以下是对几种常见设计模式的介绍，包括它们的概念以及可以解决的问题：

一、单例模式

概念：

单例模式确保一个类只有一个实例，并提供一个全局访问点。

场景：

数据库连接池管理：在一个应用程序中，为了避免频繁地创建和关闭数据库连接，可以使用单例模式来管理数据库连接池。例如，使用一个单例的 ConnectionPool 类，确保整个应用程序中只有一个连接池实例，多个模块可以通过全局访问点获取数据库连接，提高性能和资源利用率。

二、模板模式

概念：

模板模式定义了一个操作中的算法骨架，将一些步骤延迟到子类中实现，允许子类在不改变算法结构的情况下重定义某些具体步骤。

数据导出功能：假设一个应用程序支持多种数据导出格式，如 CSV、Excel 和 PDF。可以使用模板模式定义导出的基本流程，将不同格式的导出细节留给子类实现。三、代理模式

概念：

代理模式为其他对象提供一种代理以控制对该对象的访问。

解决的问题：

访问控制：可以在不修改原对象的基础上，对对象的访问进行控制，如权限控制、延迟加载、远程调用等。例如，在访问敏感资源前检查用户权限，或在实际使用对象前先进行一些准备工作。

增强功能：可以在不修改原对象的情况下，为其添加额外的功能，如添加日志记录、性能统计、异常处理等。

远程调用：在分布式系统中，代理模式可用于实现远程对象的本地代表，使客户端可以像调用本地对象一样调用远程对象。

四、工厂模式

概念：

工厂模式提供一个创建对象的接口，让子类决定实例化哪个类，将对象的创建和使用分离。

在一个日志系统中，可以使用工厂模式创建不同类型的日志记录器，如文件日志记录器、控制台日志记录器等。根据配置或用户需求，可以灵活地创建所需的日志记录器。

五、责任链模式

概念：

责任链模式将多个对象组成一条链，请求在这条链上传递，直到有一个对象处理该请求为止。

场景：

员工请假审批流程：在一个公司的请假审批系统中，不同天数的请假需要不同级别的领导审批，可以使用责任链模式。员工的请假请求从基层领导开始，根据请假天数，请求会被传递给更高级别的领导，直到被批准或拒绝。

