [1] Neither Scott nor I were comfortable with our ability to make a game based on graphics, so we welcomed the simplicity that came with creating a game after the Oregon Trail.

1.3 Motivation

This capstone was designed to apply everything I learned during my time at UAA, both inside and outside of the classroom. The code showcases my technical skills, the team environment addresses my communication skills, and the overall project development demonstrates my management skills.

To those unfamiliar with the process, developing a game can seem like an immature project. In reality, developing a game takes the same kind of planning and considerations that other projects do, but with the added stipulation that it must be fun. I'm confident that the idea behind the game is solid, and has the ability to reach its audience if it is done well which presents the possibility of turning a hobby into something more.

Regardless of what success awaits this project, it will showcase my technical skills to potential employers. Learning Unity and reaching a high enough level in it will show potential employers how quickly I can become proficient with an unfamiliar technology.

My greatest motivation with this project is completing it for eager friends and family of whom I've mentioned the project to.

1.4 Implementation

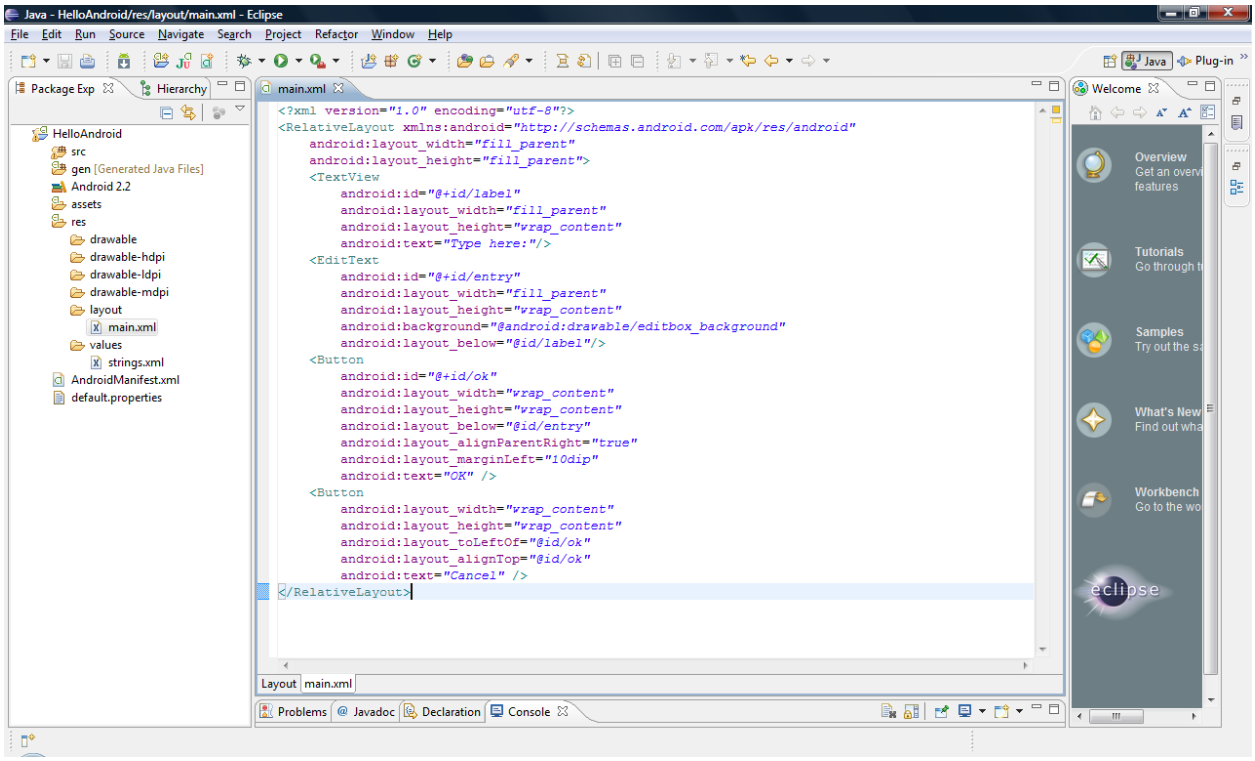


Figure 1.4.1: Developing for Android Through Eclipse.

My original plan was to develop the project in Eclipse using the Android Developer Tools plug-in since I was familiar with the Eclipse IDE from all of my programming courses, and even the ADT plug-in from an Android Programming course (CSCE A305). Despite the exposure, I was still skeptical as to whether or not I'd be able to faithfully develop Arctic Rush to the standards I had envisioned for it. A year earlier, I tried my hand at developing a simple Sudoku application for Android, and the results discouraged me from believing I'd be able to create an impressive capstone project within the time remaining.

On January 23rd, at around 3:00 pm, two hours before registration for classes was closed, I was frantically searching for another course to add, and came across an Intro to Game Development course (CS 109). I immediately e-mailed the instructor for permission to add the course, hoping for the best that he would get back to me in time with a positive response. I had emailed other instructors hoping to gain access to their class, but did not receive a response in time, or I was

rejected because the classes were already almost two weeks in. Thankfully, Dr. Richard Webb, the instructor of CS 109 was actively checking his e-mail and got back to me within a half hour with his contact information. I called him immediately and discussed my background with computer science and that I'd be able to make up for the lost time. I was accepted into the class with an hour to spare before registration was closed. At the time, I had no idea how much this decision would affect my final project.

Figure 1.4.2: Unity Logo.

It was in CS 109 that I was first exposed to Unity. Unity is a cross-platform game engine developed by Unity Technologies and used to develop video games for PC/Mac OS, various mobile devices, websites, and gaming consoles. [2]. Until the Unreal 4 Engine became free in March 2015, Unity was one of the few existing game engines that offered full functionality without an additional price.

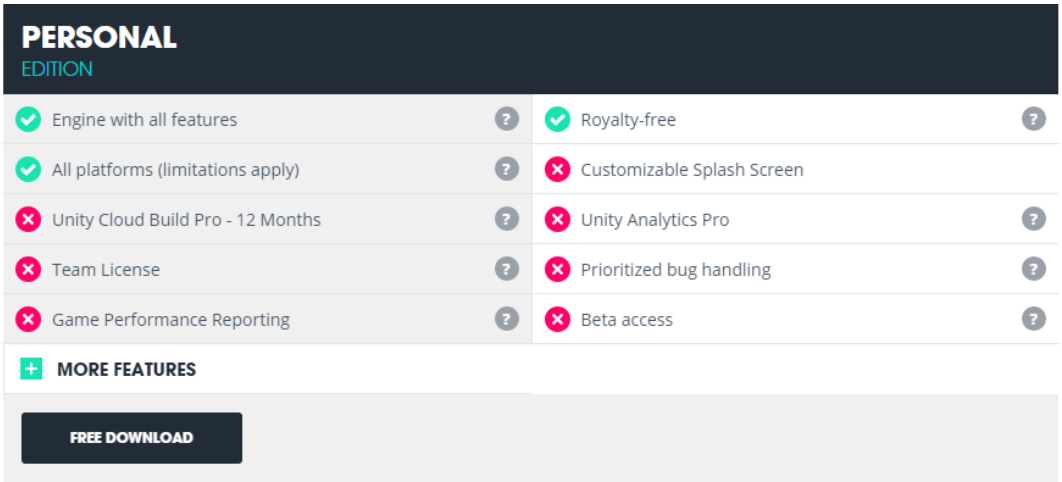


Figure 1.4.3: Features of Unity Personal Edition.

