网络编程

黑马程序员
www.itheima.com

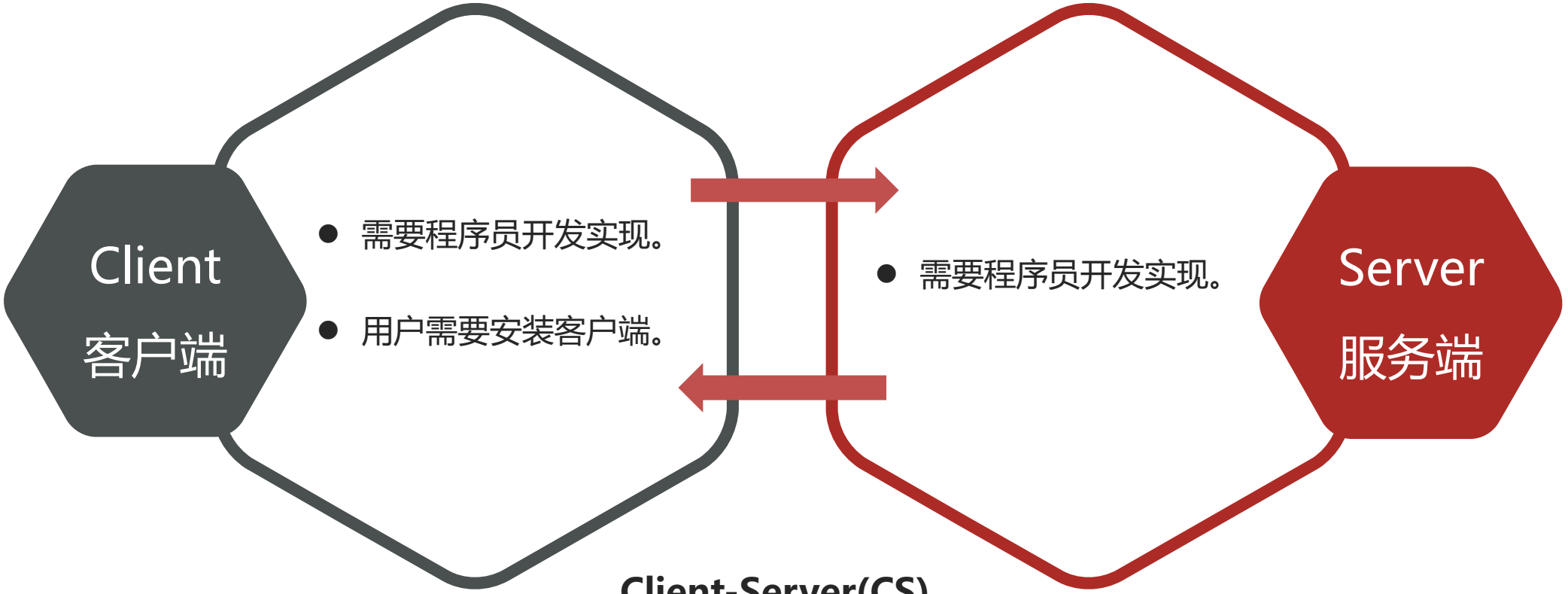
传智教育旗下
高端IT教育品牌

什么是网络编程？

- 网络编程可以让程序与网络上的其他设备中的程序进行数据交互。

网络通信基本模式

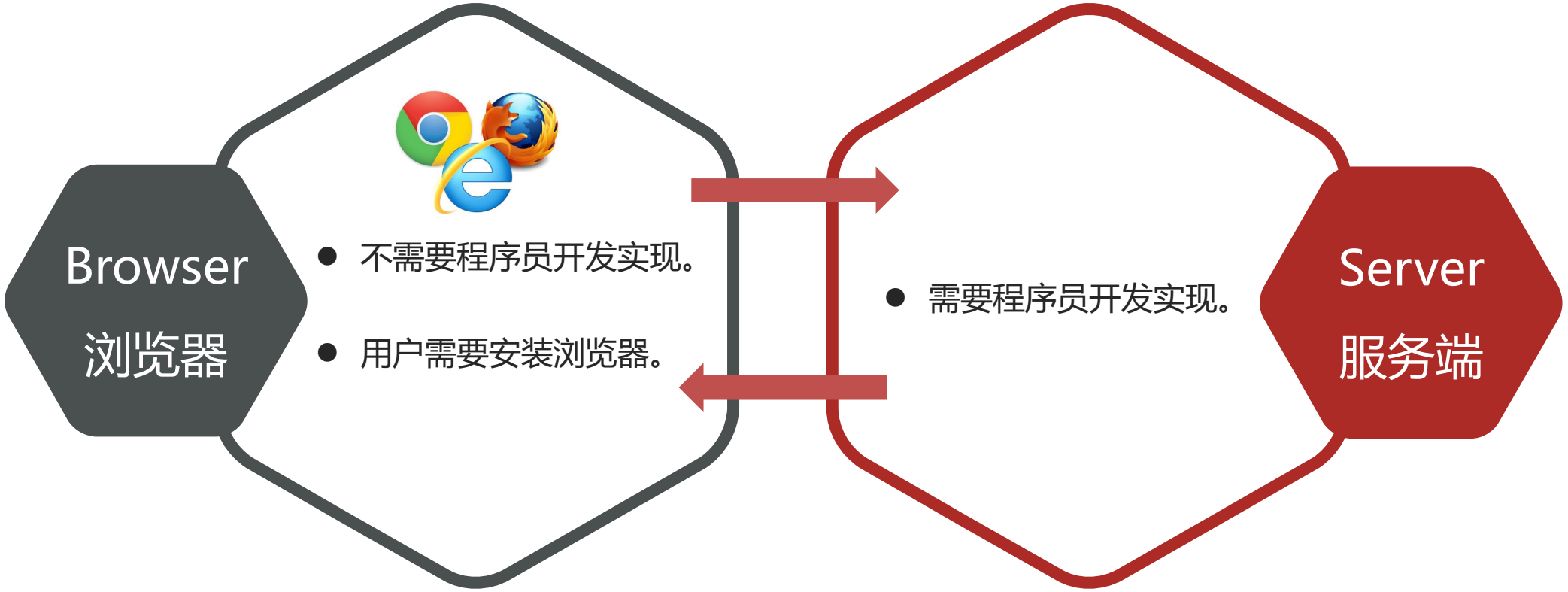
- 常见的通信模式有如下2种形式：Client-Server(CS) 、 Browser/Server(BS)



Client-Server(CS)

网络通信基本模式

- 常见的通信模式有如下2种形式：Client-Server(CS) 、 Browser/Server(BS)



Browser-Server(CS)

关于网络编程同学们需要学会什么

网络通信的三要素	UDP通信	TCP通信	即时通信	模拟BS系统
一个消息发送给对方需要哪些关键因素。	消息直接发送给对象，不确认对方是否在线，不做消息确认。	基于可靠传输的方式进行的通信模式。解决不同场景的通信需求。	如何实现即时通信，具体是是如何实现的。	WEB系统是如何支持访问到网页的，具体是如何与服务器通信的。

目录

Contents

- **网络通信三要素**
  - 三要素概述、要素一：IP地址
  - IP地址操作类-InetAddress
  - 要素二：端口号
  - 要素三：协议
- **UDP通信-快速入门**
- **UDP通信-广播、组播**
- **TCP通信-快速入门**
- **TCP通信-多发多收消息**
- **TCP通信-同时接受多个客户端消息**
- **TCP通信-使用线程池优化**
- **TCP通信实战案例-即时通信**
- **TCP通信实战案例-模拟BS系统**