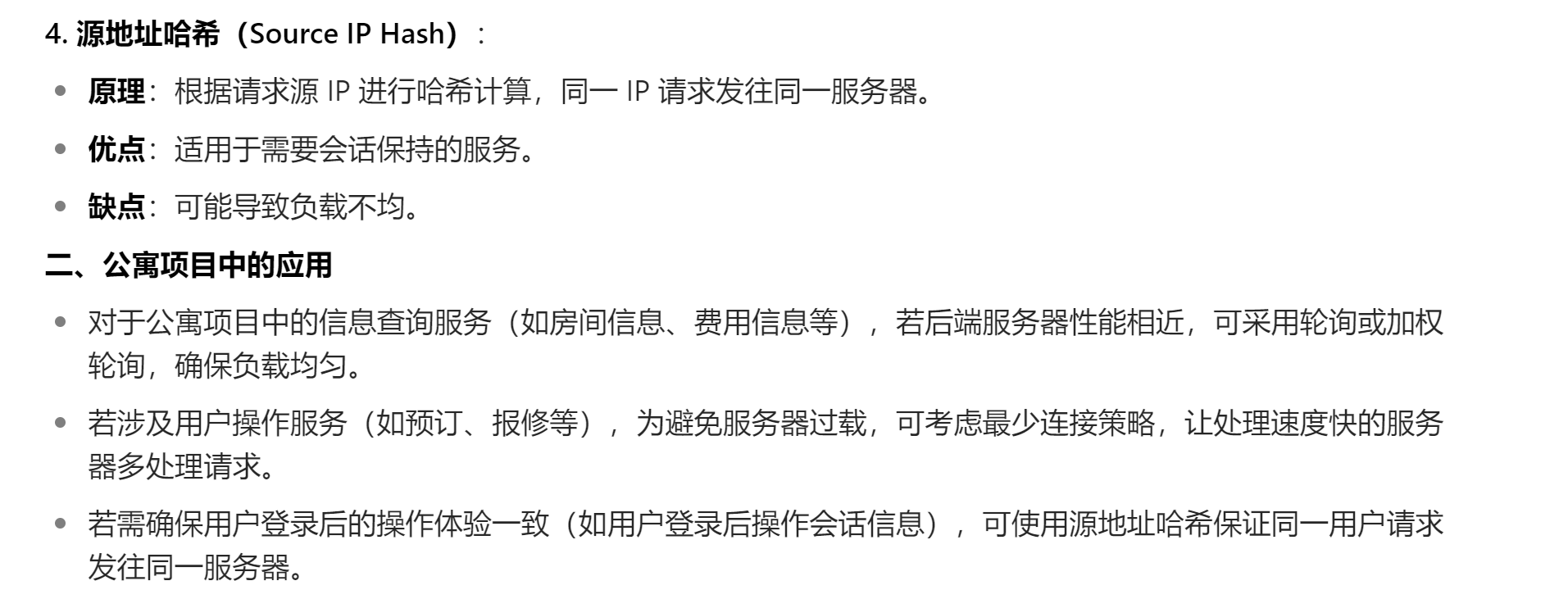