Design Process

2.1 Scope

Defining scope is perhaps the most important aspect of any project and can determine whether or not a project is successful. If scope is left unchecked, it's possible to keeping adding new features to the project until it collapses under it's own weight. In order for a project to be finished within a deadline, the scope must be defined ruthlessly and leave no gray areas.

For this project the goal was to create a simple game that emulated the gameplay of The Oregon Trail, where the player is tasked with braving random events from point A to point B with the help of the supplies they have and the choices they make.

2.2 Game Concept

...

2.3 User Interface

At the time, The Oregon Trail's user interface was acceptable, but to a generation that has grown up in a more advanced age of videogames, it can seem dated and clumsy. In the Oregon Trail, decisions were made with 'y' or 'n', and items were selected with the number keys. Figure 2.3.1 illustrates what shopping was like in The Oregon Trail. To buy 3 oxen for example you would need to type 1, and then 3.

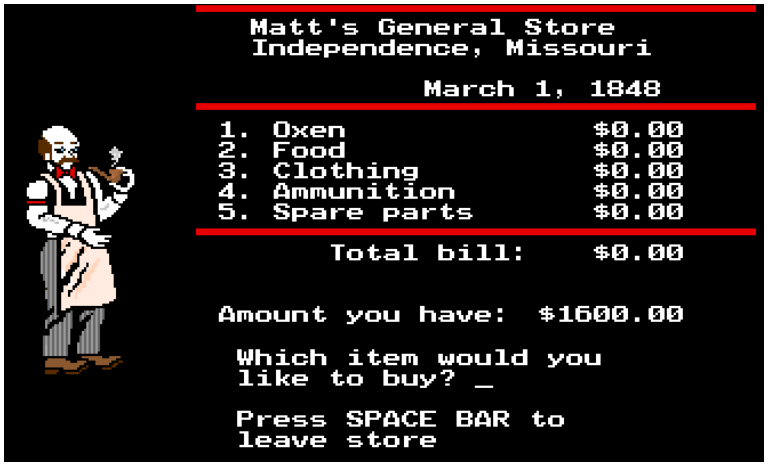


Figure 2.3.1: Oregon Trail Shop Interface.

In Arctic Rush, almost all choices and actions are made by the touch of a button. Besides simplifying input, the use of buttons allows for easy porting to mobile platforms.

Figure 2.3.2: Arctic Rush Shop Interface.

2.4 Art Style

Though Arctic Rush is intended to be inspired by The Oregon Trail, the user interface design prompted the art direction to take a more modern look. To match the game's serious atmosphere we wanted artwork to be more on the spectrum of realistic than toonish and colored in darker / duller tones.

Development Process

3.1 Code Development
3.2 Artwork Development

3.1 Code Development

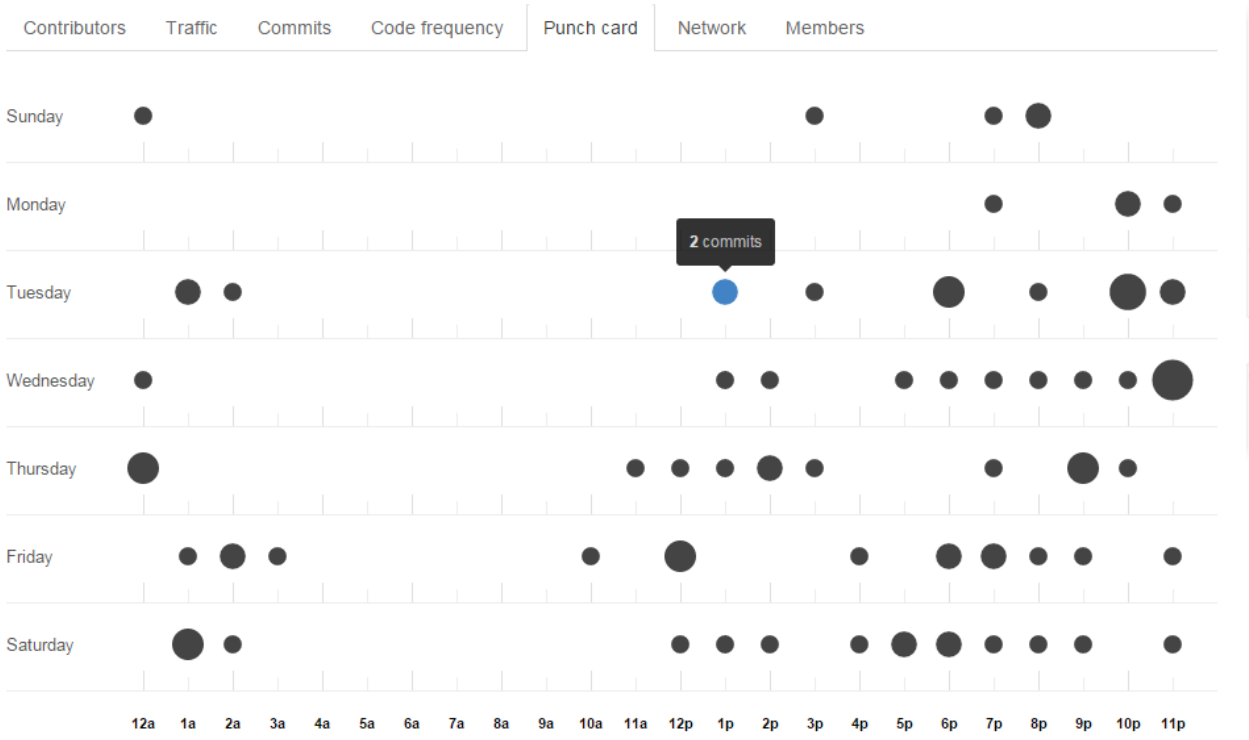


Figure 3.1.1: Coding Schedule.

Scott and I were in charge of writing the code for Arctic Rush. All of the code was written in what Unity calls JavaScript (though it is closer to ECMAScript). Since Scott was completely new to Unity, and I was going beyond topics covered in CS 109, we spent a lot of our time combing the Unity Documentation. [3].

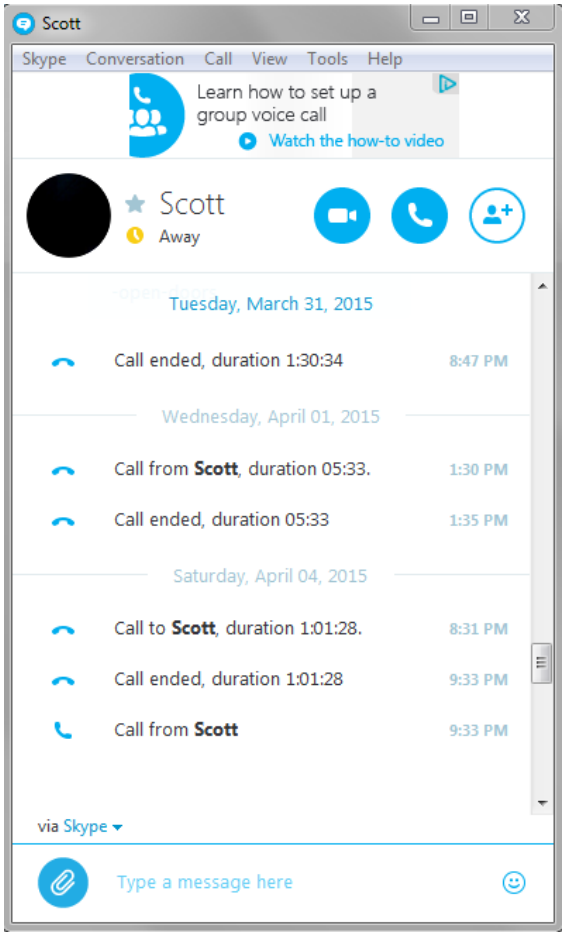


Figure 3.1.2: Regular Skype Meetings.