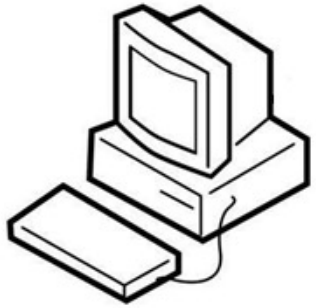
实现网络编程关键的三要素

IP地址：设备在网络中的地址，是唯一的标识。

端口：应用程序在设备中唯一的标识。

协议: 数据在网络中传输的规则，常见的协议有UDP协议和TCP协议。

发送：你瞅啥？

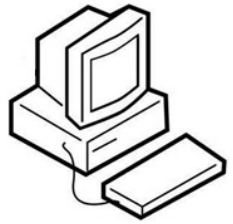


协议

端口：8888

IP地址2

IP地址1

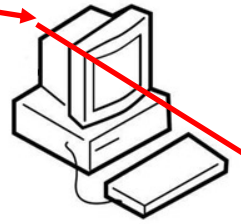


6666

7777

8888

IP地址2

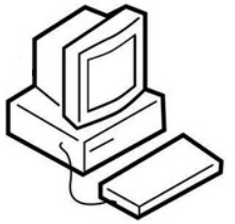


6666

7777

8888

IP地址3



6666

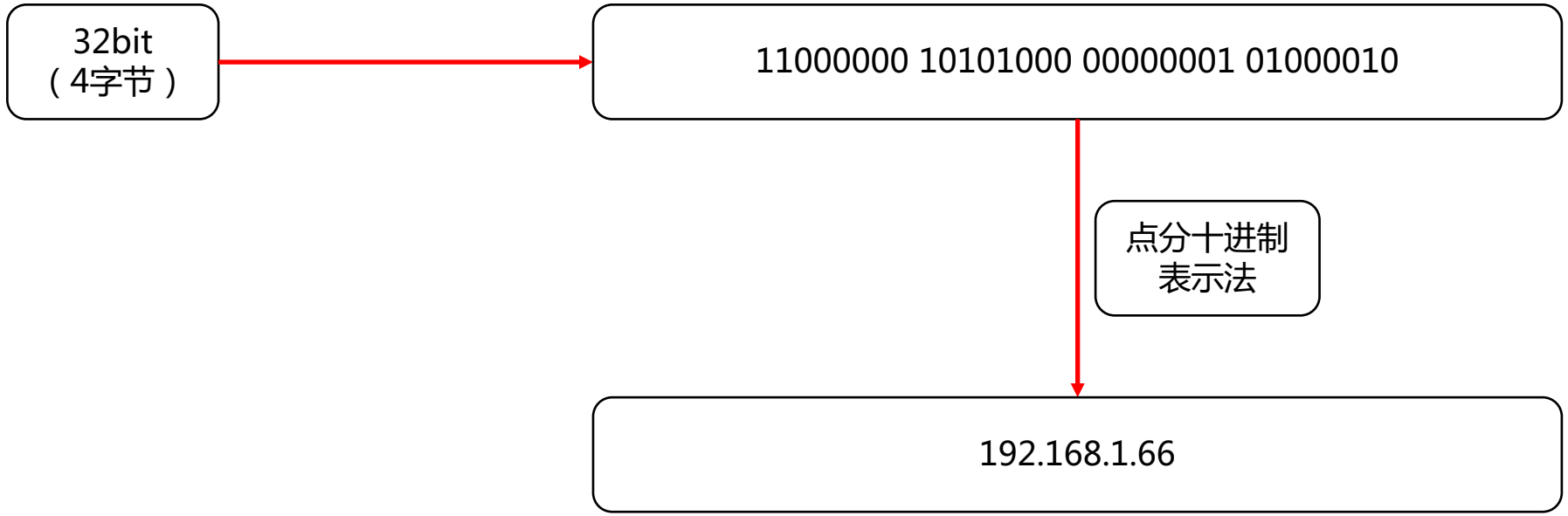
7777

8888

IP地址

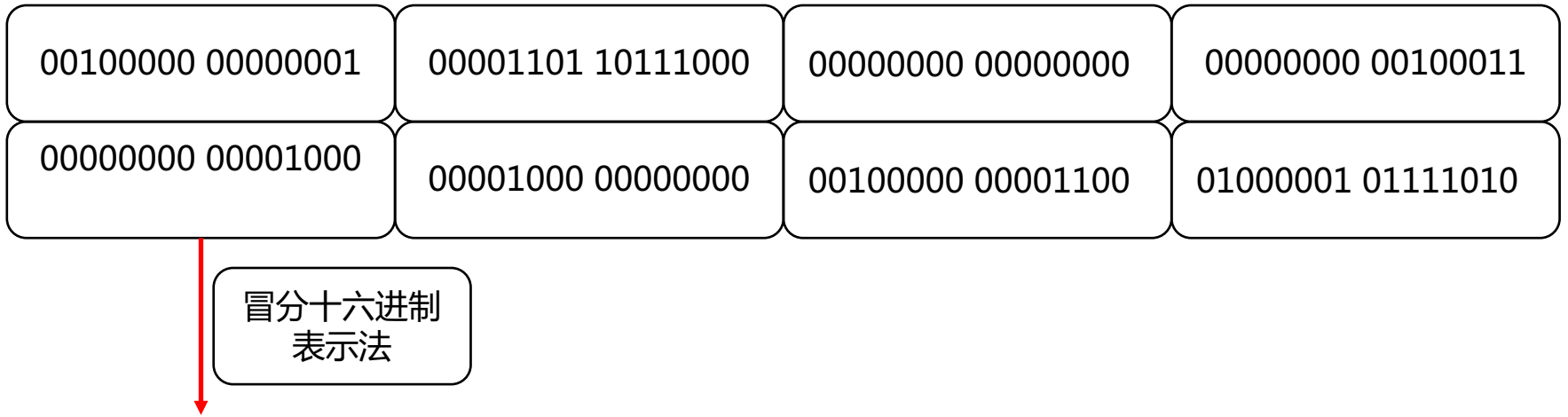
- IP（Internet Protocol）：全称”互联网协议地址”，是分配给上网设备的唯一标志。
- 常见的IP分类为：IPv4和IPv6

IPv4



IP地址

- IPv6：128位（16个字节），号称可以为地球每一粒沙子编号。
- IPv6分成8个整数，每个整数用四个十六进制位表示， 数之间用冒号（ ： ）分开。



IP地址基本寻路


域名:http://www.itcast.cn
IP地址
dns服务器
计算机
访问黑马程序员服务器
返回数据展示在浏览器上
黑马程序员服务器

IP地址形式：

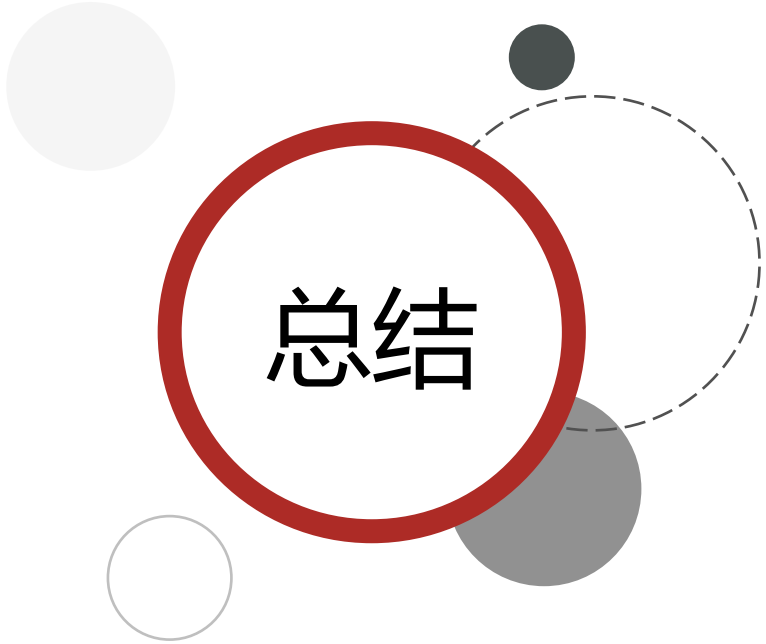
- 公网地址、和私有地址(局域网使用)。
- 192.168. 开头的就是常见的局域网地址，范围即为192.168.0.0--192.168.255.255，专门为组织机构内部使用。

