A

Networking DLL

George Gularte

Whats To Come

- **Overview**
- **MileStones**
- **Project Simulation**
- **Website Portfolio**

DLL Networking Project

- Gross Simplification
- I'm a messenger
- I just move data from one source to another
- User Input ->Unity C# -> C++ -> Another Computer

What's the Purpose

To Create Middleware

unity
asset store

Benefit

To create a simpler and faster Networking alternative.

My networking can be **tailored** to fit Users specific needs i. e; *Ease, Security, Speed, Scalability etc.*

Why I Chose this?

I want to specialize in network gameplay programming.

Few People I Follow

- Pablo Cheng, Senior Network Engineer at Double Helix
- Glenn Fielder, Senior Network Engineer at Respawn
- Yahn W. Brenier, Senior Software Engineer at Valve

Milestone Markers

1st   Design Plan

2nd UDP virtual connection

3rd   DLL Creation

4th   C# Unity

- Importing
- Wrapper
- Debugging Test

5. Peer Side prediction

Design Plan

Network Architecture

How?

- UDP/IP using Winsock library

Why?

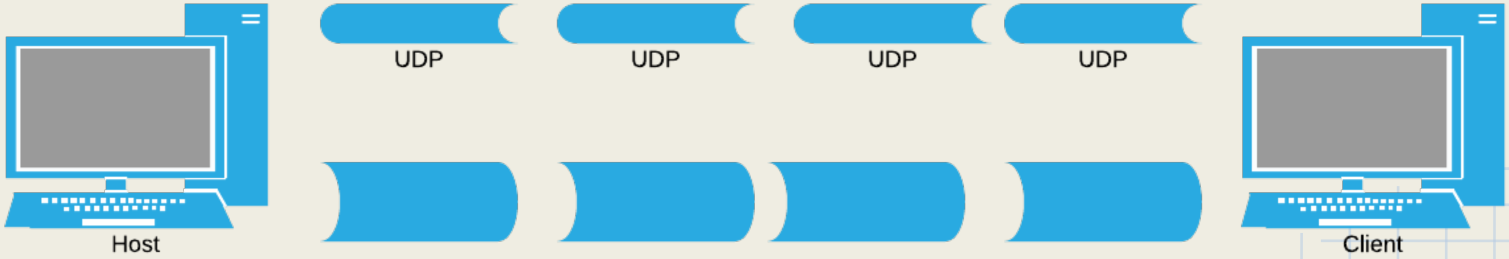
- Speed & Control(UDP is the preferred choice to fast action games)

What?

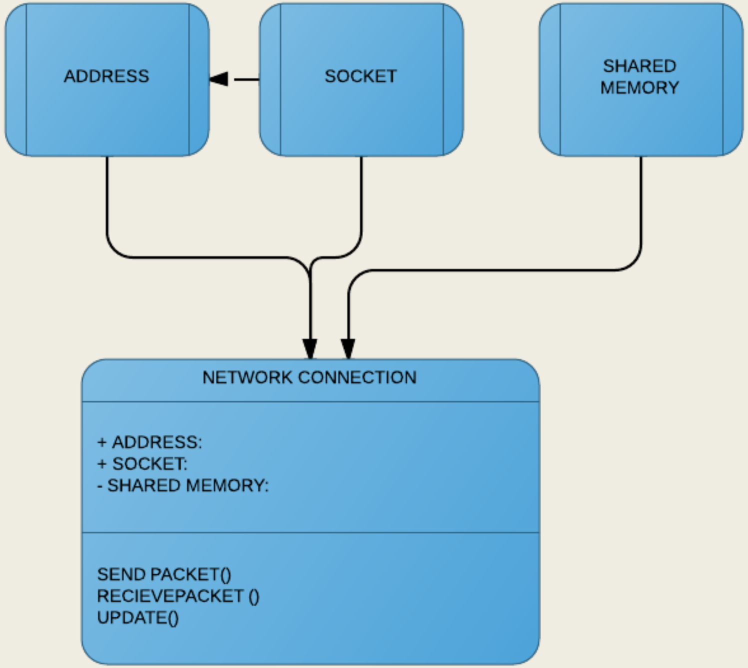
- Send and Receive
- Establish a Stable State of ASynchronized Serialized data
- Other Components Needed Thread Library and Fstream (*.txt)