We met multiple times each week to assign new user stories, and report progress on the previous week's user stories. To keep things simple and keep branching code to a minimum, Scott and I would choose user stories that didn't rely on the other person completing theirs a certain way. Meeting multiple times throughout the project motivated both of us to have something to show for the other. Often times, if one of us put in a large commit, the other would reciprocate it. This positive feedback loop allowed us to produce a lot of code in a short time.

3.2 Artwork Development

Figure 3.2.1: Sled Dog Concept Art.

During the initial planning phase, it was uncertain as to how artwork would be handled. Neither Scott nor I had any particular artistic talent, be it traditional or digital. At first we held out hope that we would find acceptable royalty free artwork, but ultimately found the selection and quality dismal. In order to keep our vision of a more modern take on The Oregon Trail intact, we were going to need additional help.

In late March, I was talking about the current status of my senior project at a family dinner, and mentioned that things were coming along but we were in dire need of an artist. My mother mentioned that my grandmother had artistic talent and might be able to help us. The next day I met my grandmother for coffee and looked over her portfolio and knew right away that she would make a great addition to the project.

After a trip to the art supplies store, my grandmother went straight to work creating art assets for the game. On completion of a drawing, I would pick it up and scan it into the computer at a high resolution and assign another drawing to be done. If the drawing was something that she wasn't familiar with drawing (such as the general store), I would provide references to existing artwork

that was roughly what we were looking for. This allowed us to get exactly what we wanted from specific pictures and create a perfect hybrid.

Once the artwork started to amass, we began to look into the idea of adding color to the drawings. Despite all the artistic ability of my grandmother, she was only one person and having her to both draw and color the artwork would cause the flow of art assets to lessen -- something that would cause serious delays in the production process.

In early April, we added Ryan Bergerson to the team to help with the art. The previous Summer he taught himself basic PhotoShop, and was confident he would be able to add color to the detailed pencil sketches that we were receiving. So far, the results have been promising. All work by Ryan has been done using GIMP.

Figure 3.2.2: Coloring Process - Before and After.

Testing Process

4.1 Testing

4.2 Future Testing

4.1 Testing

Since the project was developed with the a variation of the scrum methodology, testing was done incrementally as new features were added. While incremental testing can be time consuming, it has the distinct advantage of finding defects in components early on before they become buried in the logic of the system and other components come to rely on them.

After each user story was submitted, the scenes that relied on the submitted work were ran and values of variables were examined in the inspector. Testing was further simplified by the time spent in the design phase. By having a document of every action the user could take, and every reaction by the system, we were able to quickly root out any unintended output.

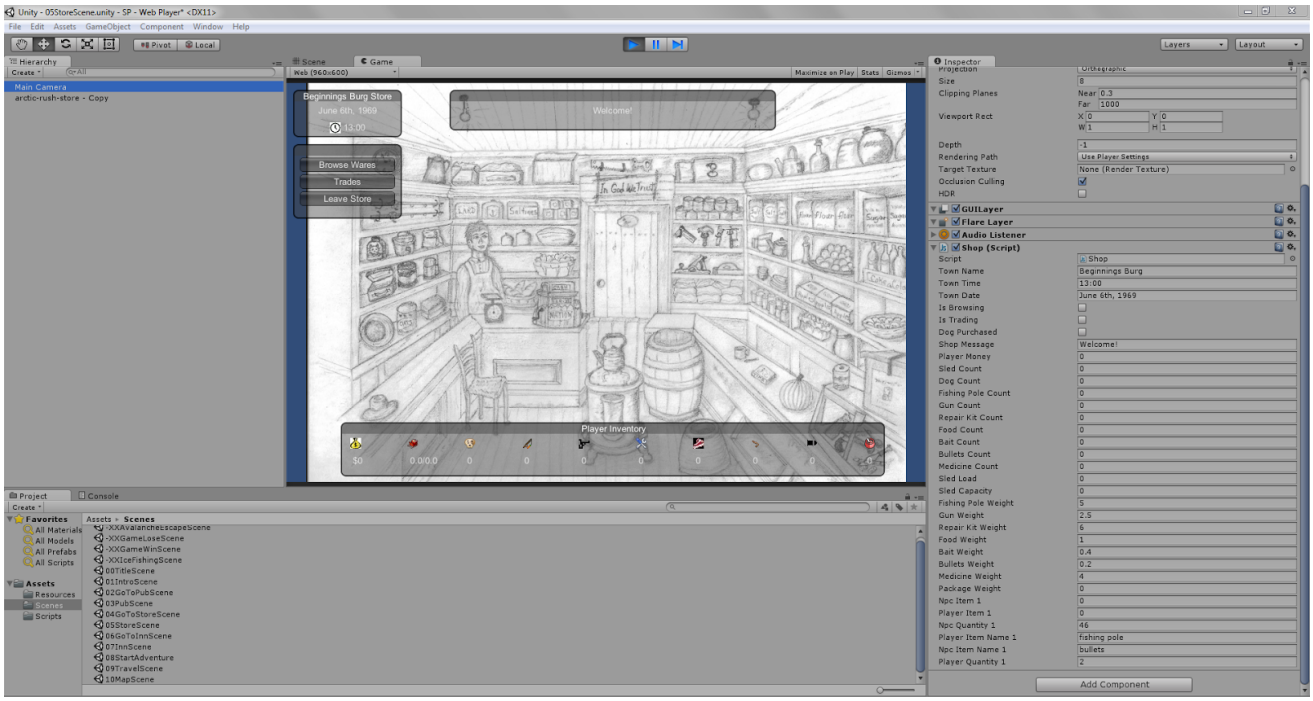
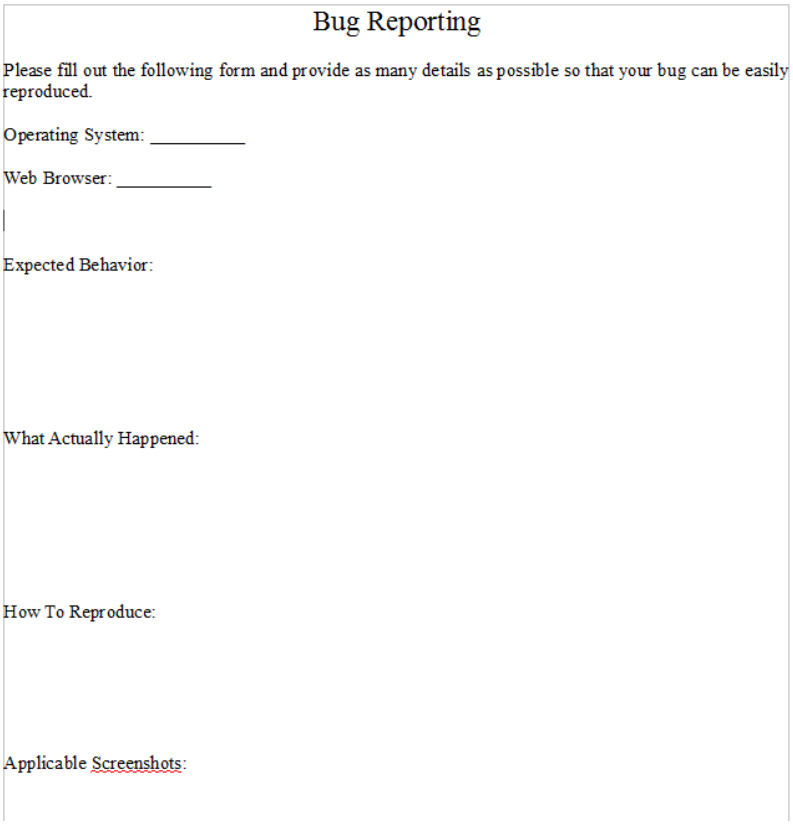


Figure 4.1.1: Unity Playtesting.