IP常用命令：

- ipconfig：查看本机IP地址
- ping IP地址：检查网络是否连通

特殊IP地址：

- 本机IP: 127.0.0.1或者localhost：称为回送地址也可称本地回环地址，只会寻找当前所在本机。



1. 说说网络通信至少需要几个要素
    - **IP、端口、协议。**
2. IP地址是做什么的，具体有几种
    - **定位网络上的设备的，有IPv4，IPv6.**
3. 如何查看本机IP地址，如何看是否与对方互通
    - **ipcofig**
    - **ping 192.168.10.23**
4. 本机IP是谁？
    - **127.0.0.1或者是localhost**

目录

Contents

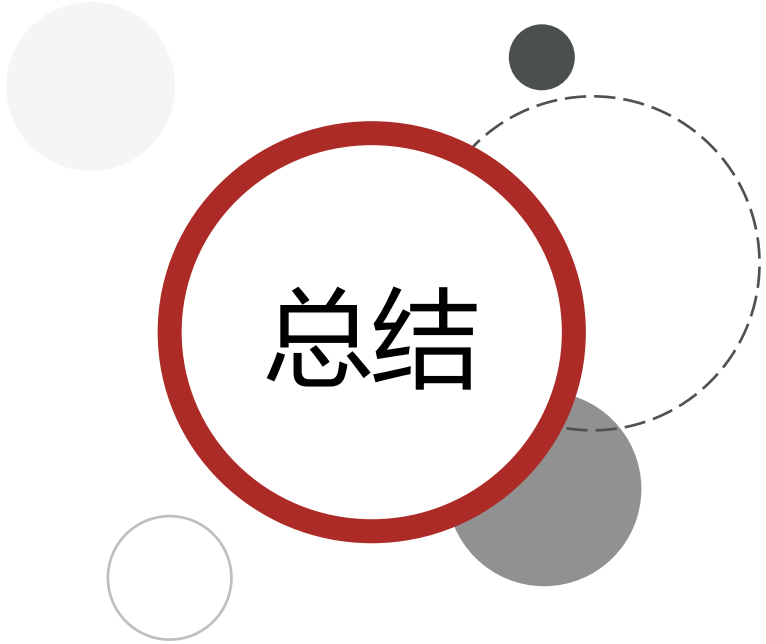
- **网络通信三要素**
  - 三要素概述、要素一：IP地址
  - IP地址操作类-InetAddress
  - 要素二：端口号
  - 要素三：协议
- **UDP通信-快速入门**
- **UDP通信-广播、组播**
- **TCP通信-快速入门**
- **TCP通信-多发多收消息**
- **TCP通信-同时接受多个客户端消息**
- **TCP通信-使用线程池优化**
- **TCP通信实战案例-即时通信**
- **TCP通信实战案例-模拟BS系统**

InetAddress 的使用

- 此类表示Internet协议（IP）地址。

InetAddress API如下

名称	说明
public static InetAddress getLocalHost()	返回本主机的地址对象
public static InetAddress getByName (String host)	得到指定主机的IP地址对象，参数是域名或者IP地址
public String getHostName ()	获取此IP地址的主机名
public String getHostAddress ()	返回IP地址字符串
public boolean isReachable(int timeout)	在指定毫秒内连通该IP地址对应的主机，连通返回true

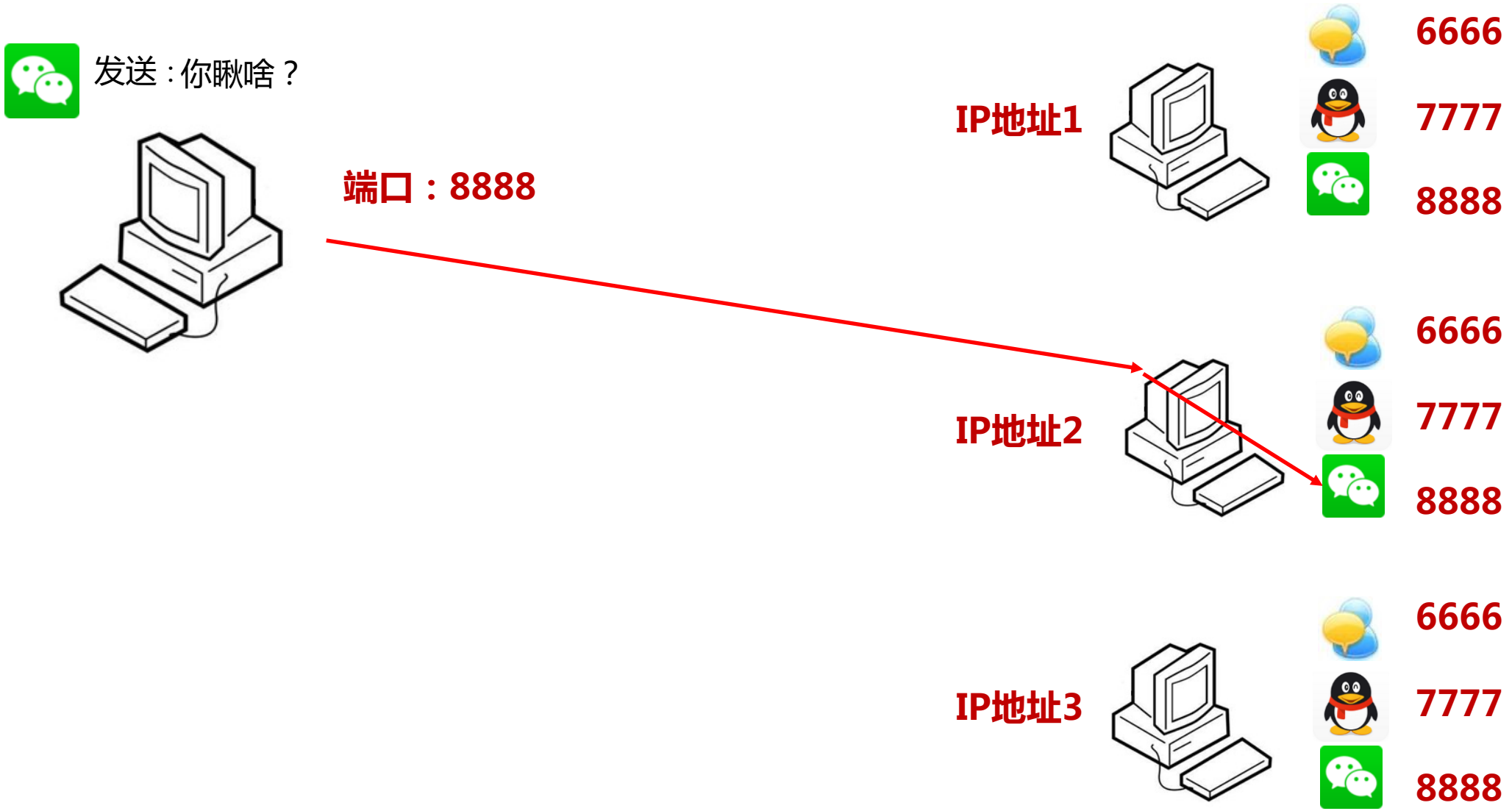


1. IP地址的代表类是谁？
    - **InetAddress类**
2. 如何获取本机IP对象
    - **public static InetAddress getLocalHost()**
3. 如何判断与该IP地址对象是否互通？
    - **public boolean isReachable(int timeout)**

