Design Document for:

Squirvival

A Squirrel-Based Survival Game

"Can you squirvive?!"™

Written by: Thomas Allenbaugh, Stephen Erickson, Daniel Frecka, Jared Hagans, Cameron Hopkins, Brandon Patridge

Version # 1.2

Wednesday, April 23, 2014

Design History

This is a brief explanation of the history of this document.

Version 1.0

Created the initial document.

Version 1.1

Added additional details about multiplayer player roles and the player's home. Added the boar to list of AI added a description.

Version 1.2

Changed up details to match what we accomplished over the semester rather than what we hoped to accomplish

Game Overview

Philosophy

Losing Is Fun

The game should be difficult in a way that evokes amusement and/or the spirit of competition instead of frustration. By making a fun-to-play game with an easy learning curve, which doesn't ask the player to invest any time getting used to or preparing to play the game, it is our hope that short and fatal playthroughs shall be perceived as small packets of frantic and unpredictable fun rather than as a hellish and frustrating slog.

Replayability

Replayability and new experiences are very valuable for survival games and is a major goal for Squirvival. Although we are drawing from a rather small pool of assets with which to populate the world, we want to let the random nature of the level generation introduce lots of different situations and challenges for the player to deal with in multiple

playthroughs, with the aim of reducing overall repetition.

Fluid Controls

Controls should be fluid and flexible. Games can be fun by being able to move through the world easily and in novel ways. We want our game to utilize intuitive controls that offer the player the freedom to have fun just by moving around in the world.

Common Questions

What is the game?

Squirvival is presented in third-person perspective, and contains a few game modes. In the solo mode, the player takes control of a flying squirrel attempting to gather enough food (Nuts) before Fall ends and winter comes. To survive, the player must explore the procedurally-generated forest environment to find nuts while avoiding ground and air based predators who will attempt to corner and eat the player whenever possible.

In the multiplayer game modes, you can team up with other squirrel players against predator players on a private server, or compete squirrel-against-squirrel in a race for the nuts.

Finally, Parkour Mode lets you practice your flying and climbing skills, to traverse a tree-based obstacle course over deadly terrain.

Why create this game?

Of the many survival themed games out on the market, very few of them involve playing as animals and less as squirrels. With their agile movement and tree climbing it seemed like it would be a fun creature to play as. Also, their small scale will allow a different perspective than the typical human ones seen in many character-centered games.

Where does the game take place?

The game takes place in various deciduous environments; low to high density forest, grasslands, and mountainous areas. Time passes as you run around, marked by the shadows under the trees, around and on which you run and climb.

What do I control?

In the single-player modes, you can only be a flying squirrel of a random color. In the multiplayer modes, you can be a squirrel, fox, wolf, or bear.

How many characters do I control?

One.

What is the main focus?

The focus of the game is to survive in the forest as long as possible and collect enough nuts to survive each coming winter while avoiding predators, unless you are said predators, in which case your job is to devour squirrels..

What's different?

The procedurally generated territory and general controls and handling of a squirrel are a change for the survival exploration type games, which normally feature humanoid characters. Few games include generalized tree climbing as a primary focus. It's generally a zany, unusual blend of the simulation and action-survival genres.

Feature Set

General Features

Large, procedurally generated world.
A Day/Night cycle.
Neutral and Hostile AI agents by land and air.
3D graphics.
General tree-climbing and actual real-life squirrel functionality lifted straight from reality.

Multiplayer Features

Up to 8 players.
Easy to find a game.
Ability to play cooperatively with multiple squirrel players, or competitively with a combination of predators.

Gameplay

A fluid movement system that allows the player to take full advantage of a flying squirrel's ability to climb and glide off of trees. The squirrel will attach and run freely around on any tree's surface.

Physics-based movement, either while running or while jumping and gliding.

A set of rich ambient and hostile AI actors who will help or harm the player in interesting and challenging ways.

Changing weather and day/night cycles to keep the player on their toes.

A scent and track based system that forces the player to think carefully about how they move through the world.

The Game World

Overview

The game will take place in a deciduous, hilly forest world bounded by uncrossable cliff and mountain features.