4.2 Future Testing

No matter how confident our team is that we've caught all of the currently existing bugs, we are humble enough to realize that as humans we are capable of making mistakes. Rather than allow these bugs to reach the post-production phase, there will be a beta testing round in an attempt to catch any remaining bugs, and gain valuable feedback about the game's appeal.

The beta testing pool will consist of roughly a dozen individuals with different levels of experience with beta testing and playing games.



Bug Reporting

Please fill out the following form and provide as many details as possible so that your bug can be easily reproduced.

Operating System: _____

Web Browser: _____

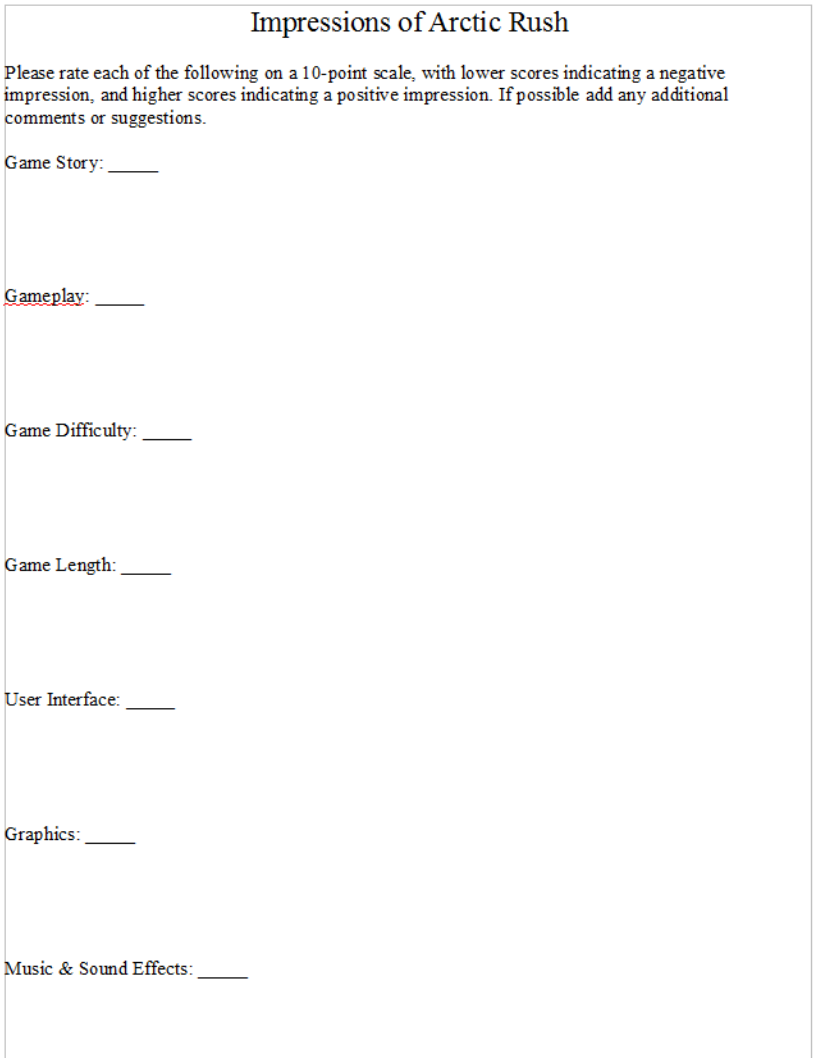
Expected Behavior:

What Actually Happened:

How To Reproduce:

Applicable Screenshots:

Figure 4.2.1: Example Bug Reporting Form.



Impressions of Arctic Rush

Please rate each of the following on a 10-point scale, with lower scores indicating a negative impression, and higher scores indicating a positive impression. If possible add any additional comments or suggestions.

Game Story: _____

Gameplay: _____

Game Difficulty: _____

Game Length: _____

User Interface: _____

Graphics: _____

Music & Sound Effects: _____

Figure 4.2.1: Example Feedback Form.

User Manual

5.1 Setup
5.2 About
5.3 Controls
5.4 Tips & Tricks

5.1 Setup

Step 1 - Start by opening your favorite web browser and typing in the following address: https://github.com/aodegaa/SP.

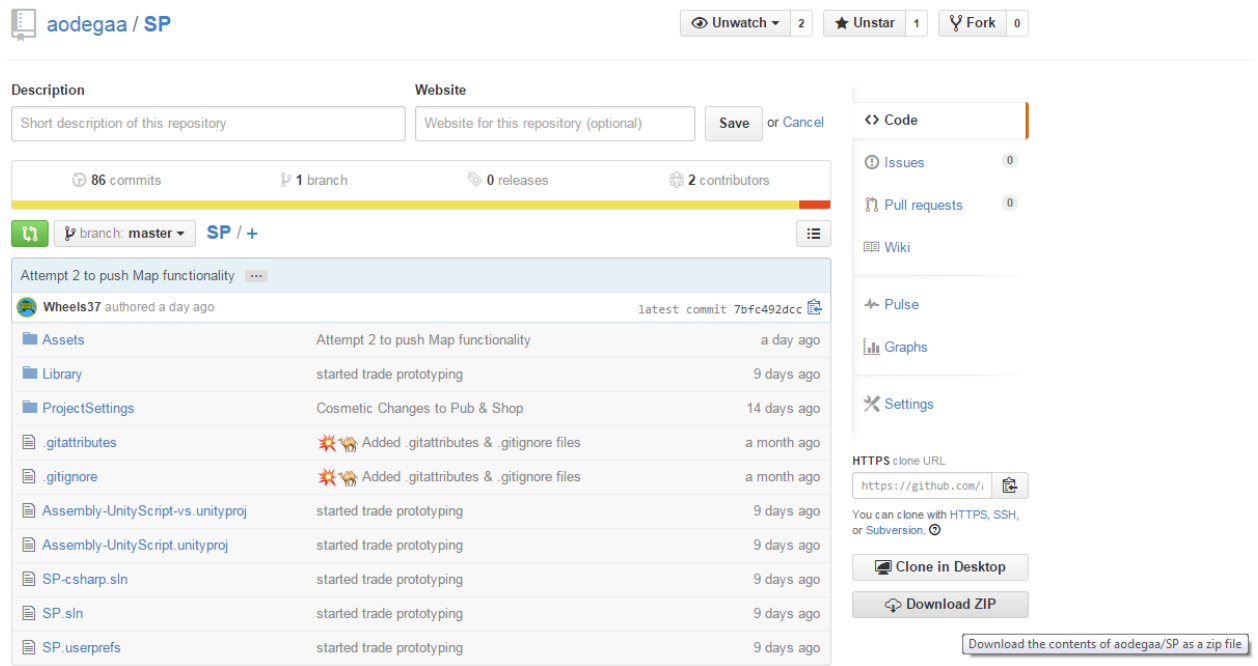


Figure 5.1.1: GitHub Repository for Arctic Rush.