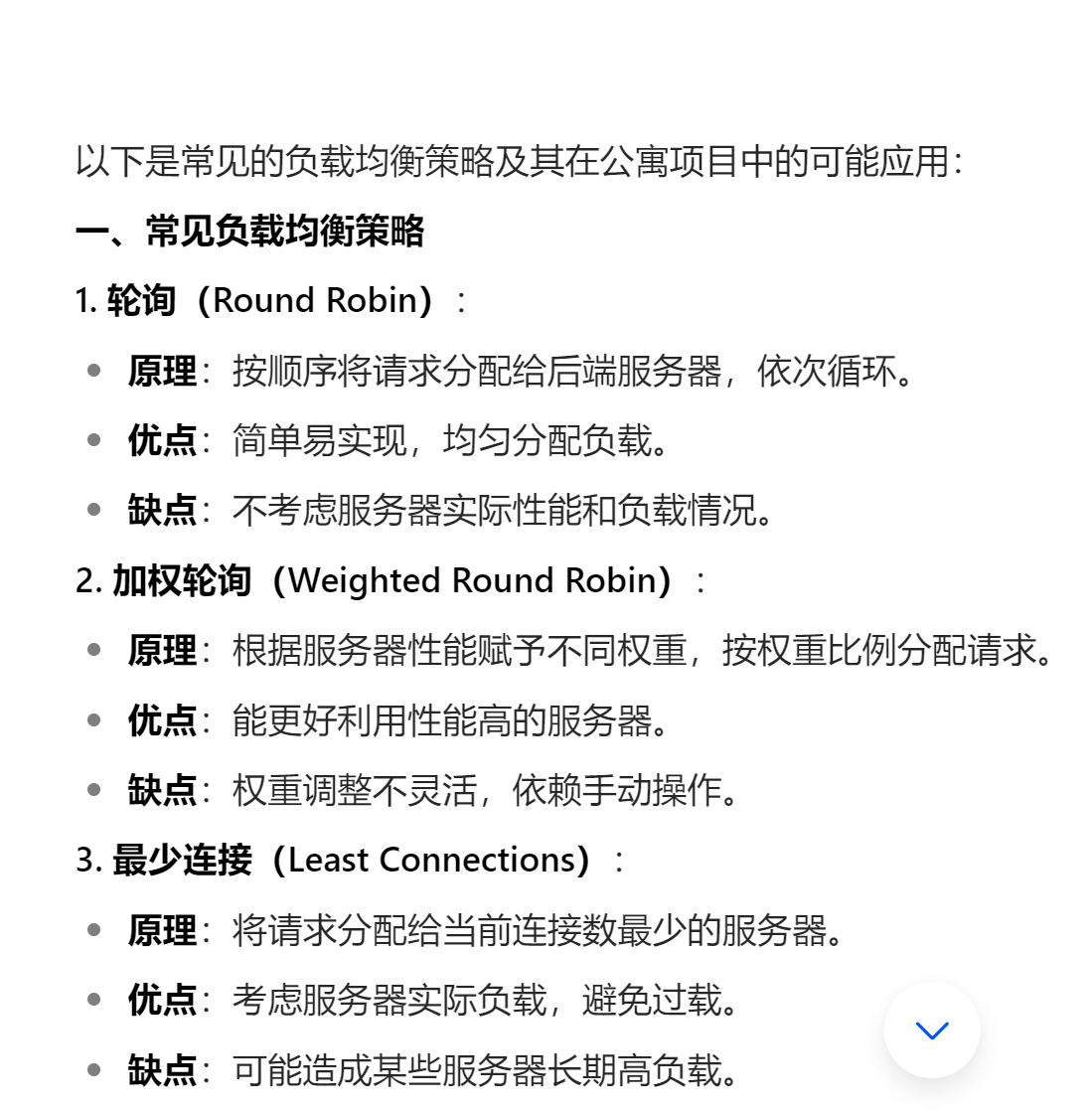
六、组合模式