Procedural Generation

Each time a new game is started, the player will be placed into a newly-generated world which follows a set of modifiable parameters to create a world of a general sparseness/density based on the player's preferences. A noise algorithm will first create a smooth landscape which will be bounded by impassible features and then filled with trees, brush, AI spawn points, the player's home, and other interactable features through several passes of another algorithm.

Full Access to the Y Axis

The player will be able to move not only along the ground, but also through the air and up/along the trees of the forest.

Environment Interaction

The player will be able to climb trees and pick up acorns and various power ups. The squirrel will also be able to glide, so that their descent from treetops to the ground is smooth and controlled.

The Physical World

Overview

The physical world will consist of variously arranged trees, rocks, bushes, hills and dales.

Key Locations

Key locations include the squirrel's nest (The players 'Home base') where collected nuts must be deposited and the various food spawn points scattered around the world. These spawn points will provide the necessary supply of nuts for the player to win.

Player Home

While the map will be procedurally generated with every time a player plays a game, the player's home (Their "Home Tree") will be centered in on the map and be situated in a familiar way.

Travel

Travel is achieved by running and jumping along the ground as well as by climbing up trees and along their branches. Players also have a limited gliding ability that allows them to jump off of tree and move between trees, given that they have enough initial altitude and that the target tree is near enough.

Scale

The scale of the game is based off the size of typical squirrel, meaning that trees and wolves will be several times more massive than the player, while nuts will be head-sized or smaller. The world will appear massive and take a quite decent amount of time to traverse.

Objects

The objects included in the world are a subset of things normally found in a forest, such as trees and bushes. Along with these objects, there are acorns to collect, and eerie otherworldly power-ups, floating in random spots in the forest and waiting to be collected.

Day and Night/Time

The game is a timed game, with the condition that the player must collect enough food before winter comes. Winter will arrive after *n* amount of days and each day night cycle will occur a variable time of 5-10 minutes.

Rendering System

Overview

The map will be procedurally generated and then built into the scene.

2D/3D Rendering

The Unity graphics engine will be used.

Camera

Overview

The camera will be variable between 3rd and 1st person (In certain situations), but the primary view will be 3rd person. This camera will follow the player and adjust it's position based on their current movement and surroundings.

Intelligent Following

The camera will move around the player based on obstructions and their current movement pattern. It will also be slightly controllable by the player to allow for better viewing of the environment.

View Variability

The camera will be able to be changed between first, 'over-the-shoulder' close third person , and far third person in certain situations such as when the player is hiding in a confined space.

Game Engine

Overview

The game engine used for this project will be Unity.

Scene-Based Organization

Unity works in terms of Scenes within a single project, so each of our atomic components (The game world, menus, credits, etc..) will be stored as different scenes.

Collision Detection

Unity provides 'Rigid Body' components along with various collider components that allow us to use the 'onCollisionEnter' and the 'onCollisionTrigger' methods to handle collisions between in-game components.

Lighting Models

Overview

We are using Unity's graphics engine to light our scene.

Day/Night Lighting

The Day/Night cycle is created using a single, powerful light that migrates around the terrain's plane over the course of 10 minutes. The skybox also fades from a day box to a night box slowly as the cycle runs.

The World Layout

Overview

The world will be laid out in a psuedo-random manner to keep each game new and fresh. The gameplay elements of the world will be laid out in such a manner so as to keep the world challenging.

Player Home

The player home is in the trunk of a unique tree and is placed in the center of the world. The generator will also ensure that the home is away from all of the enemy spawns either in a tree or at the base of a tree depending on the world's parameters.

Game Characters

Overview

The game will include a player character (Flying squirrel), several ambient, neutral AI players (Boars, Stags, and Rabbits), and several enemy AI players (Wolves, Bears, Eagles, and Foxes).

Creating a Character

The player will play a non-customizable flying squirrel in single player. This is created upon entering the game world. In Multiplayer, squirrels can be chosen with different stats. These squirrels are differentiated by their color (Red, Black, and Gray). Players in the Versus mode will be able to join the game as a predator as well and can choose between being a fox, wolf, or bear.

Enemy and Neutral Characters

Wolves - The primary ground predator. They are powerful and fast and seek out the squirrel quickly upon sight.

Foxes - A secondary, less aggressive ground predator than the Wolf. Still aggressive, but do less damage.

Eagle - The sole air-based enemy. Able to knock the player out of trees and kill them in flight. Ignores the player on the ground.