Step 2 - Click “Download ZIP” to add the contents of the repository to your computer.

Step 3 - Find the ZIP folder on your computer and extract it.

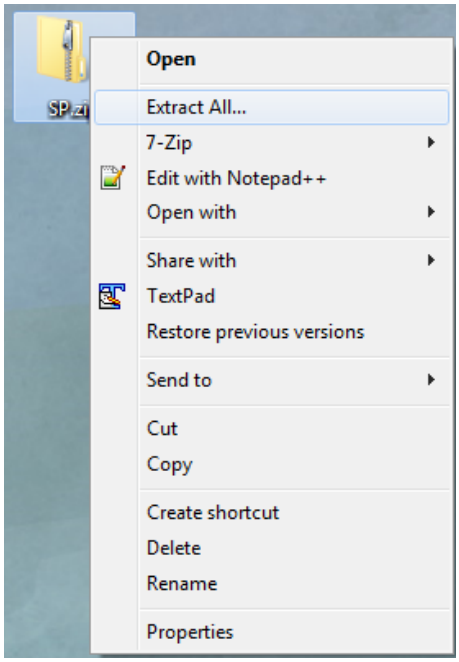


Figure 5.1.2: Extracting the ZIP File.

Step 4 - Open the extracted folder and navigate to the "Assets" folder. Inside you will find "Arctic Rush.html". To launch the game, simply double click the html file.

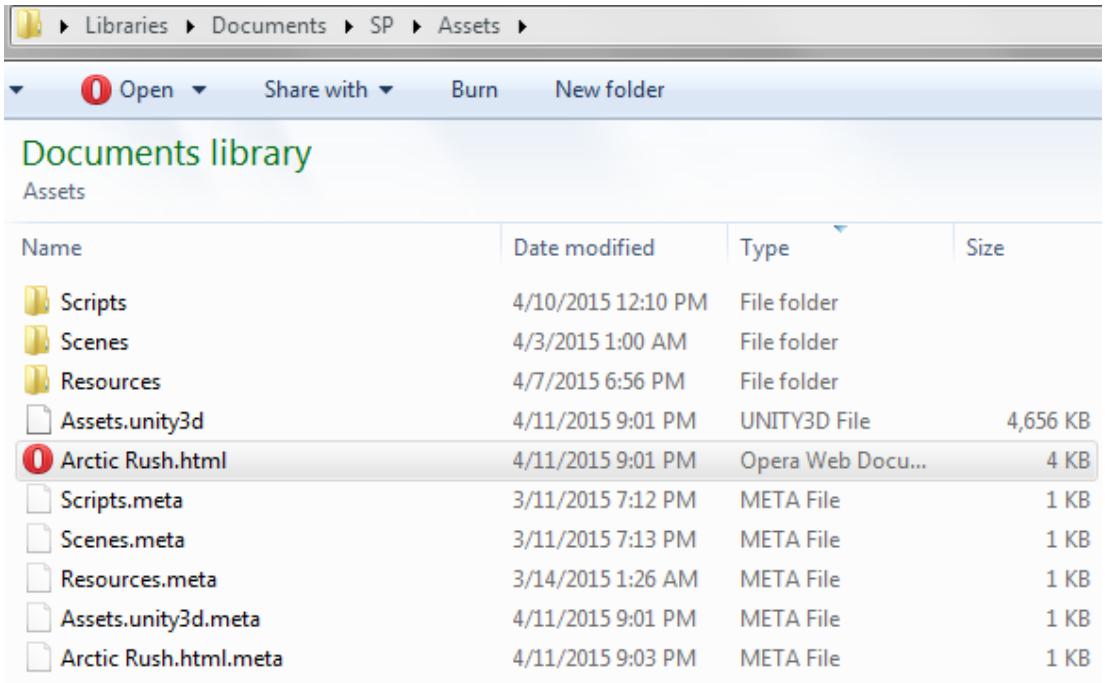


Figure 5.1.3: Locating the HTML File in Assets Folder.

Step 5 - If you do not already have the Unity Web Player Plug-In, you will need to install it. Simply click the icon and go through the installation process.

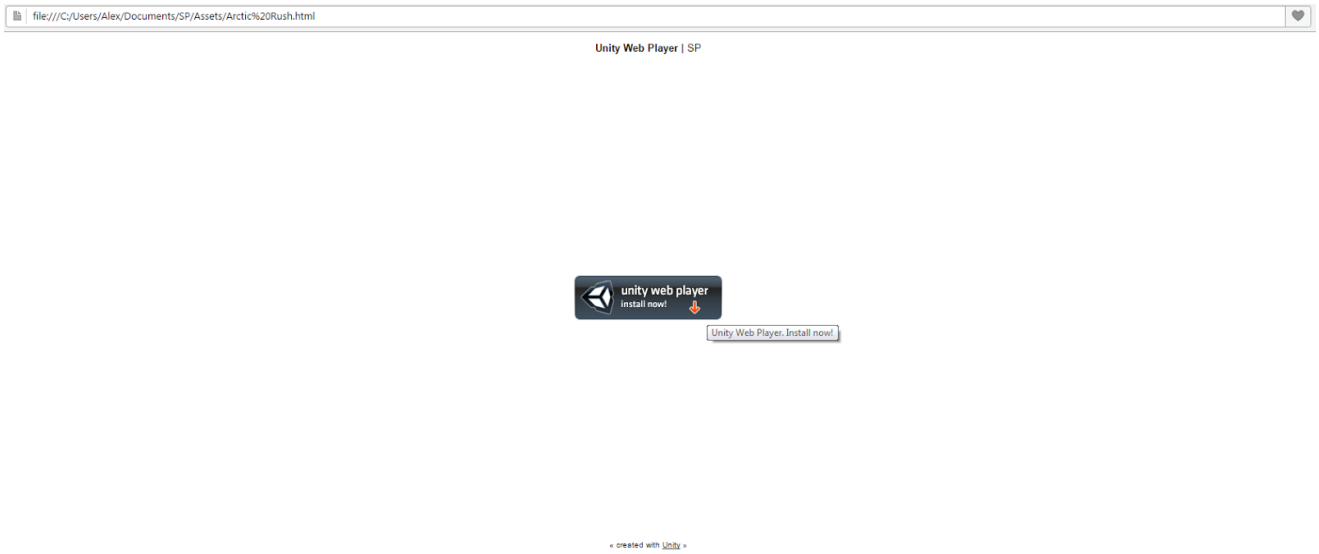


Figure 5.1.4: Installing the Unity Web Player Plug-In.

Step 6 - After the Unity Web Player Plug-In has been installed, refresh the page and the game should be loaded. Simply hit “New Game” to begin your adventure. (Note: Some web browsers require you to allow the plug-in after adding it. Look in the upper right hand corner of your browser for a pop-up and allow the plug-in to run.)