目录

Contents

- **网络通信三要素**
  - 三要素概述、要素一：IP地址
  - IP地址操作类-InetAddress
  - 要素二：端口号
  - 要素三：协议
- **UDP通信-快速入门**
- **UDP通信-广播、组播**
- **TCP通信-快速入门**
- **TCP通信-多发多收消息**
- **TCP通信-同时接受多个客户端消息**
- **TCP通信-使用线程池优化**
- **TCP通信实战案例-即时通信**
- **TCP通信实战案例-模拟BS系统**


发送：你瞅啥？
端口：8888
IP地址1
6666
7777
8888
IP地址2
6666
7777
8888
IP地址3
6666
7777
8888

端口号

- 端口号：标识正在计算机设备上运行的进程（程序），被规定为一个 16 位的二进制，范围是 0~65535。

端口类型

- 周知端口：0~1023，被预先定义的知名应用占用（如：HTTP占用 80，FTP占用21）
- **注册端口**：1024~49151，分配给用户进程或某些应用程序。（如：Tomcat占 用8080，MySQL占用3306）
- 动态端口：49152到65535，之所以称为动态端口，是因为它 一般不固定分配某种进程，而是动态分配。

注意：我们自己开发的程序选择注册端口，且一个设备中不能出现两个程序的端口号一样，否则出错。

总结

1. 端口号的作用是什么？

   - **唯一标识正在计算机设备上运行的进程（程序）**

2. 一个设备中，能否出现2个应用程序的端口号一样，为什么？

   - **不可以，如果一样会出现端口冲突错误。**

目录

Contents

- **网络通信三要素**
  - 三要素概述、要素一：IP地址
  - IP地址操作类-InetAddress
  - 要素二：端口号
  - 要素三：协议
- **UDP通信-快速入门**
- **UDP通信-广播、组播**
- **TCP通信-快速入门**
- **TCP通信-多发多收消息**
- **TCP通信-同时接受多个客户端消息**
- **TCP通信-使用线程池优化**
- **TCP通信实战案例-即时通信**
- **TCP通信实战案例-模拟BS系统**

通信协议

- 连接和通信数据的规则被称为网络通信协议

网络通信协议有两套参考模型

- OSI参考模型：世界互联协议标准，全球通信规范，由于此模型过于理想化，未能在因特网上进行广泛推广。
- TCP/IP参考模型(或TCP/IP协议)：事实上的国际标准。

OSI参考模型	TCP/IP参考模型	各层对应	面向操作
应用层	应用层	HTTP、FTP、DNS、SMTP…	应用程序需要关注的：浏览器，邮箱。程序员一般在这一层开发
表示层			
会话层			
传输层	传输层	TCP、UDP…	选择使用的TCP，UDP协议
网络层	网络层	IP、ICMP…	封装源和目标IP，进行路径选择
数据链路层	数据链路层+物理	物理寻址、比特流…	物理设备中传输
物理层			

传输层的2个常见协议

- TCP(Transmission Control Protocol) ：传输控制协议
- UDP(User Datagram Protocol)：用户数据报协议

TCP协议特点

- 使用TCP协议，必须双方先建立连接，它是一种面向连接的可靠通信协议。
- 传输前，采用“三次握手”方式建立连接，所以是可靠的 。
- 在连接中可进行大数据量的传输 。
- 连接、发送数据都需要确认，且传输完毕后，还需释放已建立的连接，通信效率较低。

TCP协议通信场景

- 对信息安全要求较高的场景，例如：文件下载、金融等数据通信。

TCP三次握手确立连接

TCP三次握手确立连接

你瞅啥？

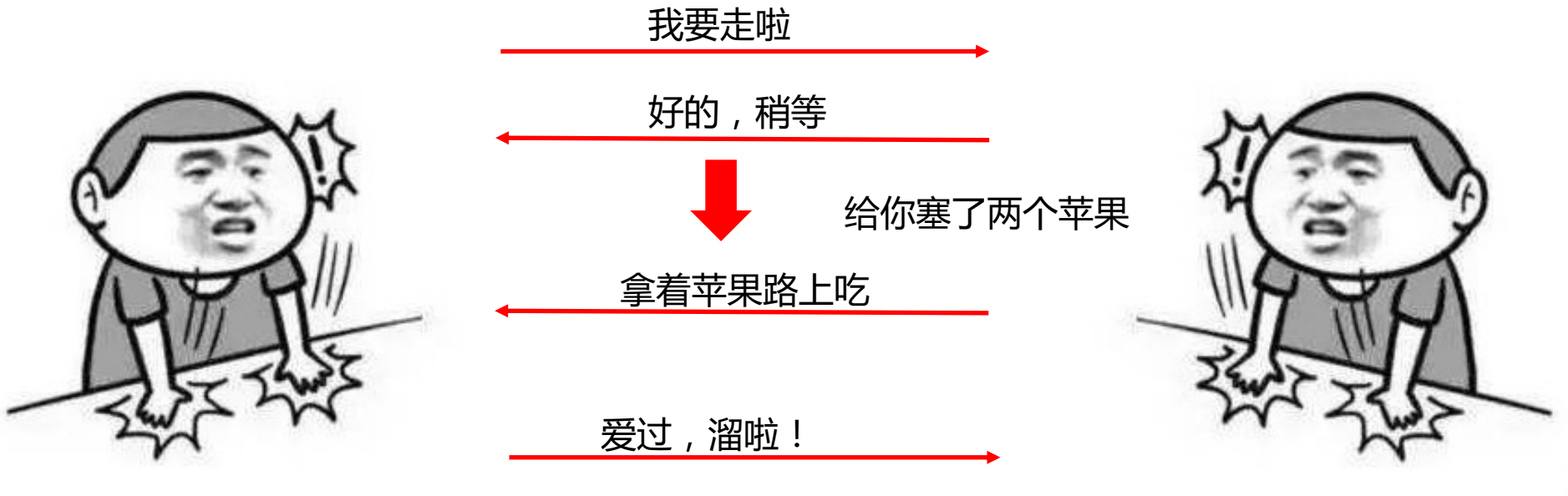
瞅你咋地？

走，咱俩唠唠

于是两人关系确立

TCP四次挥手断开连接

TCP四次挥手断开连接

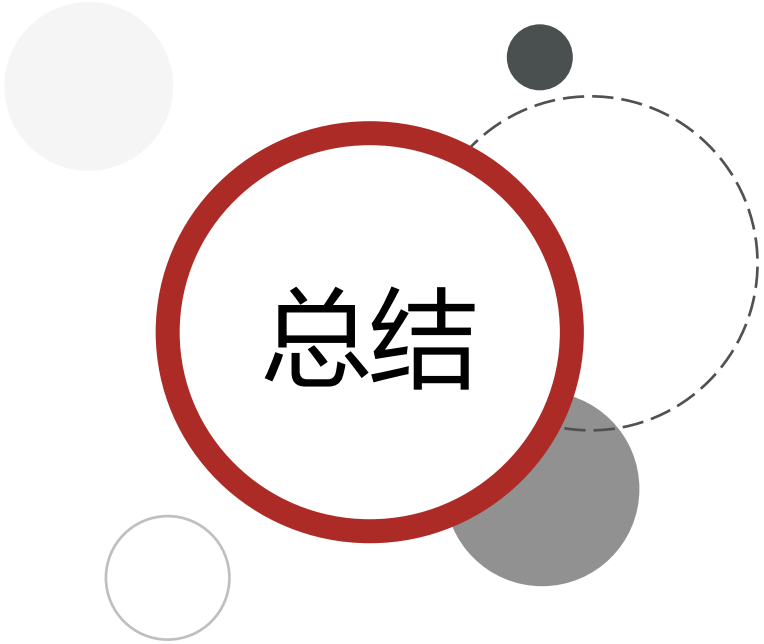


UDP协议：

- UDP是一种无连接、不可靠传输的协议。
- 将数据源IP、目的地IP和端口封装成数据包，不需要建立连接
- 每个数据包的大小限制在64KB内
- 发送不管对方是否准备好，接收方收到也不确认，故是不可靠的
- 可以广播发送 ，发送数据结束时无需释放资源，开销小，速度快。