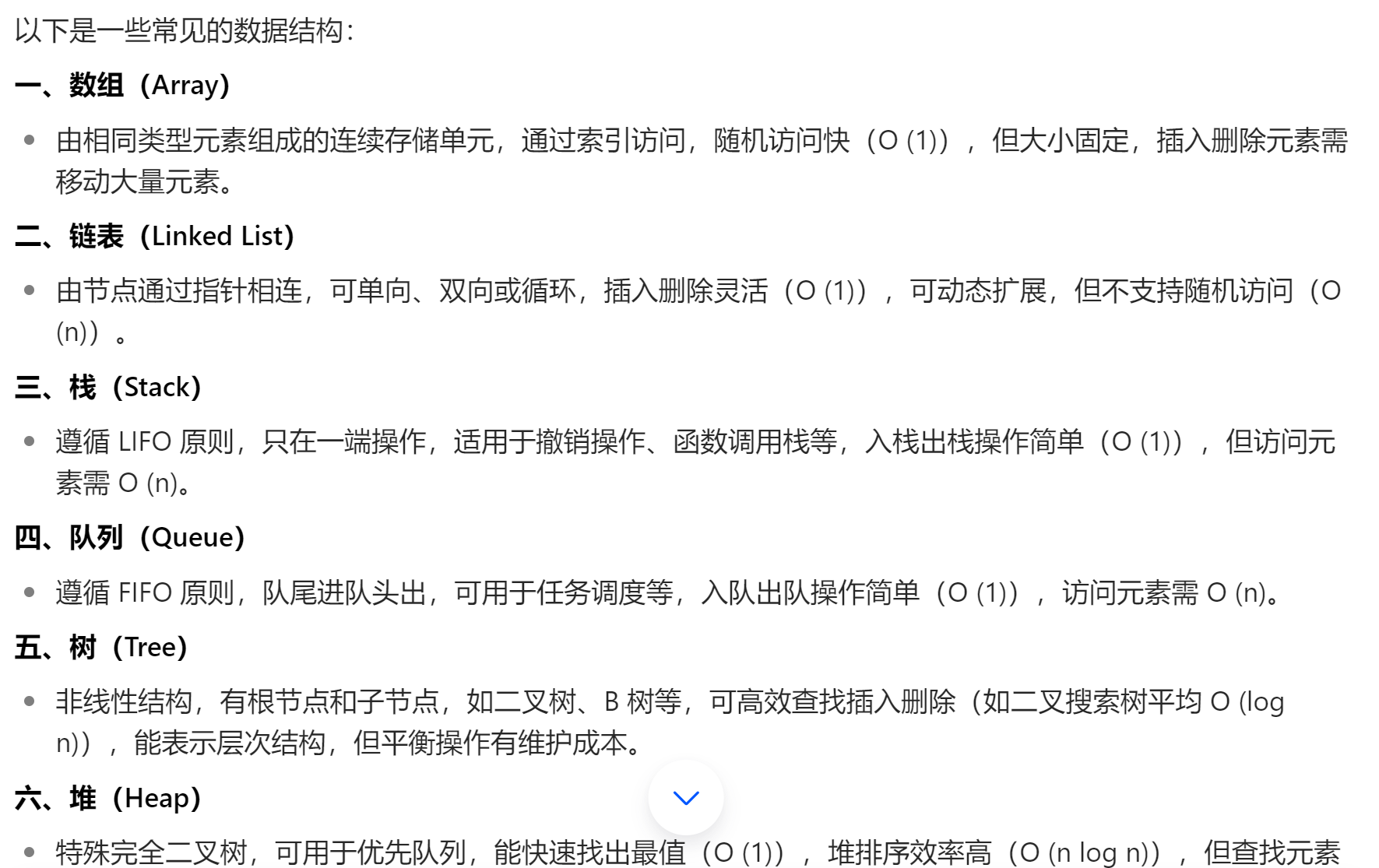
概念：

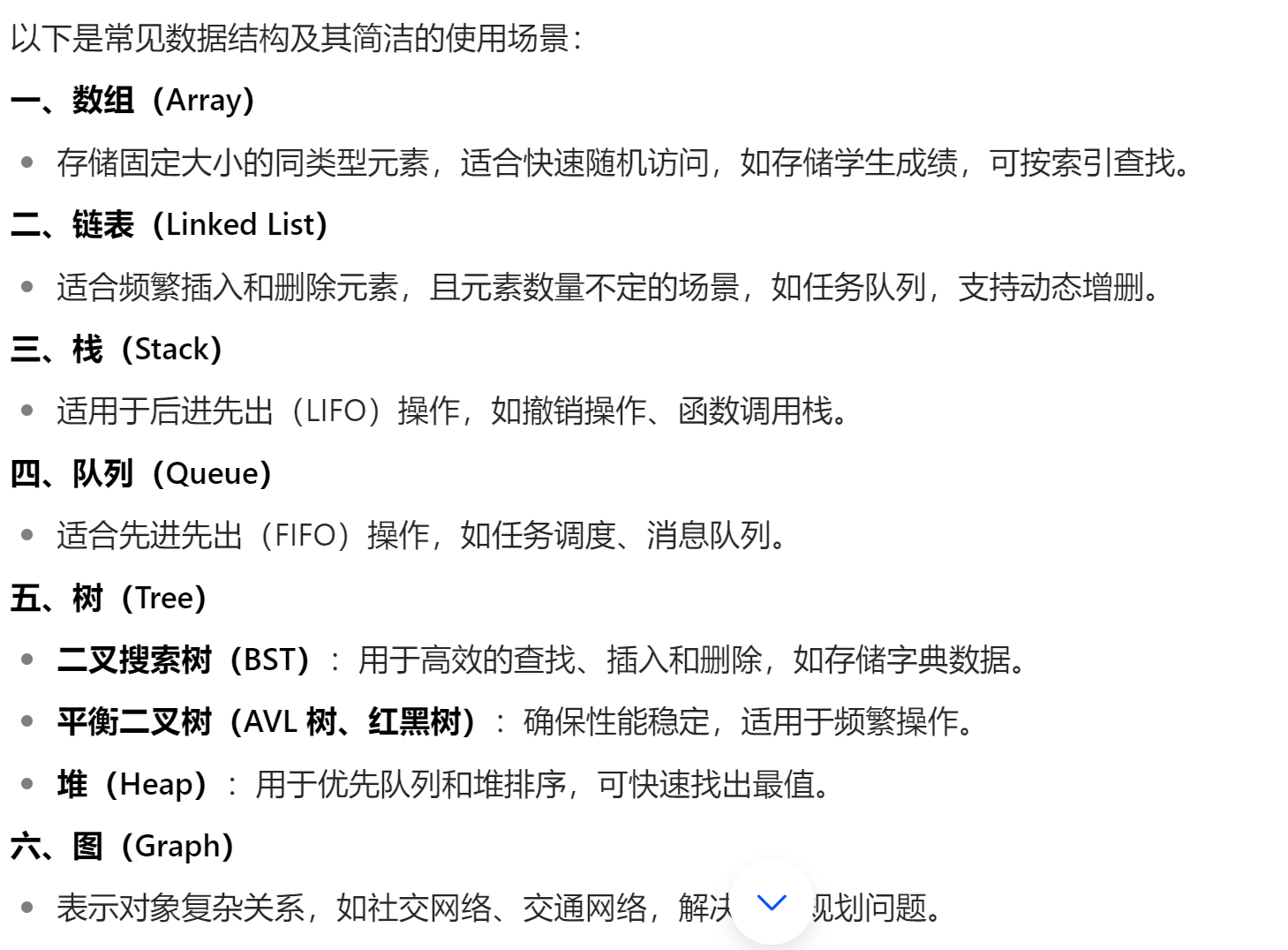