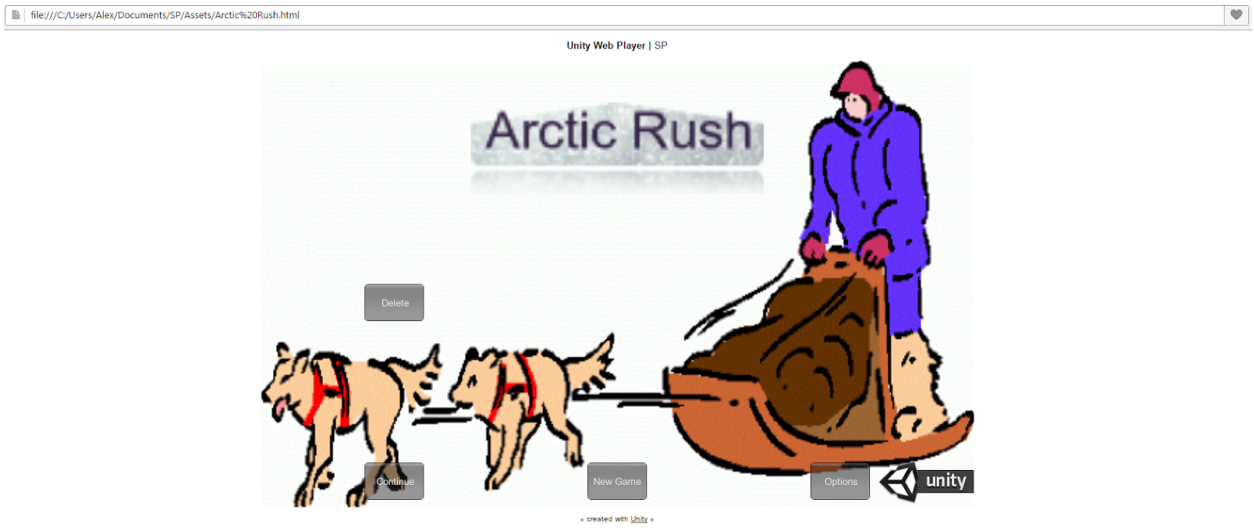


Figure 5.1.5: Arctic Rush Title Screen.

5.2 About

Arctic Rush is inspired by the story of the 1925 serum run to Nome, where diphtheria antitoxin was transported by dog sled over 674 miles to save the small town of Nome, Alaska from an epidemic.

The gameplay is based off of the popular 1985 Apple II classic, The Oregon Trail, a largely text-based adventure that put the player into the shoes of a pioneer making their way to Oregon's Willamette Valley from Independence Missouri by covered wagon in 1848.

In Arctic Rush, you assume the role of a rookie musher whose town has started to come down with a mysterious sickness. A cure has been developed, but it is several hundred miles away in the town of Haven, and must be picked up and delivered before the citizens of your town succumb to the illness. You start at the leftmost node, and must make your way to the rightmost node and then return as quickly as possible. Along the way you will encounter many situations that test your skills as a musher. Can you survive the unforgiving wilderness and return to your town as a hero?

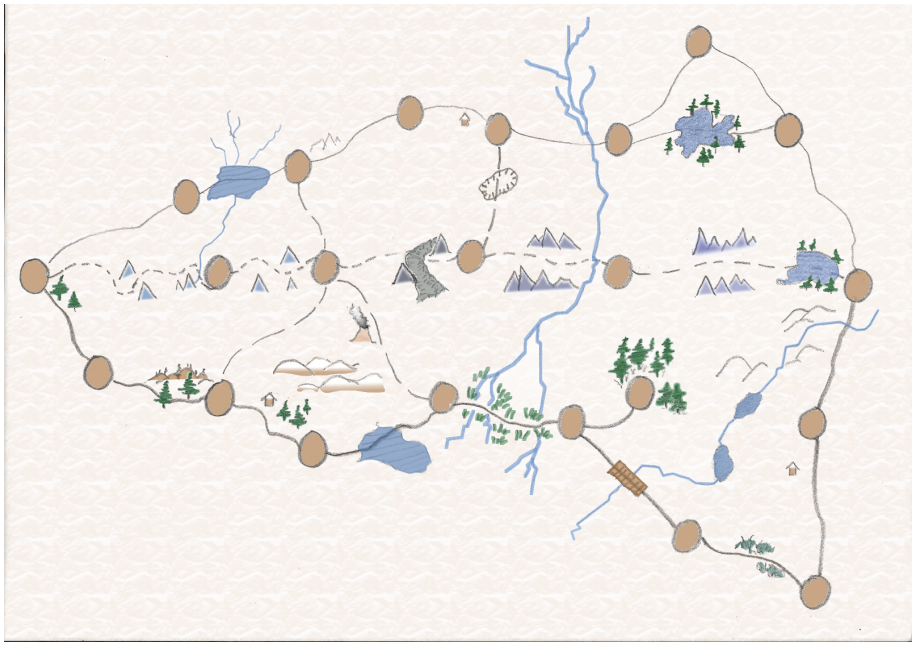


Figure 5.2.1: The Map of Arctic Rush

5.3 Controls

Mouse: Make decisions.
Interact with buttons.

Keyboard: Name entry.
Movement during avalanche scene. (WASD / Directional Arrows)

5.4 Tips & Tricks

On Movement Speed

- The healthier your dogs are, the faster you will move.
- The more dogs you have, the faster you will move.
- The less you are carrying, the faster you will move.
- If you have already taken the path on your way to Haven, you will move slightly faster on the way back.

On Final Score

- The more people saved, the greater the points awarded.
- The quicker your journey, the greater the points awarded.
- Your remaining supplies count towards your score.
- Difficulty affects the score modifier.
- Points are lost each time a dog passes away.

Conclusion

6.1 Project Results & Review
6.2 Future Work
6.3 Parting Words

6.1 Project Results & Review

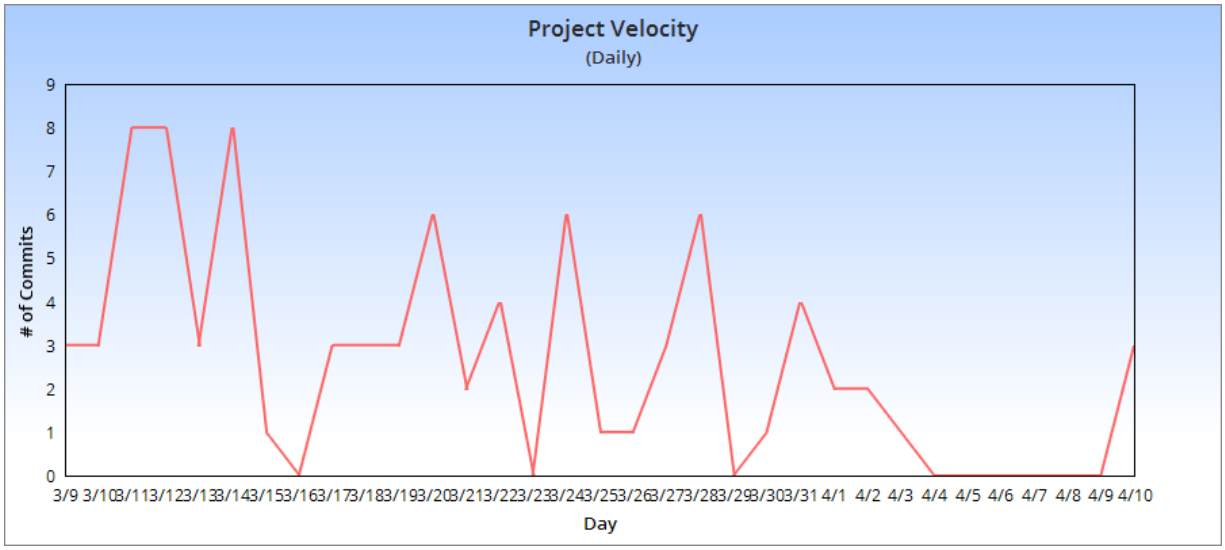


Figure 6.1.1: Project Velocity Shown As Commits Per Day.