Bear - Slower than wolves, but very powerful and able to kill the player very quickly.

Rabbit - Eaten by all enemy AIs. Quick distractions for the AI predators to kill instead of the squirrel.

Stags - Neutral AI which will be attacked by bears and wolves instead of the player if seen at the same time.

Boar - Neutral AI which will be attacked by bears and wolves instead of the player if seen at the same time.

User Interface

Overview

The player can either play as the squirrel or a predator (In multiplayer) and the UI will changed based on their selection.

Squirrel Interface

The squirrel HUD features a “nut-counter”, a health bar, a time indicator, and a compass pointing the player back to their ‘burrow’.

Predator Interface

The predator HUD features a health bar and stamina, as well as the game timer and the amount of nuts collected by the squirrel.

Musical Scores and Sound Effects

Overview

Our game will utilize basic animal sounds and ambient, changing music.

3D Sound

Certain objects in the world, such as nuts, are 3d sound sources, but the game as a whole will not have an overall ‘3D’ design. Most sounds are 2d.

Sound Design

We will play custom made instrumental music on top of ambient forest sounds for situations where no enemies are chasing the character.

Single-Player Game

Overview

The single player game will involve the player being placed into a procedurally-generated world and then attempting to survive while searching out and collecting nuts in the environment. To win, the player must gather a certain amount of food (nuts) before a certain a timer runs out.. Once this time has elapsed, the player will either die because they did not gather enough nuts for the winter, or they will be given the option to quit as a winner or to continue on to play out the fall of the next year, which will be a harder version of the game the player just won.

Nut gathering

During each season a certain number of nuts will spawn upon the map. The player’s objective is to collect a specified amount and bring them back to their home to survive. The player will only be able to carry so two nuts on them at a time.

Enemy/Ambient AI Interaction

The player will have to avoid the various AI players in the world that are hostile to them. These hostile AI will hunt the player and certain other AI players based on their character.

Story

The story is simple. You are a squirrel. You either **survive** or **die**.

Hours of Gameplay

Since the game is procedurally generated, the hours of gameplay varies based on how long the game keeps your interest. A normal single ‘season’ of gameplay is 10 minutes maximum, and the player has the option of playing as many seasons as they want.

Victory Conditions

A player wins if, when the season timer expires (Winter comes), they are still alive and have collected the allotted amount of food to survive the winter. After winning, the player is given the option to ‘continue’, which places them into a new single player game with harder parameters, or to return to the main menu.

Multiplayer Game

Overview

The Squirrels vs Predators multiplayer game will mirror the single player in game type and world. However, you will be able to customize some basic game settings, such as time limit and nut goal.

Max Players

Both multiplayer modes will support between 1-8 players in the same game.

Servers

Our multiplayer architecture is client-server and will utilize Unity's master server technology.

Customization

Players will be given the chance to either play cooperatively as squirrels versus enemy AI players or as a mix of squirrels and predators in a competitive environment. In the competitive environment, the players taking the role of the predators will have a choice between which predator they wish to play as.

Internet

The game will utilize Unity's Master Server architecture for multiplayer.

Gaming Sites

Squirvival is hosted at http://www.cse.ohio-state.edu/~frecka/

Persistence

A new world is created each time a new game is created. This world does not persist unless the server is left running with a game active.

Saving and Loading

Games will not be able to be saved.

Stretch/Unimplemented Features.

- More detailed territories and homes for other animals
- More realistic simulation of ecosystems
- More biodiversity
- Weather effects
- Changing seasons
- Intelligently-Evolving difficulty
- Some kind of narrative mode with designed levels
- Fantastical miscellaneous abilities
- Water/Swimming/Drinking
- Hollow logs, burrows you can climb inside of
- Cinematic death animations
- Birds perching

Postmortem and Team Takeaways

Below you will find personal statements from the Squirvival Team. These were written at the end of the semester and should serve either as a guide to future teams (Should they ever read this document), or as a set of general comments about their time in the course.