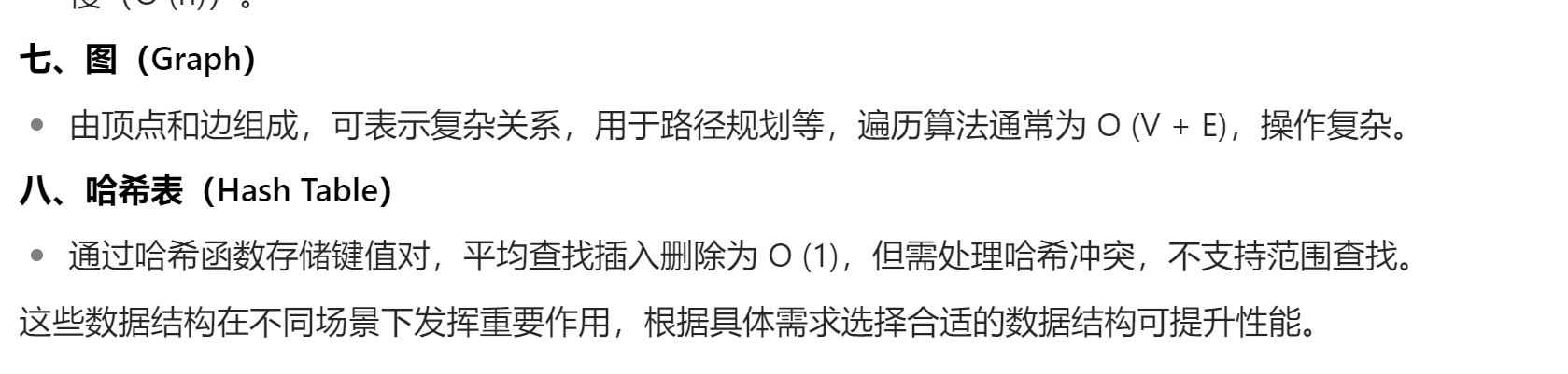
组合模式将对象组合成树形结构以表示 “部分 - 整体” 的层次结构，让客户端对单个对象和组合对象的使用具有一致性。

