【Unity基础教程】重点知识汇总

（一）

TCP、UDP和Socket的含义区别及应用

视频链接：https://www.bilibili.com/video/BV1ppBCYhEyC?vd_source=90c7d8485752ee5a35ccafa7923a69bf

TCP 传输控制协议（Transmission Control Protocol）

定义：它是一种**面向连接**的协议。确保数据能够**可靠有序**的传输到目标设备。它就像打电话一样，在通话（数据传输）之前，需要先**建立连接**，并且在整个通话过程中会**保证信息的完整性**。

特点：

- **可靠性。**它会保证数据的完整性和顺序。如果数据在传输过程中丢失或损坏。TCP会负责重新传输。
- **面向连接。**TCP在通信前需要建立连接**（三次握手）**并在通信结束时关闭连接**（四次挥手）**。
- **有序性。**通过序列号，TCP会确保数据按正确的顺序到达。
- **慢速但稳定。**由于需要处理重传、确认等机制，因此TCP的性能比UDP慢，但适合需要高可靠性的场景，比如聊天程序、游戏中的账号信息、存档数据的上传和下载等。

TCP三次握手（含义及解释）

三次握手：是指TCP在建立连接时，**客户端**与**服务器**之间交换**三个包**的过程。目的是**确保双方都能够正常发送和接收数据。**

第一次握手：由客户端向服务器发送一个带有**SYN**标志的数据包，表示请求连接，并告诉服务器自己将要使用的序列号。

第二次握手：服务器收到**SYN**包后，给客户端回应一个带有**SYN**和**ACK**标志的数据包，表示同意建立连接，同时也告诉客户端自己的序列号。

第三次握手：客户端收到**SYN**和**ACK**包后，发送一个带有**ACK**标志的数据包给服务器，表示确认建立连接。之后服务器收到**ACK**包后正式建立连接。

TCP三次握手（过程演示）

第一次握手：**客户端** —— 发送SYN包 请求连接 ——→ **服务器**

第二次握手：**客户端** ←—— 发送SYN+ACK包 同意连接 —— **服务器**

第三次握手：**客户端** —— 发送ACK包 确认连接 ——→ **服务器**

（请求连接 → 同意连接 → 确认连接）

TCP四次挥手（含义及解释）

四次挥手：是指TCP断开连接时，**客户端**与**服务器**之间交换**四个包**的过程，目的是**保证双方都能够安全的断开连接。**

第一次挥手：客户端向服务器发送一个带有**FIN**标志的数据包，表示自己已经没有要发送的数据了，准备断开连接。

第二次挥手：服务器收到**FIN**包后，回复一个带有**ACK**标志的包给客户端，表示知道了。

第三次挥手：服务器在确认没有要发送的数据后，给客户端发送一个**FIN**包，表示服务器也准备断开。

第四次挥手：客户端收到**FIN**包后给服务器回复一个**ACK**包，表示确认断开连接。这时客户端进入等待状态，等待一段时间后彻底断开连接（防止服务器未收到**ACK**包后而重传**FIN**包）。服务器收到**ACK**包后立即断开连接。

TCP四次挥手（过程演示）

第一次挥手：**客户端** —— 发送FIN包 请求断开 ——→ **服务器**

第二次挥手：**客户端** ←—— 发送ACK包 同意断开 —— **服务器**

第三次挥手：**客户端** ←—— 发送FIN包 准备断开 —— **服务器**

第四次挥手：**客户端** —— 发送ACK包 确认断开 ——→ **服务器**

（请求断开 → 同意断开 → 准备断开 → 确认断开）

UDP 用户数据报协议（User Datagram Protocol）

定义：它是一种**无连接**的，**不可靠**的传输协议。**它不保证数据的可靠，也不保证传输的顺序。**可以把它想象成**寄明信片**，把明信片寄出去后，不能确定对方是否能收到，也不知道明信片是否会按你发送的顺序到达。

特点：

- **无连接。**UDP不需要建立连接，直接发送数据**（没有三次握手和四次挥手）**。
- **不可靠。**UDP不保证数据一定到达，也不保证顺序。如果传输过程中数据丢失或损坏，**UDP不会重传。**
- **高效性。**UDP没有复杂的握手和确认机制，因此传输效率高，适合**实时传输**的场景。比如在线游戏的实时位置同步、音视频数据传输或需要**低延迟**的应用。
- **轻量性。**UDP的头部信息非常少，**数据开销小**。

TCP与UDP的区别（对比）

TCP 和 UDP 的对比

特性	TCP	UDP
是否可靠	是（有重传、确认机制）	否
是否有序	是	否
连接方式	面向连接	无连接
速度	慢（因为有额外的握手、确认等开销）	快（无握手、确认机制）
适用场景	需要高可靠性的数据通信（如文件传输）	需要实时性的数据传输（如游戏）

Socket（含义及解释）

它是**网络通信**的基础，是操作系统提供的一种机制。用于在网络上两个程序之间进行数据传输。可以把它理解为通信的**桥梁**。它屏蔽了底层复杂的网络协议细节。在编程中，它是**实际用于发送和接收数据的对象**。在Unity中可以用它实现实时的多人游戏、聊天系统和远程控制等功能。

Socket工作流程表

Socket工作流程表

角色	步骤	描述
服务器端	1. 创建Socket	创建一个Socket实例，为通信提供接口。
	2. 绑定（Bind）	将Socket绑定到指定的IP地址和端口号。
	3. 监听（Listen）	开始监听客户端连接请求，进入等待状态。
	4. 接受（Accept）	接受客户端连接请求并建立通信通道。
	5. 数据交互	通过Socket与客户端交换数据（如发送或接收消息）。
客户端	1. 创建Socket	创建一个Socket实例，为通信提供接口。
	2. 连接（Connect）	连接到服务器的指定IP地址和端口号。
	3. 数据交互	通过Socket与服务器交换数据（如发送或接收消息）。

课程总结（TCP、UDP与Socket）

TCP、UDP和Socket它们都是与计算机，网络通信密切相关的基本技术和概念。TCP和UDP是两种**截然不同**的数据传输协议，而Socket是按照协议具体传输数据的一个**工具**。

【Unity基础教程】重点知识汇总

（一）

TCP、UDP和Socket的含义区别及应用

视频链接：https://www.bilibili.com/video/BV1ppBCYhEyC?vd_source=90c7d8485752ee5a35ccafa7923a69bf