UDP协议通信场景

- 语音通话，视频会话等。

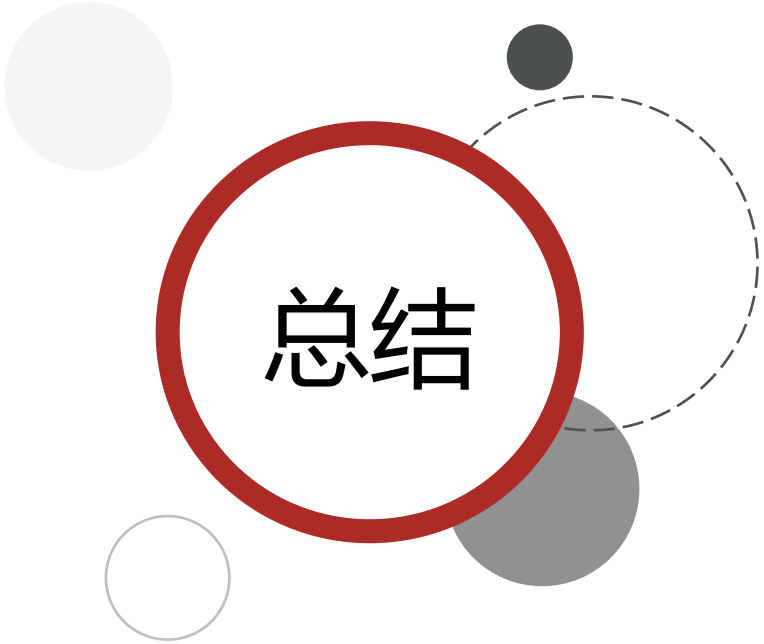


1、通信协议是什么？

- **计算机网络中，连接和通信数据的规则被称为网络通信协议。**

2、TCP通信协议的特点是什么样的?

- **它是一种面向连接的可靠通信协议。**
- **传输前，采用“三次握手”方式建立连接，点对点的通信，所以可靠。**
- **在连接中可进行大数据量的传输。**
- **通信效率较低。**



3、UDP协议的特点是什么

- **用户数据报协议(User Datagram Protocol)**
- **UDP是面向无连接，不可靠传输的通信协议。**
- **速度快，有大小限制一次最多发送64K，数据不安全，易丢失数据。**

目录

Contents

- 网络通信三要素
- UDP通信
  - UDP通信：快速入门
  - UDP通信：多发多收
- UDP通信-广播、组播
- TCP通信-快速入门
- TCP通信-多发多收消息
- TCP通信-同时接受多个客户端消息
- TCP通信-使用线程池优化
- TCP通信实战案例-即时通信
- TCP通信实战案例-模拟BS系统

UDP协议的特点

- UDP是一种无连接、不可靠传输的协议。
- 将数据源IP、目的地IP和端口以及数据封装成数据包，大小限制在64KB内，直接发送出去即可。

UDP协议通信模型演示

街道

发送端

接收端

DatagramPacket：数据包对象（韭菜盘子）

构造器	说明
public DatagramPacket(byte[] buf, int length, InetAddress address, int port)	创建发送端数据包对象 buf：要发送的内容，字节数组 length：要发送内容的字节长度 address：接收端的IP地址对象 port：接收端的端口号
public DatagramPacket(byte[] buf, int length)	创建接收端的数据包对象 buf：用来存储接收的内容 length：能够接收内容的长度

DatagramPacket常用方法

方法	说明
public int getLength()	获得实际接收到的字节个数

DatagramSocket：发送端和接收端对象（人）

构造器	说明
public DatagramSocket()	创建发送端的Socket对象，系统会随机分配一个端口号。
public DatagramSocket(int port)	创建接收端的Socket对象并指定端口号

DatagramSocket类成员方法

方法	说明
public void send(DatagramPacket dp)	发送数据包
public void receive(DatagramPacket p)	接收数据包

步骤

使用UDP通信实现：发送消息、接收消息

需求：客户端实现步骤

① 创建DatagramSocket对象（发送端对象） → 扔韭菜的人

② 创建DatagramPacket对象封装需要发送的数据（数据包对象） → 韭菜盘子

③ 使用DatagramSocket对象的send方法传入DatagramPacket对象 → 开始抛出韭菜

④ 释放资源

步骤

使用UDP通信实现：发送消息、接收消息

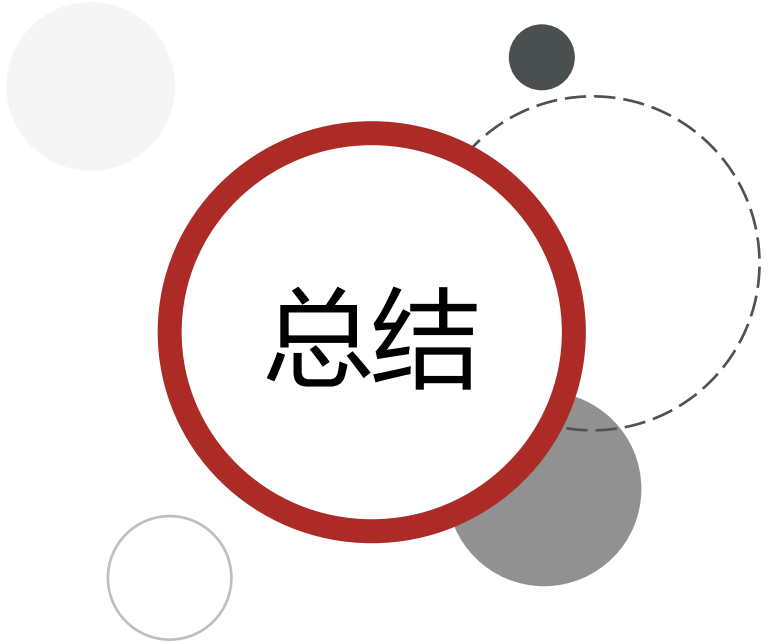
需求：接收端实现步骤

① 创建DatagramSocket对象并指定端口（接收端对象） → 接韭菜的人

② 创建DatagramPacket对象接收数据（数据包对象） → 韭菜盘子

③ 使用DatagramSocket对象的receive方法传入DatagramPacket对象 → 开始接收韭菜

④ 释放资源



1．UDP发送端和接收端的对象是哪个？

- **public DatagramSocket() ：创建发送端的Socket对象**
- **public DatagramSocket(int port)：创建接收端的Socket对象**

2．数据包对象是哪个？

- **DatagramPacket**

3．如何发送、接收数据包

- **使用DatagramSocket的如下方法：**
- **public void send(DatagramPacket dp)：发送数据包**
- **public void receive(DatagramPacket dp) ：接收数据包**

目录

Contents

- 网络通信三要素
- UDP通信
  - UDP通信：快速入门
  - UDP通信：多发多收
- UDP通信-广播、组播
- TCP通信-快速入门
- TCP通信-多发多收消息
- TCP通信-同时接受多个客户端消息
- TCP通信-使用线程池优化
- TCP通信实战案例-即时通信
- TCP通信实战案例-模拟BS系统