Development of this project start March 9th, and paused April 14th. In that span of time we were able to write 3440 lines of code, and produce 14 pieces of original artwork, making this the greatest undertaking we have ever been involved with. Figure 6.1.1 shows a rough summary of the project's velocity through commits per day, while Figure 6.1.2 shows a rough summary of each programmer's contribution to the project. There was a strong initial push made during spring break. After that momentum slowed as classes resumed and more demanding user stories arose, until eventually momentum was lost when I moved to writing documentation, and Scott lost direction in the project. During our time we created the foundation for the game, a basic tutorial, various buildings, a working store and inventory system, character creation, health bars, rumors in the pub, basic user interface, travelling and updating map. Currently the outstanding user stories include general user interface optimizations for each scene, finalized artwork for various scenes, a polished tutorial, trading in the store, jobs in the pub, an implemented inn, town

names, balancing of game mechanics, finalized ice fishing and avalanche mini-games, player and dog attrition while travelling, random events, win/lose conditions, options, and music. While this may seem like quite a bit, we have already implemented proofs of concepts for many of the user stories that will help with development.

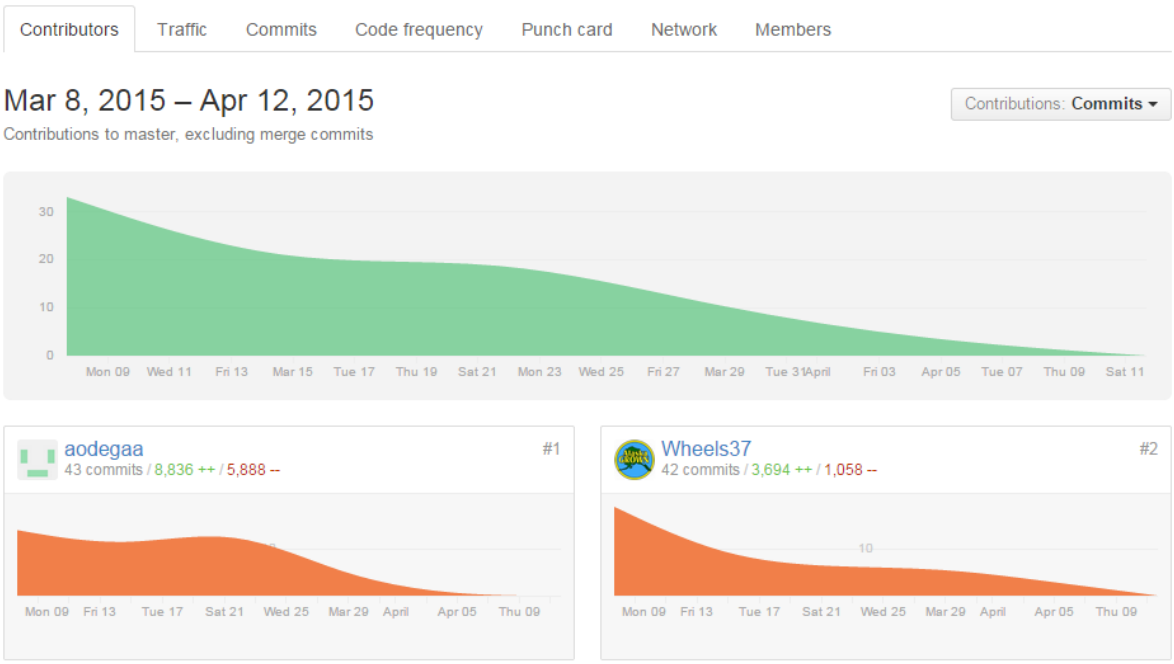


Figure 6.1.2: Commits By Each Member.

6.2 Future Work

Work will resume on the project after a brief respite from finals week, likely around May 11th. From there, the first task is to complete outstanding user stories that were unable to be finished during capstone. These user stories consist of implementing the remaining pieces of the game, and overall cleaning up the user interface and art assets in preparations for the beta testing round.

A beta testing round will last approximately two weeks, during which the beta testers will play the game, search for bugs, and submit their overall impressions of the game. While the closed beta session is going on, Scott and I will be optimizing the game for release to Kongregate.com, a flash games portal that accepts games created in Unity. For better exposure from Kongregate,

we will also be implementing their API in our game to allow for leaderboards, and game specific achievements.

Once the results from the beta are in, final optimizations and changes will be made if needed. The game will be submitted first to FGL.com, a website that allows thousands of registered publishers to view and bid on your game. Before submitting the game to Kongregate.com for the world to see and review, we will wait take about two weeks to see if there are any offers through FGL.com. From here, everything depends on the response of the community to Arctic Rush. If the response is positive, we will produce the game on another platform, and repeat the optimization phase.

6.3 Parting Words

Overall, I'm very pleased with what my team and I were able to create in the span of a few months. The only thing I would change about the development of this project is introducing artwork earlier. Had I noticed the potential my grandmother had as a traditional artist earlier, drawings could have been contracted earlier and have led to a more polished demo for capstone.

There were many risks involved with taking on a project of this scale with such a short timeline. Since my teammates were volunteers, keeping them motivated and involved with the project could be challenging at times. If it weren't for the momentum provided by the scrum methodology, where new features were rolled out on an almost daily basis, it is quite possible our interest in the project would waned before we could have delivered in time.

The scope had to be defined ruthlessly to assure that the project would not collapse upon itself. If there is one thing to take away from this project, it is to define the scope of your project early on, such that you will be confident that you will be able to finish it before you lose interest in it.

Appendix A: Additional Figures

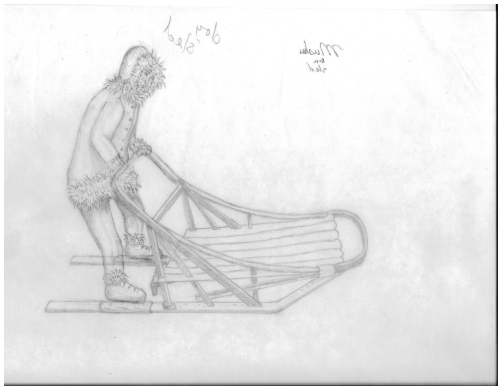


Figure A.1.1: Musher and Sled Concept Art.

Figure A.2.1: Bear Encounter Concept Art.

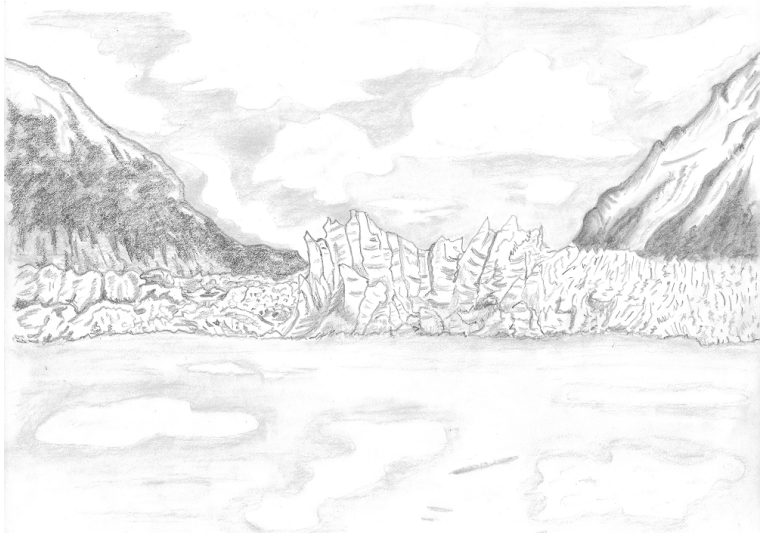


Figure A.3.1: Glacier Scene.