场景：

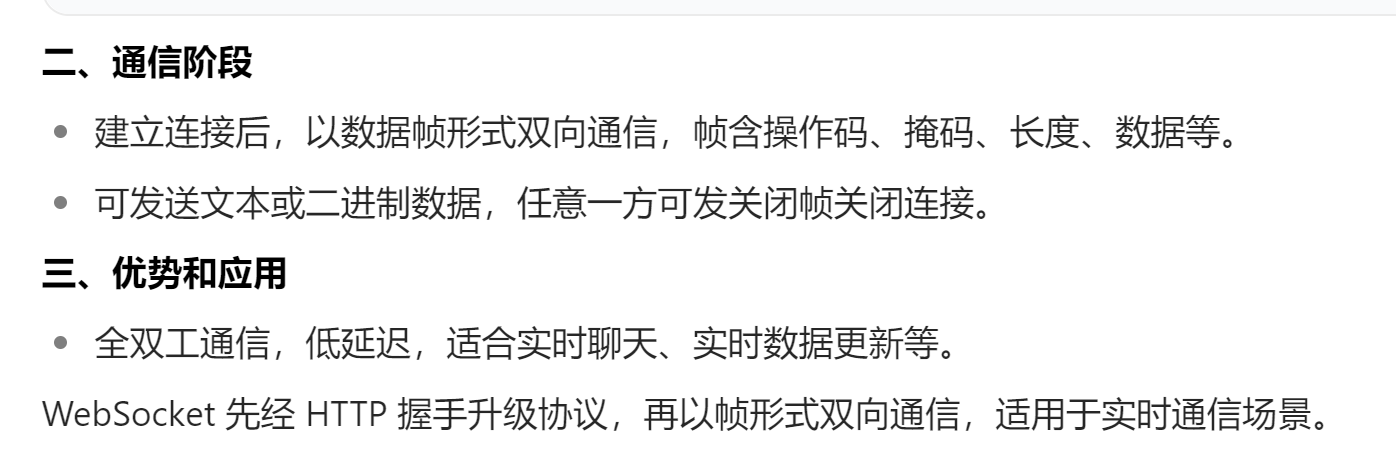
文件系统结构表示：在操作系统中，文件系统具有层次结构，可以使用组合模式表示文件和文件夹。文件夹可以包含文件和其他文件夹，对它们可以进行统一的操作，如复制、删除等。











以下是对 WebSocket 的理解：

一、基本概念

WebSocket 是一种在单个 TCP 连接上进行全双工通信的协议。它提供了一种持久连接，允许客户端和服务器之间进行双向实时通信，克服了传统 HTTP 协议的请求 - 响应模式的限制。

二、与 HTTP 的区别

通信模式：

HTTP 是一种无状态的请求 - 响应协议，客户端发起请求，服务器响应，通信结束后连接关闭。而 WebSocket 建立持久连接，可随时双向发送数据。

HTTP 每次请求都包含大量头部信息，开销较大；WebSocket 一旦连接建立，后续数据传输仅需发送数据部分，通信开销小。

三、建立连接过程

开始时，WebSocket 会借助 HTTP 协议进行握手，客户端发送一个特殊的 HTTP 请求，将 Upgrade 头设置为 websocket，并包含 Connection: Upgrade 及 Sec-WebSocket-Key 等信息，请求将协议从 HTTP 升级到 WebSocket。

服务器若支持，会返回 101 Switching Protocols 状态码及相应的 Sec-WebSocket-Accept 头，完成协议升级。

四、数据传输

数据以帧的形式在客户端和服务器之间传输，包括操作码（如文本、二进制、关闭帧等）、数据长度、掩码等信息。

支持文本和二进制数据传输，可实现不同类型数据的实时交互，满足各种应用需求。

五、应用场景

实时通信应用：

聊天应用，用户可以即时收发消息，无需像 HTTP 那样不断轮询服务器。

实时协作工具，如在线文档编辑，多个用户可同时操作，实时同步数据。

实时监控，如股票行情、设备监控等，能实时更新数据。

六、优势

实时性高：能快速更新数据，避免 HTTP 轮询或长轮询带来的延迟和资源浪费。

低延迟：减少头部信息，数据传输效率高，适合需要即时响应的场景。

资源节省：减少频繁建立和关闭连接的开销，对于大量实时通信的场景，能节省服务器和客户端资源。

七、开发使用

在前端，JavaScript 可使用 WebSocket API 轻松创建 WebSocket 连接并处理消息。