Brandon's Takeaway

Making games in a team is both a fun and challenging experience and that would be one of the bigger things I learned, this being the first game I have ever officially made with a team (the first level of Mario doesn't count). Unity definitely makes work a lot easier for many steps of the game development process as well as provide some decent visuals to aid in things such as scaling and collision box checking. One really big thing I learned from this class is that while an anxious, idea producing team can make some pretty cool concepts, it can result in a bit of a lack of focus on some of the more basic concepts of a team. While I would like to try to work more on this game in concept, I almost want to start over, get the basics working and then trying to implement some of our current features better as well as adding those stretch goals.
That being said I will definitely continue making games hopefully professionally one day but for sure as a hobby.

Cameron's Takeaway

My dream since I was young was to make video games! This is the first time I was able to come close to making a real life video game that was interesting. I did make a video game for the Android platform but that wasn't as great as this capstone project. It has been a pleasure working with my teammates and I think this semester turned out well. I enjoyed Dr. Crawfis and his teaching style. Working with Unity gives you such an advantage on developing games. I think this capstone has put me one step closer to my dreams! I plan on continuing game development as a track and hopefully it'll all work out.

Dan's Takeaway

Game design is not the easiest thing in the world. While the ideas may be aplenty, narrowing them down and deciding on the best one is not an obvious endeavor. In such a limited amount of time it is never going to be possible to implement all of your brilliant designs. In addition, there needs to be a solitary, concrete scheme behind the game, or the lack of unitary cohesion between members will cause the game to suffer. What I mean by this is that all team members should be on the same page in their vision of what they want to create. Otherwise, the final product may not please everyone. Inevitably, this class is about learning the process behind making games. In this regard, I think we succeeded greatly. Regardless, I am happy with the final product, hopefully marking the beginning of an exciting career in video games.

Jared's Takeaway

Team meetings always went rather well. In most meetings the action items got completed in a timely manner but our organization could have been much better. Sometimes it was difficult to organize six different people doing six different actions. We then get into the next topic of communication and coordination. Overall the group chemistry was very good. We went through team meetings smoothly and did not have any significant problems. All group discussions ran smoothly and all team members were respectful to other team members. We successfully used the website GitHub, which allowed us to use many innovative and helpful features like packaging and creating files directly. This tool was a valuable resource to our team.

Tom's Takeaway

It seems to me that Unity as an engine is very good for enabling people from multiple disciplines to work together on a product, so it was interesting to be on a team where everyone was a programmer. It was difficult to work with different coding styles and to keep everyone's vision of the game uniform. I think if I had another chance to do this project I would want to ensure that the design of the overall game was simple and lucid from the beginning of development.

Stephen's Takeaway

This project was a very difficult one and it posed many new technical and social challenges that have helped me grow a lot as a programmer and as a team member over the course of the semester. As a first time scrum master, I was able to experience the great difficulty that comes with attempting to organize other technically-oriented people around a single project and learned a lot about what needs to be done (And what shouldn't be done!) to keep people focused and busy. Out of these lessons, I think the three most important center around project planning, teamwork, and team pride.

Although I have come to abhor the extreme amount of documentation and inane, repetitive graph making that comes with traditional 'enterprise' programming, I realize now that it can be invaluable to spend the time to lay out an architecture before anyone begins coding. Our team ran into a good deal of trouble when attempting to implement and maintain our multiplayer code because our initial base was not built with this goal in mind and continued to change in ways that made multiplayer unnecessarily buggy and painful.

In addition to more structured planning, I believe that finding a way for team members to work together in a common location would have been very valuable. Because our team was very spread out and quite busy during the semester, we had to work individually for most of the time and this lead to many awkward merges and a generally fragmented codebase. Pair programming, or at least programming in the same room, would have allowed more dialog between team members that could have helped keep us all on the same page. Implementing a team style guide and having code reviews would have also been very helpful with this problem.

Finally, I believe that a good team leader should strive to ensure that each member of their team becomes a 'stakeholder' in the project in a way that they take pride in what the team is producing and feel some sort of personal ownership of the implementation and ideas behind the game. During the semester, I tried hard to make sure that our team (which is comprised almost entirely of 'big idea' types) didn't commit to too much work because I was afraid of having us bite off more than we could chew. A side effect of this attitude, unfortunately, was that a lot of ideas got cut and our team ended up not having a well defined game to strive for until the very end of the semester. This caused working on the game to become tedious and uninteresting at times. It also made it hard to have enough work to assign team members, causing the workload to be uneven or overly difficult for members who had to keep jumping between systems that they had never worked on.