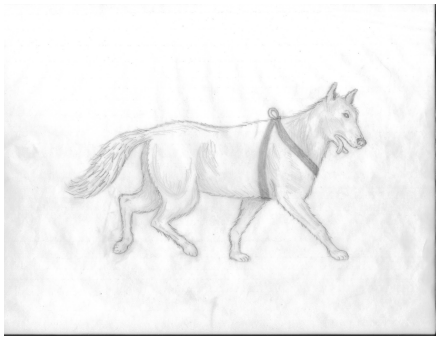


Figure A.4.1: Dog Movement Frame.

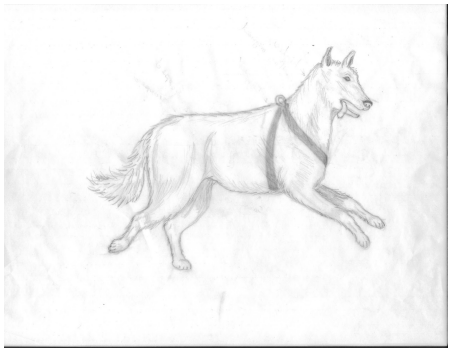


Figure A.4.2: Dog Movement Frame.

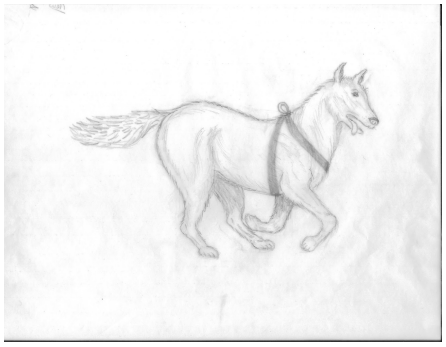


Figure A.4.3: Dog Movement Frame.

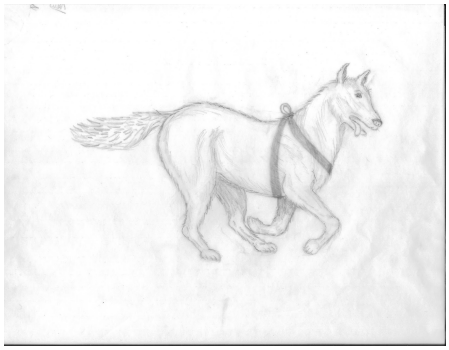


Figure A.4.4: Dog Movement Frame.

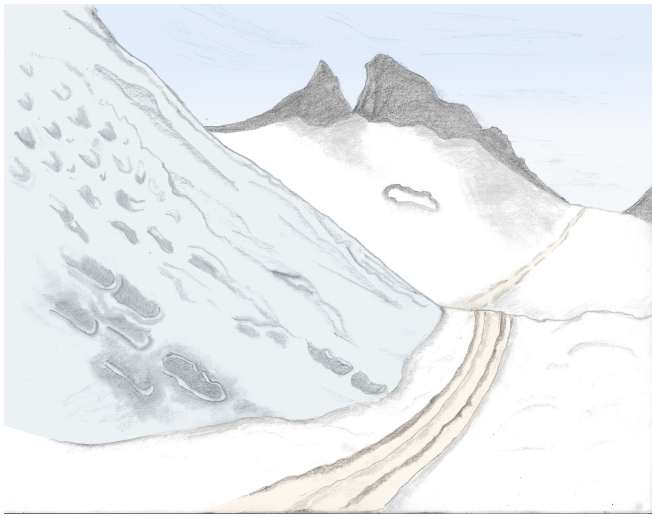


Figure A.5.1: Mountain Scene.

Figure A.6.1: Implemented Store Rough Draft.

Appendix B: Planning Stage

Oregon Trail Inspired Game for Web

http://gamedev.stackexchange.com/questions/2467/how-do-you-start-development-of-a-game

github account names:
Wheels37
aodegaa

Description:

- Sled dog theme;
- Iditarod-based
- Setting: Goldrush times (1850s-1900)
- Build in Unity engine?

Objective:

- Brave the challenges and survive to the end.
- Most points, send the medicine

Features:

- Use RNG seeds.
- Barter items
- Multiple paths to the final destination (some have greater risk/reward)
- Stores in major stops (prices generally increase. can depend on the population of the stop)
- ?Ice fishing (at the cost of bait you can play a reaction minigame to increase food stores)
- ?Multiple difficulties. (could affect RNG or starting resources)
- High scores (local) | ?based on… surviving members, items, time taken, etc.
- Save session. (store other data as well)

Elements the user can control:

- Pace (slow, medium, fast)
- At the beginning of each “turn” or “tick” the user can decide whether to advance or stop (may be better to have it auto advance at a slow rate so the user doesn’t have to keep clicking “advance”.)

- Food consumption rate (people food and dog food, both can be consumed by either party)

Events:

- Crossing a frozen lake (pass/fail based on weight and other factors?, or slows progress?)
- Blizzards (can potentially last for many turns | during a blizzard, harder to travel, can delay supply drops, parts break)
- white-outs (can't see, no progress for several hours)
- Random diseases (dysentery, hypothermia, broken limbs, cholera, , etc)
- dogs can get tangled (if more than 1 dog)

Resources:

dogs
dog food
human food (fish?)
bait (for fishing?)
?sled parts?
?fire starting tools?
medicine (bonus points, also heals)

Objects:

- Doge (max of 8)
    - hunger between 0 and 100; affects performance, life
    - life; affected by hunger, sleep, etc
    - accessories (booties, hats, monocles, visors, etc.)
    - breeds: malamutes, or siberian huskies
- Main Character

Out of Scope (for now)

?Support a multiplayer (can compete on same seed at synchronized pace)

Progression of Events

- Start App
    - New Game -> step 1
    - Continue
    - About
    -

- Step 1
    - Some sort of character creation
        - Character name
        - ?Stats?

- Step 2
    - buy doges/supplies at general store
    - name dogs

Music?:
http://audionautix.com/~audion/Music/ModernCombat.mp3
Night-time music
http://audionautix.com/~audion/Music/MiddleEarth.mp3
LOSE MUSIC
http://audionautix.com/~audion/Music/LegendsOfTheRiver.mp3
Cruising music

Things to consider:
- max rate is based on the slowest dog

Variables:
- sled 100lb ~$500
- equipment/player weight
    - food 6,25lb/dog/day (at avg pace) + 5lb/day for musher @
    - dog stuff
        - booties
        - straw beds
    - spare parts

        - Runners
        - basket
    - medicine? 20lb per 100 people (100 doses weighs 20lbs)
- max rate 12 mph (8 dogs full stats)
- min rate (probably shouldn't be 0) 5 mph
- standard rate (one dog, full stats) 8 mph

Formulas:

- rate as a function of number of dogs 100% + 60% per dog
- rate as a function of hunger/fatigue/health (either separate formulas or one master formula)
- dog hunger increase as a function of rate 25% per day
- player hunger/fatigue increase
- amount fatigue restored by resting as a function of time

References

[1] Lipinski, Jed. “The Legend of The Oregon Trail”
https://web.archive.org/web/20130731090300/http://mentalfloss.com/article/51930/legend-oregon-trail

[2] “Unity - Game Engine”
http://unity3d.com

[3] “Unity - Manual: Unity Manual”
http://docs.unity3d.com/Manual/index.html

[4] “Kongregate Developers”
http://developers.kongregate.com

[5] “FGL.com - The place indie game developers go”
https://www.fgl.com