前言

1、Request组成

- **客户端发送一个 HTTP 请求到服务器的请求消息包括以下格式：**
  - 请求行（request line）、请求头部（header）、空行和请求数据四个部分组成。
  - 第一部分：请求行，用来说明请求类型,要访问的资源以及所使用的 HTTP 版本.
  - 第二部分：请求头部，紧接着请求行（即第一行）之后的部分，用来说明服务器要使用的附加信息
  - 第三部分：空行，请求头部后面的空行是必须的
  - 第四部分：请求数据也叫主体，可以添加任意的其他数据。
- ok，来看一个标准的Http请求
  - ```
    POST /api/v1/orders HTTP/1.1
    Content-Type: application/json
    Content-Length: 45
    Authorization: Bearer eyJhbGci...xyz

    {"productId": "P1001", "quantity": 2}
    ```
头部字段	作用
`Host`	目标域名（HTTP/1.1 必须）
`Content-Type`	请求体格式（如 `text/html/JSON`）
`Content-Length`	请求体字节数（POST 必须）
`Authorization`	身份凭证（如 `Bearer token`）

2、Response 组成

一般情况下，服务器接收并处理客户端发过来的请求后会返回一个 HTTP 的响应消息。

HTTP 响应也由四个部分组成，分别是：状态行、消息报头、空行和响应正文。

- ok，来看一个标准的Http响应
    ```

- ```
  HTTP/1.1 200 OK
  Content-Type: application/json
  Content-Length: 42
  Date: Wed, 11 Jun 2025 11:08:06 GMT
  Server: Nginx/1.18

  {"status": "success", "data": {"id": "A203"}}
  ```

`Content-Type`	**响应体格式（如 `text/html`）**
`Content-Length`	响应体字节数
`Server`	服务器类型（如 `Nginx`）
`Set-Cookie`	设置客户端Cookie

3、HTTPS 中的 SSL 握手建立过程

简化如下：
1、客户端和服务端建立 SSL 握手，客户端通过 CA 证书来确认服务端的身份；
2、互相传递三个随机数，之后通过这随机数来生成一个密钥；
3、互相确认密钥，然后握手结束；
4、数据通讯开始，都使用同一个对话密钥来加解密；

4、响应码

- 1** 信息，服务器收到请求，需要请求者继续执行操作
- 2** 成功，操作被成功接收并处理
- 3** 重定向，需要进一步的操作以完成请求
- 4** 客户端错误，请求包含语法错误或无法完成请求
- 5** 服务器错误，服务器在处理请求的过程中发生了错误

5、socket 概念

套接字（socket）是通信的基石，是支持 TCP/IP 协议的网络通信的基本操作单元。它是网络通信过程中端点的抽象表示，包含进行网络通信必须的五种信息：连接使用的协议，本地主机的 IP 地址，本地进程的协议端口，远地主机的 IP 地址，远地进程的协议端口。

6.责任链模式（设计模式）

意图：将多个处理对象连接成一条链，并且沿着这条链传递请求，直到有对象处理它为止。

主要解决：职责链上的处理者负责处理请求，客户只需要将请求发送到职责链上即可，无须关心请求的处理细节和请求的传递，所以职责链将请求的发送者和请求的处理者解耦了。

在Android开发中，学过自定义view的应该也知道：ViewGroup 事件传递的递归调用就类似一条责任链，一旦其寻找到责任者，那么将由责任者持有并消费掉该次事件，具体体现在 View 的 onTouchEvent 方法中返回值的设置，如果 onTouchEvent 返回 false，那么意味着当前 View 不会是该次事件的责任人，将不会对其持有；如果为 true 则相反，此时 View 会持有该事件并不再向下传递。

01.OkHttp介绍

1.1 库的作用
  ```

OkHttp提供了对HTTP请求的底层封装，包括GET、POST等方法的实现。它的核心作用包括：

- 建立和管理HTTP连接
- 处理请求和响应
- 提供高效的网络通信能力

1.2 项目中使用的原因（优点）

- 支持现代HTTP协议（HTTP/2和SPDY）
  - HTTP/2支持多路复用（Multiplexing），允许在同一个TCP连接上同时发送多个请求和响应，从而减少延迟。SPDY是HTTP/2的前身，也有类似特性。OkHttp自动使用这些协议（如果服务器支持），否则会回退到HTTP/1.x。
- 连接池（Connection Pooling）
  - 在HTTP/1.x中，OkHttp通过连接池复用连接，避免频繁建立和断开TCP连接的开销。这显著提高了性能，尤其是在需要多次请求同一主机时。
- 透明的GZIP压缩
  - OkHttp自动添加`Accept-Encoding: gzip`请求头，并处理GZIP压缩的响应体，减少数据传输量，节省带宽。
- 响应缓存
  - OkHttp可以配置缓存，将响应存储到本地文件系统。当再次请求相同资源时，如果缓存有效，则直接使用缓存，避免网络请求。
- 自动重试与故障转移
  - OkHttp支持自动重试失败的请求（如连接超时），并且对于同一个主机有多个IP地址的情况，它会尝试其他IP地址（例如IPv4和IPv6）。
- 其他优点
  - **拦截器（Interceptors）**：强大的拦截器机制，允许在请求和响应过程中添加自定义逻辑（如日志、认证、重试）。
  - **WebSocket支持**：支持WebSocket协议，用于实时通信。
  - **易于集成**：与Retrofit等库无缝集成，简化REST API调用。