UDP Virtual Connection

- Create a stable connection between 2 computers
- Asynchronously sending and receiving data



UML: Network Code



ADDRESS

SOCKET

SHARED MEMORY

NETWORK CONNECTION

+ ADDRESS:
+ SOCKET:
- SHARED MEMORY:

SEND PACKET()
RECIEVEPACKET ()
UPDATE()

DLL Pre Prep

UnitTest

- Test if Clients can connect to the server
- Check When Clients have disconnected
- Check To see if Clients and Server can Send data without waiting to receive data - Asynchronous

String Parsing

- “10.229.20.147”
- 182785179
- 0x0ae5149b

Thread Test

- Need To Launch Threads Independently from the Main Thread
- Threads need to be constantly running
- Threads are NOT going to be rejoined

Logging

- Writing data results to a Text File

DLL

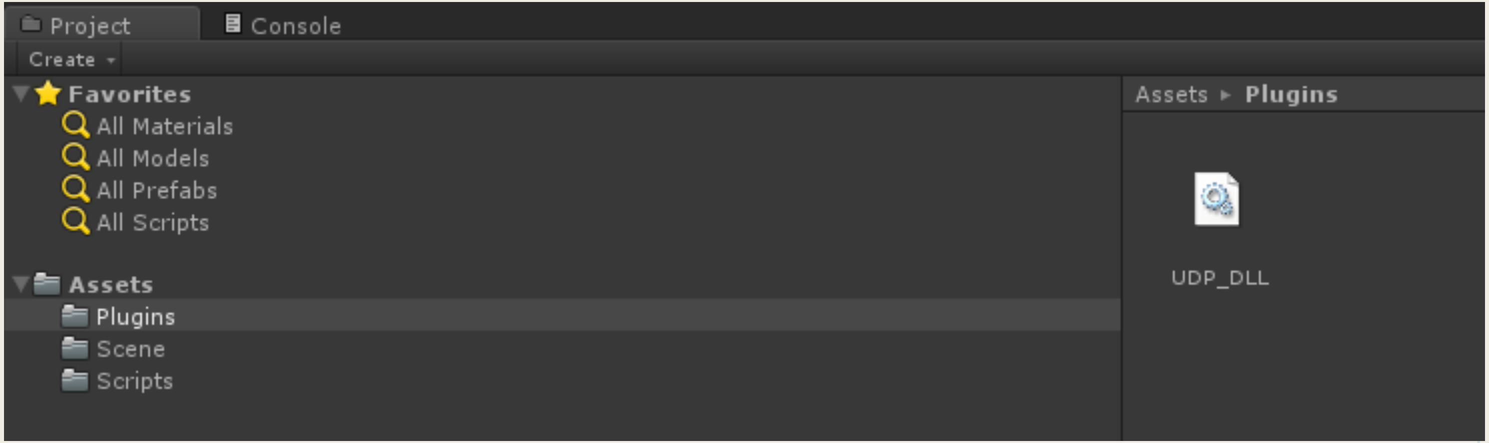
4 Public Functions

- Create Server
- Create Client
- Send
- Recieve

```
extern "C"

__declspec(dllexport) int CreateClient(const int ClientPort, unsigned char
ServerIP[], const int serverPort )

__declspec(dllexport) bool SendPacket(void *arg, int size)
```