案例 使用UDP通信实现：多发多收消息

需求

- 使用UDP通信方式开发接收端和发送端。

分析

① 发送端可以一直发送消息。

② 接收端可以不断的接收多个发送端的消息展示。

③ 发送端输入了exit则结束发送端程序。

步骤

发送端可以反复发送数据

需求：客户端实现步骤

① 创建DatagramSocket对象（发送端对象） → 扔韭菜的人

② 使用while死循环不断的接收用户的数据输入，如果用户输入的exit则退出程序

③ 如果用户输入的不是exit，把数据封装成DatagramPacket → 韭菜盘子

④ 使用DatagramSocket对象的send方法将数据包对象进行发送 → 开始抛出韭菜

⑤ 释放资源

步骤

接收端可以反复接收数据

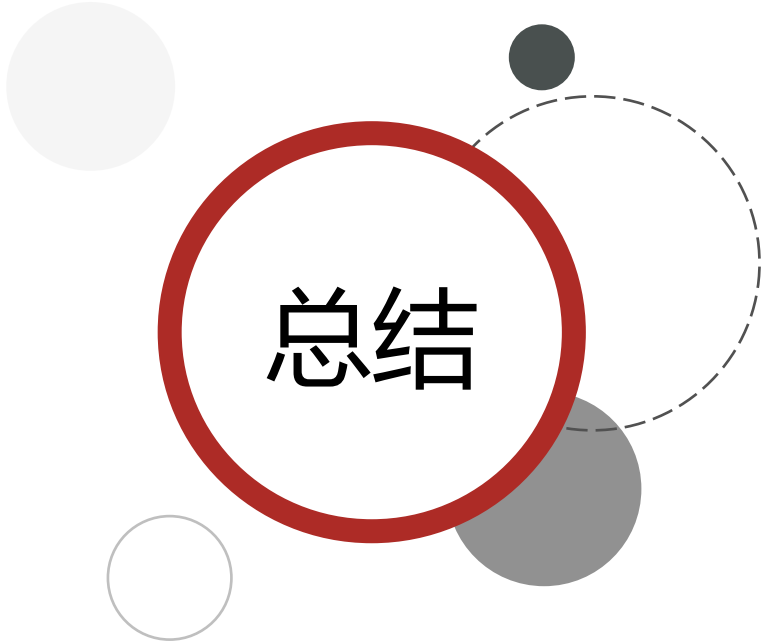
需求：接收端实现步骤

① 创建DatagramSocket对象并指定端口（接收端对象） → 接韭菜的人

② 创建DatagramPacket对象接收数据（数据包对象） → 韭菜盘子

③ 使用while死循环不断的进行第4步

④ 使用DatagramSocket对象的receive方法传入DatagramPacket对象 → 开始接收韭菜



1. UDP的接收端为什么可以接收很多发送端的消息？

- **接收端只负责接收数据包，无所谓是哪个发送端的数据包.**

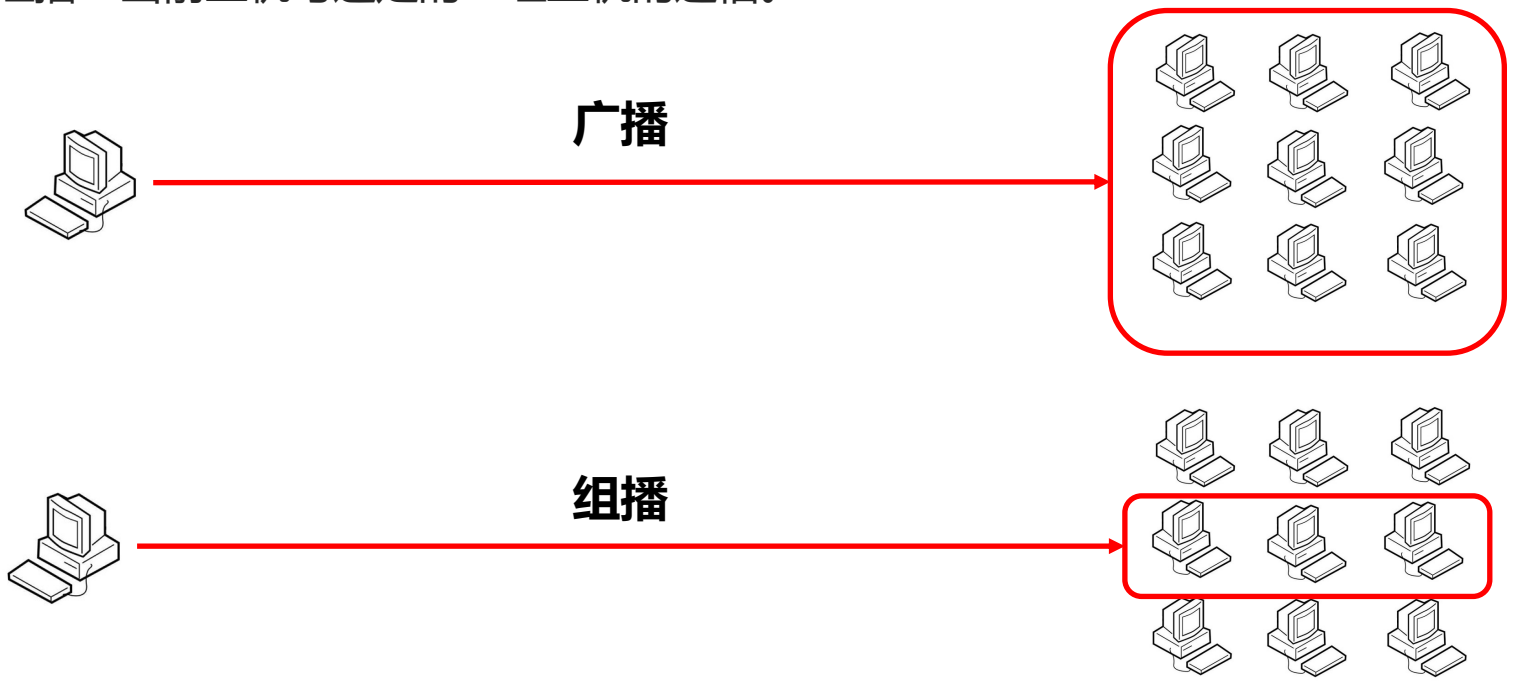
目录

Contents

- 网络通信三要素
- UDP通信-快速入门
- UDP通信-广播、组播
- TCP通信-快速入门
- TCP通信-多发多收消息
- TCP通信-同时接受多个客户端消息
- TCP通信-使用线程池优化
- TCP通信实战案例-即时通信
- TCP通信实战案例-模拟BS系统

UDP的三种通信方式

- 单播：单台主机与单台主机之间的通信。
- 广播：当前主机与所在网络中的所有主机通信。
- 组播：当前主机与选定的一组主机的通信。



UDP如何实现广播

- 使用广播地址：255.255.255.255
- 具体操作：
  ① 发送端发送的数据包的目的地写的是广播地址、且指定端口。（255.255.255.255 ， 9999）
  ② 本机所在网段的其他主机的程序只要注册对应端口就可以收到消息了。（9999）

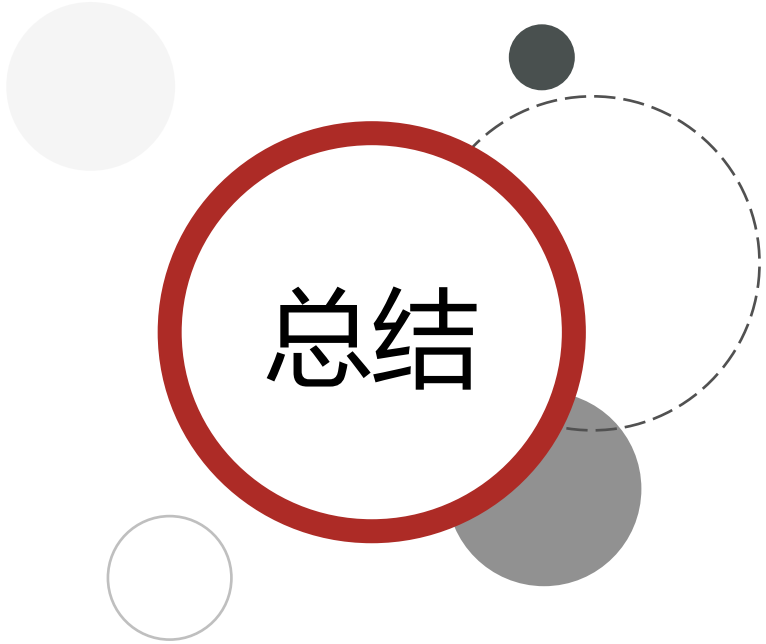
UDP如何实现组播

- 使用组播地址：224.0.0.0 ~ 239.255.255.255
- 具体操作：

① 发送端的数据包的目的地是组播IP（例如：224.0.1.1，端口：9999)

