Assignment 1 – Simplest Client-Server

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?

NetworkDriver – Network drivers serve as an essential bridge between your computers operating system and its network hardware.

ReliableAndInOrderPipeline/nonReliableNotInOrderedPipeline – defines a network pipline that ensures reliable and in-order delivery/ defines a network pipeline for unreliable and unordered data transmission.

Allocator.Persistent – specifies that a memory allocation should persist beyond the current frame. Used for allocating memory in native collections.

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?

Used to manage and store client connections in an efficient and performance-oriented way. Makes sure that the server can handle many clients without performance degradation due to garbage collection. This uses memory, so it needs to be disposed of manually.

2.2. What is the NetworkDriver struct?

Used to create and bind to a Network Endpoint. This is responsible for managing communication between the server and the clients over the network. Essentially a bridge between the computers operating system and its network hardware.

2.3. What is the NetworkConnection struct?

Used to manage the state and data flow of a Network connection between a client and the server. It's responsible for identifying connection, sending and receiving data, tracking connection status, and handling disconnections. It's central to managing individual connections in Unity's low-level networking system. Uses NetworkDriver.

2.4. What is the NetworkPipeline struct?

NetworkPipeline represents a set of processing stages for data sent over the network. Defines how data is handled between server and clients (reliability, ordering, fragmentation) by using different types for network communication. Reliable and ordered / unreliable and unordered.

2.5. What is the FragmentationPipelineStage struct?

Breaks down data into smaller, manageable pieces called fragments. Necessary because networks often have a maximum transmission unit (MTU) size for packets, meaning that if you try to send data that

exceeds the MTU It will be either dropped or result in network errors. Makes sure the network communication can handle large data sizes efficiently and transparently.

2.6. What is the ReliableSequencedPipelineStage struct?

Ensures that network communication is both reliable and ordered. It provides guarantees that data packets are transmitted if they are lost during transmission (reliability), and that packets will be received in the same order in which they were sent ensuring the integrity of the message order (sequencing).

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

This function constructs a pipeline by taking in types of pipeline stages as arguments. These stages determine the behaviour of the pipeline during data transmission. Each stage in the pipeline is applied to the data in sequence. This returns a NetworkPipeline which can be used in other methods. You can create one for reliableInOrderPipeline/nonreliableNotinOrderedPipeline.

2.8. What is the NetworkEndPoint struct?

Represents the network endpoint (IP address and port) used for network communication. Designed to define where data should be sent or received from that location. Especially useful when binding to a local address (server-side) or connecting to a remote address.

2.9. What is the NetworkEndPoint.Parse() function?

This function is used for dynamically creating network endpoints from strings, such as those obtained from user input, configuration files, or other external sources. Automates and simplifies the process of interpreting and converting network addresses into a useable format for network communication.

2.10. What is the NetworkDriver's ScheduleUpdate() function?

Used to schedule a background update task for handling network communication. It helps manage low-level network operations asynchronously, which is important for maintaining performance in real0time applications like games or networked simulations. This way the server doesn't get blocked while it waits for network I/O. Keeps main thread free to handle other tasks like processing input, updating game logic or rendering frames.

2.11. What is the NetworkDriver's ScheduleUpdate().Complete() function?

The Complete() function is called after ScheduleUpdate(). Used to synchronize the task ensuring that the scheduled network operations are completed before the program moves onto the next task. Ensures that all of the network processing tasks initiated by ScheduleUpdate() are finished during the current frame. Enables efficient and responsive network communication in a real-time multiplayer application.

2.12. What is the NetworkEvent.Type enum?

Used to represent the different types of events that can occur using network communication. It's important because it helps identify the type of network event that the server, or client, should handle such as receiving data, client disconnections and other network-related events.

2.13. What is the NetworkConnection’s PopEvent() function?

Used to retrieve and process network events from a particular connection. Is a key part of handling incoming data and network events asynchronously. Helps managing communication over a network by notifying the application of events like incoming data or connection changes.