Assignment 1: Simplest Client-Server

1. Get Set Up

~~If you are not already using gitlab or github, you are now.~~

~~Create two Git repos, for your Server and Client projects. Name them TicTacToeClient and TicTacToeServer since tic tac toe is the final boss in this course.~~

~~Using Git Desktop (or cmd line), clone your repos so that you have access to two project folders being managed by Git.~~

~~Create a Unity project in both of your git repo folders.~~

~~In both projects, go to Window->Package Manager, in “Packages:Unity Registry” search for “Unity Transport” and install.~~

Get the simplest server and client components from:

https://gitlab.com/netcode-in-games-campaign-manual/simplestclient/-/blob/main/Assets/NetworkClient.cs

https://gitlab.com/netcode-in-games-campaign-manual/simplestserver/-/blob/main/Assets/NetworkServer.cs

~~Add the NetworkClient script to your client project. Add the NetworkServer into your server project.~~

~~In each project, create a gameobject and attach the appropriate network script to it.~~

~~Push an initial commit on both of your projects.~~

Link me as a collaborator on both of your projects.

https://gitlab.com/FernandoTeaches

https://github.com/WindJesterFernando

2. Initial Demonstration

~~Open NetworkClient.cs, find the string constant named IPAdrees and then edit it to your local IP Address. To discover your local IP, on Windows, use ipconfig in the windows terminal or, if you are on Mac, via your apple icon -> Sys Pref -> Network menu options.~~

~~Open your server program and run it. Now run your client program. Does it connect (and stay connected)? When you focus on your client program and hit the 'a' key, do you get a hello from your client in your server debug log?~~

~~Is there a delay in your debug log msgs? Do they not show up until you select the project? By default unity does not run in the background. To change this, go to Project Settings -> Player -> Resolution and Presentation and then toggle on the Run in Background option.~~

3. The "out" Parameter Modifier

What is the "out" keyword parameter modifier in C#? What does it do?
Why do we need to use the out keyword parameter mod with int values but not when we pass class instances? The "out" keyword requires the method to modify the parameter before it returns. Class instances don't need to be tagged with this keyword because they themselves are reference types.

4. Reading Code

The cornerstone of this course is the code contained with NetworkClient.cs and NetworkServer.cs. You are being challenged not just to understand the gist of it, but instead, seek to understand every line of it.

Reading code is a skill to get good at in and of itself. As you may have intuitively deduced, contrary to the way we read a book, the process of reading code is not as linear. Thus the question opens, how does one best read through code? Consider the following approach which is optimized to reduce cognitive overload, stress.

1. Scan through the code, look for any keywords or syntax that you do not understand. These are easily isolated and can initially be figured out. Answer the following: what keywords/syntax are not immediately known to you and (after some research) what do they do?
2. Scan through the code and look for external classes, structures, enumerations and function calls. Read the documentation on them. This should take a while. Answer for each of the following:
    2.1. What is the NativeList struct? Similar to NativeArray, it is a dynamic and resizable list in native memory.

2.2. What is the NetworkDriver struct? Handles the sending and receiving of data through a connection.

2.3. What is the NetworkConnection struct? The connection between two endpoints.

2.4. What is the NetworkPipeline struct? A sequence of pipeline stages where data is processed before sending and after receiving.

2.5. What is the FragmentationPipelineStage struct? The pipeline stage fragments large packets, ensuring it fits within the network's maximum packet size.

2.6. What is the ReliableSequencedPipelineStage struct? The pipeline stage that ensures packets are delivered reliably and in the correct order.

2.7. What is the NetworkDriver's CreatePipeline() function? Creates a NetworkPipeline.

2.8. What is the NetworkEndPoint struct? Represents the address for network communication.

2.9. What is the NetworkEndPoint.Parse() function? It parses the IP address and network port to create a NetworkEndpoint.

2.10. What is the NetworkDriver's ScheduleUpdate() function? This schedules a network update job that processes network events.

2.11. What is the NetworkDriver's ScheduleUpdate().Complete() function? This completes the scheduled network update job.

2.12. What is the NetworkEvent.Type enum? This defines the type of network events that can occur during network communication (Empty, Data, Connect, Disconnect).

2.13. What is the NetworkConnection's PopEvent() function? This retrieves the next available network event.

3. Learn the name of each member variable and function.

4. Read through the code, more or less, linearly. Seek to read each line and understand what it does.

5. Read through the code again and generate a list of any unanswered questions you have.

Note; in some cases, an effective description of a thing can be produced by restating the name of the thing, when this is the case, it is fine to do, but if there is more to say, seek to say it.

Congratulations, transporting data across the internet is a thing you know how to do!