② 接收端必须绑定该组播IP(224.0.1.1)，端口还要注册发送端的目的端口9999 ，这样即可接收该组播消息。

③ DatagramSocket的子类MulticastSocket可以在接收端绑定组播IP。



1. 如何实现广播，具体怎么操作？
   - **发送端目的IP使用广播IP： 255.255.255.255 9999。**
   - **所在网段的其他主机对应了端口（9999）即可接收消息。**
2. 如何实现组播，具体怎么操作？
   - **发送端目的IP使用组播IP，且指定端口。**
   - **所在网段的其他主机注册了该组播IP和对应端口即可接收消息。**

目录

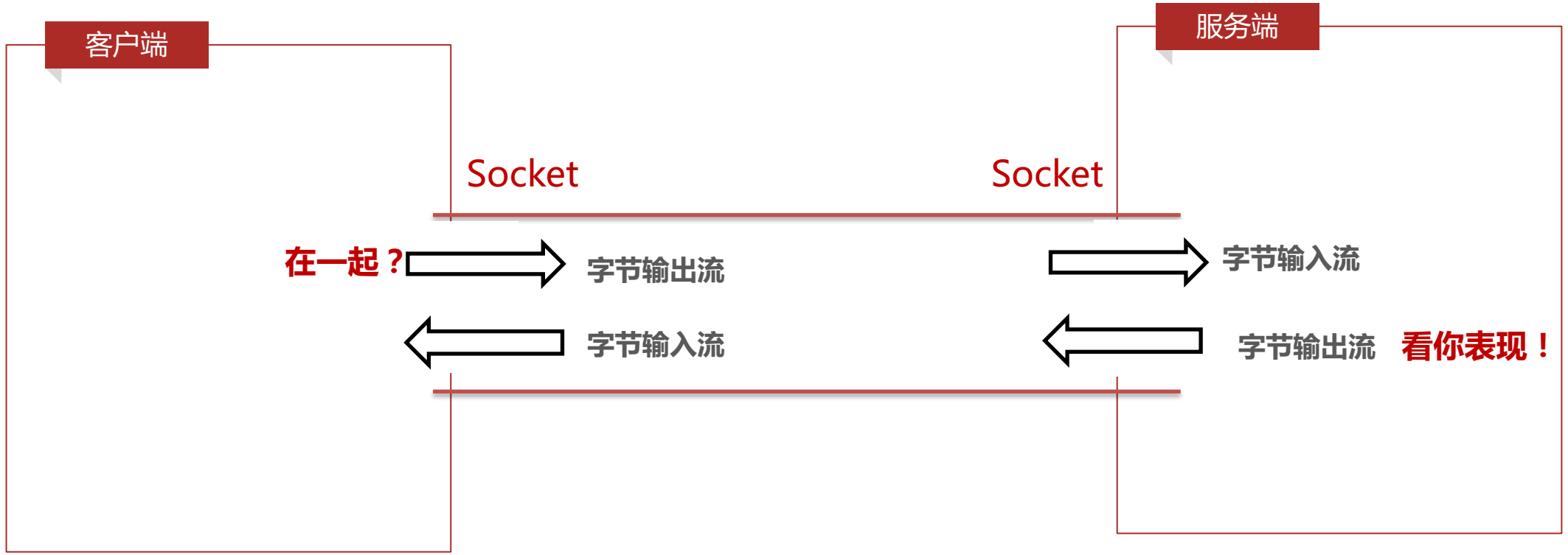
Contents

- 网络通信三要素
- UDP通信-快速入门
- UDP通信-广播、组播
- TCP通信-快速入门
  - 编写客户端代码
  - 编写服务端代码、原理分析
- TCP通信-多发多收消息
- TCP通信-同时接受多个客户端消息
- TCP通信-使用线程池优化
- TCP通信实战案例-即时通信
- TCP通信实战案例-模拟BS系统

TCP协议回顾：

- TCP是一种面向连接，安全、可靠的传输数据的协议
- 传输前，采用“三次握手”方式，点对点通信，是可靠的
- 在连接中可进行大数据量的传输 □

TCP通信模式演示：



注意：在java中只要是使用java.net.Socket类实现通信，底层即是使用了TCP协议

Socket

构造器	说明
public Socket(String host , int port)	创建发送端的Socket对象与服务端连接，参数为服务端程序的ip和端口。

Socket类成员方法

方法	说明
OutputStream getOutputStream()	获得字节输出流对象
InputStream getInputStream()	获得字节输入流对象

