Netty Pokes

Corinne O'Flaherty
Jake Ruth
Logan Seeley
Dustin Yost
https://pineapple.champlain.edu/projects/fa17-egp405-networking/repository?utf8=%E2%9C%93&rev=feature%2Ffinal

Project Plan

Concept

Netty Pokes is a simple 2D-2.5D top down game based on 1990-2000's Pokemon games. Players will be able to battle each other and NPC wild pokemon, while exploring a world.

Purpose and Requirements

The purpose of our final project, Cretins/ChampNet, is to create our own custom networking framework that we can use in Unity Engine to create a multiplayer game. Using .dll files we can create a plugin for Unity that will allow the C# scripts to communicate with our network messaging code in C++.

We also plan on creating a standalone 'server' application that will manage the connections of the players in the game as well as update the status of the server and players.

The game itself will be a multiplayer Pokemon-inspired game that will utilize an open-world exploration map, as well as allow players to battle other players or NPCs in a turn-based battle system.

Mechanics/Features

Top Down 2D Movement

A simple free-movement system in 2D space with appropriate collisions and sprite orientations. Note: To make things simpler, players and NPCs will not collide with each other, just terrain.

Multiplayer / Server + Client

We will be using the server - client model, where the server is not a client, but it's own separate entity. The clients will connect to the game and be able to send their data to

the server, which will be broadcast appropriately to all the other clients. The server will also be able to handle multiple asynchronous battles going on at once.

Random NPC Spawning

Throughout the game world there will be many, randomly spawned, NPCs that can be battled with. We will implement a fairly simple system to populate these NPCs.

Battles

Upon entering combat, there will be a transition screen and the players will be brought to a different battle interface. They will proceed to take turns using attacks until one wins, at which point the players both return to the world-space.

Interfaces

The game will have a simple user interface that will show the status of his/her 'pets', as well as any other general information needed. We will also implement a keybind to pull up a map that will detail the area as well as show the locations of the other players.

Systems/Frameworks

RakNet

RakNet is a C++ middleware library used to handle networked connections between computers. In utilizing this library, programmers are able to write an interface to more easily interact with Unity. Unity inherently utilizes RakNet, however writing a new interface will allow for programmers to more easily handle modular control over sending data in a network connection.

UnIput

UnIput is a Unity C# driven framework designed to handle multiple input formats at once. It handles the input of keyboard, mouse, xbox controller, PS4 controller, or Switch JoyCon. By using configuration files, it is able to map button, axes, and triggers into a single format. This format is broadcasted to UnityEvents, allowing them to be exposed to scripts within the Unity inspector. This enables designers to easily map an action or function to multiple control inputs at once.

Topics

Networked Architecture

Our architecture consists so far of a server - client model, where the server is not a client, but it's own separate entity (See: Multiplayer / Server + Client Section).

Asynchronicity

The game will feature asynchronous messaging during the battle sequences. After the transition from open-world to client-client(PvP) or client-server(PvE) battle types, the game will use asynchronous messaging to handle player commands.

Synchronicity

Synchronous messaging will be used during the majority of the gameplay experience, when the player and other characters are present in the open world environment.

Improving UNet

Unity's networking is robust and we do not have the capabilities to improve on Unet, rather we intend to make a framework in C++ that gives us significantly more control over the networked aspect of the game. By doing this, we hope to make a simpler and faster implementation of networking than what Unity provides.

Milestone Schedule

October 5th, 2017:
- Proposal

October 12th, 2017:
- Open-world
    - Movement mechanics
    - Collisions with world
- Main Menu Designed
    - Play button functionality
    - Connect button exists (does nothing)
    - Exit button works

October 19th, 2017:
- Transitions to battle
    - Mock battle state/ **scene**
- Open-world designed/ mapped out
    - Key placement
    - areas blocked out in engine
- Network Framework

        - Server (**console application**)
        - Dll available for the game

October 26th, 2017:

- Main Menu Completed
    - Connect button functionality
        - Sub menus
- Battles Fully functional
    - Works on its own, not integrated in overworld

November 2nd, 2017:

- Battles integrated in openworld
    - Battles synced with online play
- Random monster Spawning

November 9th, 2017:

- Pause Menu completed

November 16th, 2017:

- All art implemented

November 23th, 2017:

- Fix bugs
- Add stretch goals

November 30th:

- Final Project version
- Prep for the presentation

December 4th-8th:

- Presentation