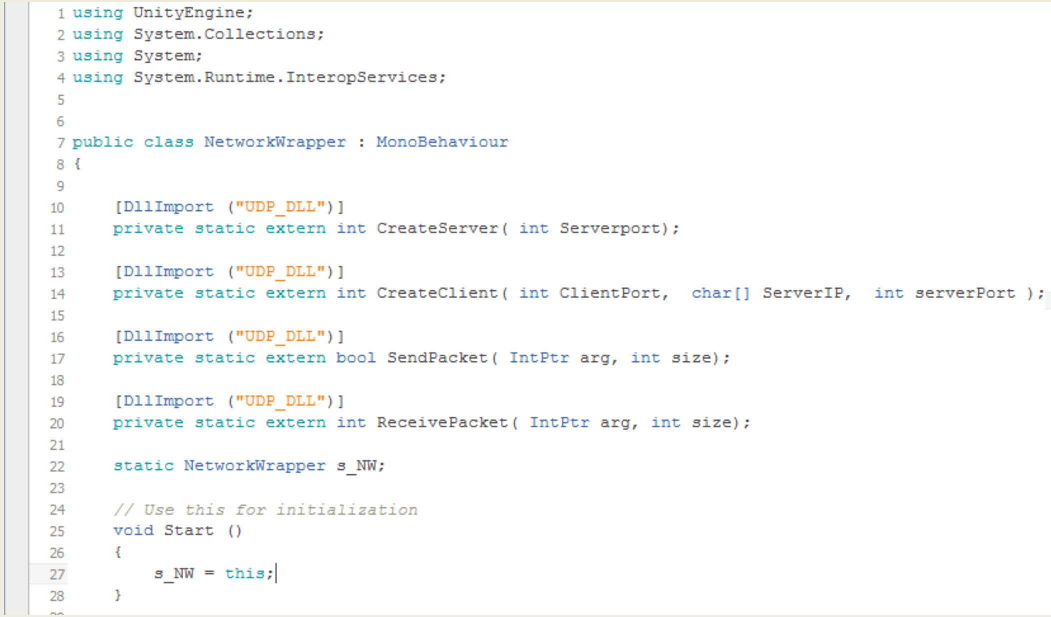
Unity: Getting it Started

Place the DLL in the Plugins Folder



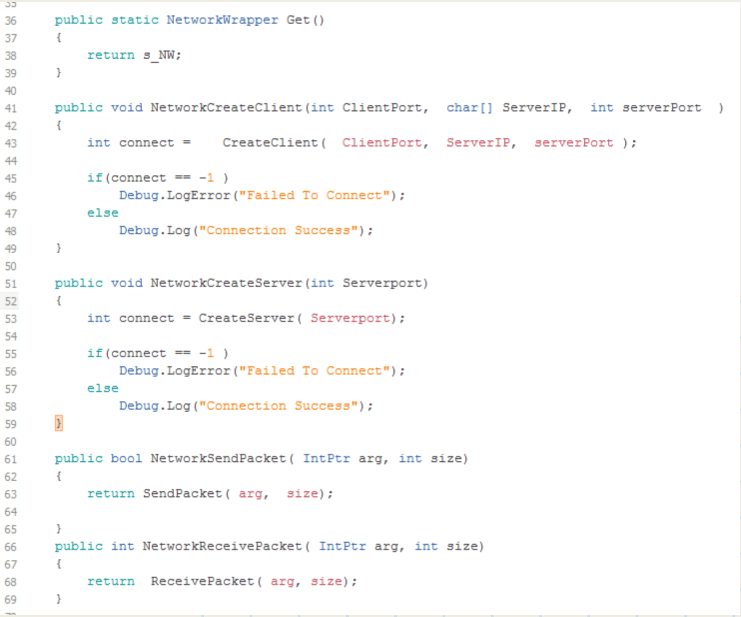
Unity: C#

My Network wrapper IMPORT



```
using UnityEngine;
using System.Collections;
using System;
using System.Runtime.InteropServices;


public class NetworkWrapper : MonoBehaviour
{

    [DllImport ("UDP_DLL")]
    private static extern int CreateServer( int Serverport);

    [DllImport ("UDP_DLL")]
    private static extern int CreateClient( int ClientPort,   char[] ServerIP,   int serverPort );

    [DllImport ("UDP_DLL")]
    private static extern bool SendPacket( IntPtr arg, int size);

    [DllImport ("UDP_DLL")]
    private static extern int ReceivePacket( IntPtr arg, int size);

    static NetworkWrapper s_NW;

    // Use this for initialization
    void Start ()
    {
        s_NW = this;
    }
```

Users Call Functions



```
public static NetworkWrapper Get()
{
    return s_NW;
}

public void NetworkCreateClient(int ClientPort,   char[] ServerIP,   int serverPort   )
{
    int connect =    CreateClient(  ClientPort,   ServerIP,   serverPort );

    if(connect == -1 )
        Debug.LogError("Failed To Connect");
    else
        Debug.Log("Connection Success");
}

public void NetworkCreateServer(int Serverport)
{
    int connect = CreateServer( Serverport);

    if(connect == -1 )
        Debug.LogError("Failed To Connect");
    else
        Debug.Log("Connection Success");
}

public bool NetworkSendPacket( IntPtr arg, int size)
{
    return SendPacket( arg,  size);

}
public int NetworkReceivePacket( IntPtr arg, int size)
{
    return  ReceivePacket( arg, size);
}
```

Unity DLL Debugging

OMG OMG OMG

Its a challenge, the debugger is not natively set to work outside of Monodevelop.

Debugging Process

- Crash
- Check if it was a based on Unitys side
- Check if it was logic error in the DLL Side
- logged out data to text.file read it
  - Change things,
- Repeat and pray

Crash Prep

Computer Crashes Happened

Precautions if a freeze were to happen

- Time Out Functions in UNITY

```
if (timeOut > 20000)
 {
            #if UNITY_EDITOR
             UnityEditor.EditorApplication.isPlaying = false;
             #else
             Application.Quit ();
             #endif
 }
```

- Time Out Functions in C++
  - Assert, and Close.Sockets
- GUI Button to exit Unity

Most of my time was Debugging

Beginning of the End

The Check List

<u>Step 1</u>

[✚] Create Connection between 2 Computer in C++

    [✚] Send and Receive Data

<u>Step 2</u>

[✚] Import a DLL into Unity

<u>Step 3</u>

[✚] Establish a Server and Client a connection in Unity

➡ [ ] Have the User Send and Receiving data using the DLL Call Functions

[ ] Unity Client Side predictions

The Problems I encountered

There were Many Obstacles in this project

- Infinite loops
- Threads closing aborting on their own
- Receiving Junk Data
- Setting up debugging environment on 2 machines,

However there were 2 Problems that consumed the most time, in not completing the primary objective in the given time.

The Data being passed from Unity to the DLL

- Servers IP address from Unity
- Send Function

Char

2 problems same location

No Null Terminator

```
string serverIP = "10.229.20.156";
```

Soultion "10.229.20.156\0"

Char is 2 Bytes in Unity not 1

```
public void NetworkCreateClient(int ClientPort,  char[] ServerIP,  int serverPort )
                    CreateClient(  ClientPort,  ServerIP,  serverPort );
```

Solution:

```
          byte[] IpByte = new byte[sizeOfServer];
for(int i = 0 ; i < sizeOfServer; ++i  )
           IpByte[i] = (byte)ServerIP[i];
CreateClient(  ClientPort,  IpByte,  serverPort );
```

Send Function

This is crash

C#
public bool NetworkSendPacket(IntPtr arg, int size)

C++
*__declspec(dllexport) bool SendPacket(void *arg, int size)*

Doesn't enter the function System Crash at the very call

Lessons

Patience

Build a debugging environment

Know More about data Conversions between 1 source and another