步骤

客户端发送消息

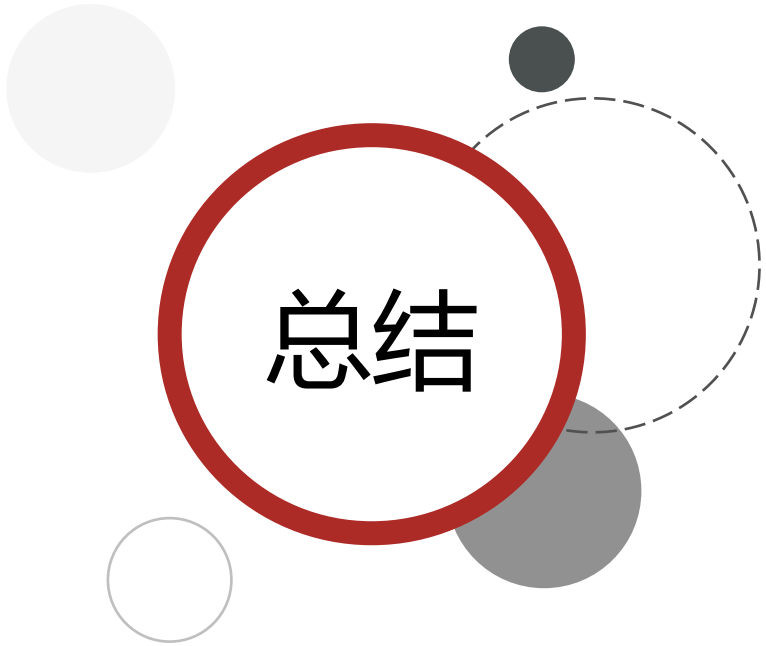
需求：客户端实现步骤

① 创建客户端的Socket对象，请求与服务端的连接。

② 使用socket对象调用getOutputStream()方法得到字节输出流。

③ 使用字节输出流完成数据的发送。

④ 释放资源：关闭socket管道。



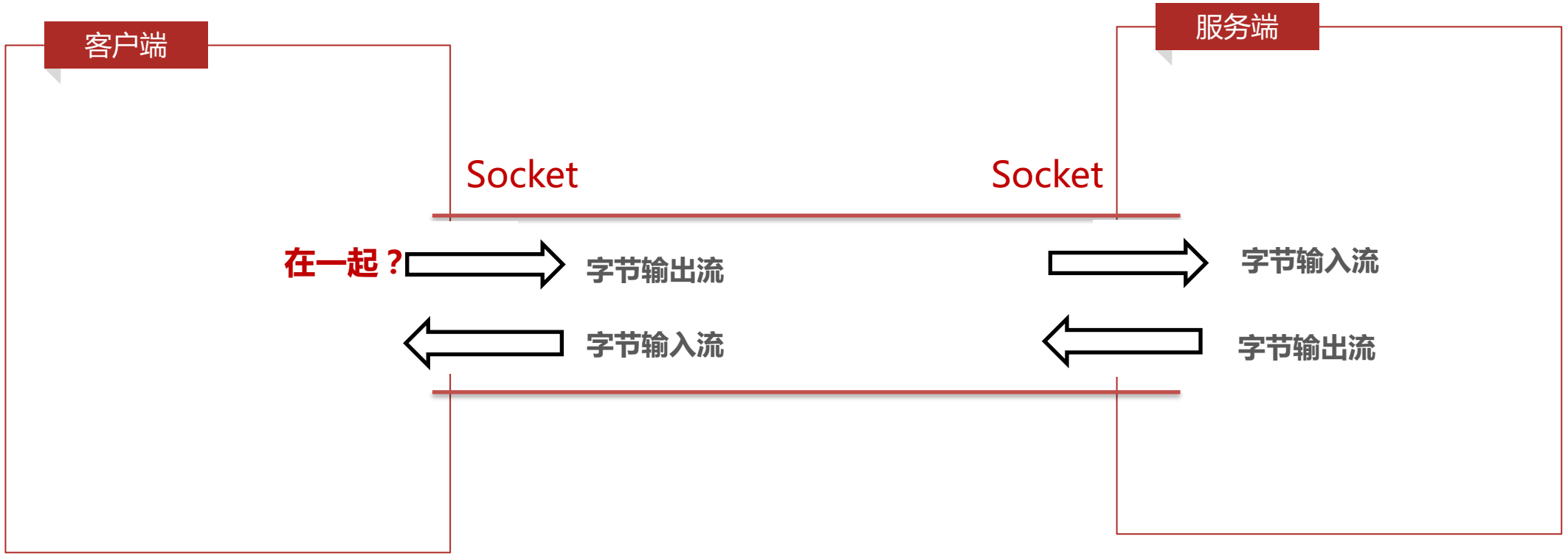
1. TCP通信的客户端的代表类是谁？
   - **Socket类**
   - **public Socket(String host , int port)**
2. TCP通信如何使用Socket管道发送、接收数据。
   - **OutputStream getOutputStream()：获得字节输出流对象（发）**
   - **InputStream getInputStream()：获得字节输入流对象（收）**

目录

Contents

- 网络通信三要素
- UDP通信-快速入门
- UDP通信-广播、组播
- TCP通信-快速入门
  - 编写客户端代码
  - 编写服务端代码、原理分析
- TCP通信-多发多收消息
- TCP通信-同时接受多个客户端消息
- TCP通信-使用线程池优化
- TCP通信实战案例-即时通信
- TCP通信实战案例-模拟BS系统

TCP通信模式演示：



ServerSocket(服务端)

构造器	说明
`public ServerSocket(int port)`	注册服务端端口

ServerSocket类成员方法

方法	说明
`public Socket accept()`	等待接收客户端的Socket通信连接 连接成功返回Socket对象与客户端建立端到端通信

步骤

服务端实现接收消息

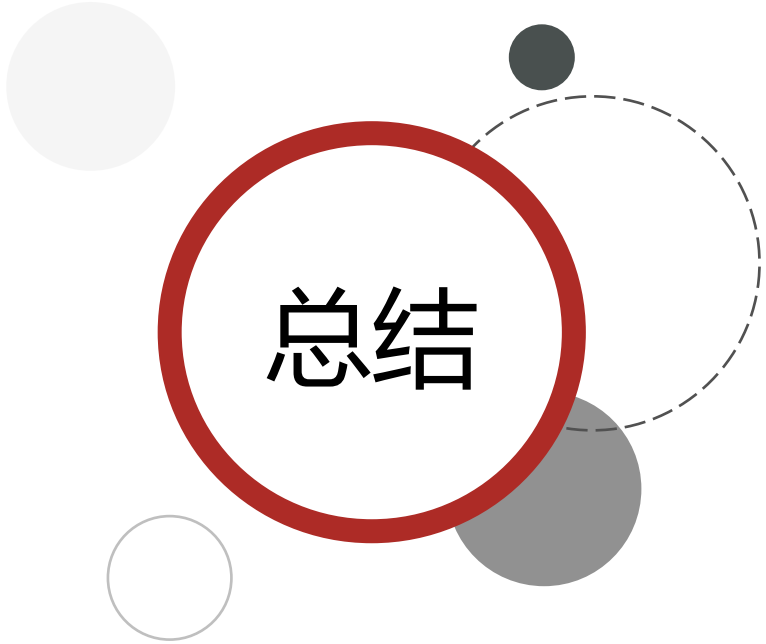
需求：服务端实现步骤

① 创建ServerSocket对象，注册服务端端口。

② 调用ServerSocket对象的accept()方法，等待客户端的连接，并得到Socket管道对象。

③ 通过Socket对象调用getInputStream()方法得到字节输入流、完成数据的接收。

④ 释放资源：关闭socket管道



1．TCP通信服务端用的代表类？

- **ServerSocket类,注册端口。**
- **调用accept()方法阻塞等待接收客户端连接。得到Socket对象。**

2．TCP通信的基本原理？

- **客户端怎么发，服务端就应该怎么收。**
- **客户端如果没有消息，服务端会进入阻塞等待。**
- **Socket一方关闭或者出现异常、对方Socket也会失效或者出错。**

目录

Contents

- 网络通信三要素
- UDP通信-快速入门
- UDP通信-广播、组播
- TCP通信-快速入门
- **TCP通信-多发多收消息**
- TCP通信-同时接受多个客户端消息
- TCP通信-使用线程池优化
- TCP通信实战案例-即时通信
- TCP通信实战案例-模拟BS系统

案例

使用TCP通信实现：多发多收消息

需求：使用TCP通信方式实现：多发多收消息。

具体要求：

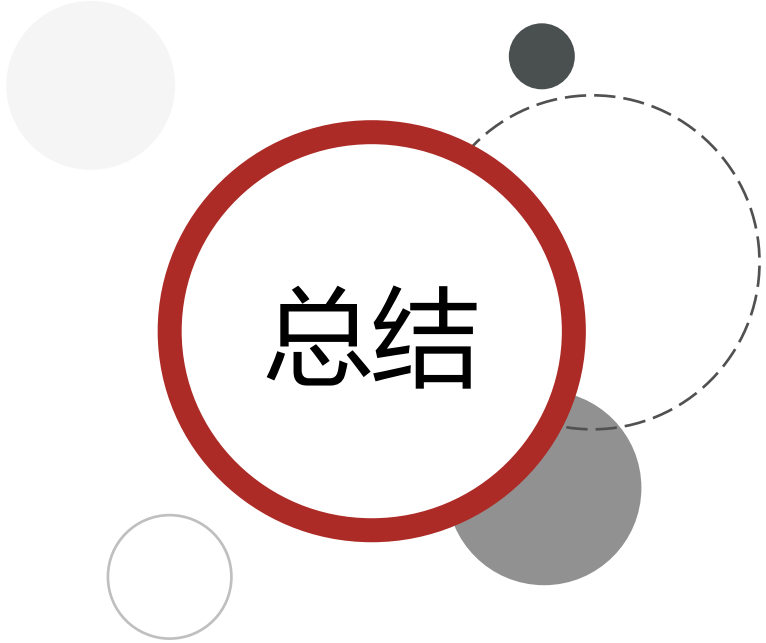
① 可以使用死循环控制服务端收完消息继续等待接收下一个消息。

② 客户端也可以使用死循环等待用户不断输入消息。

③ 客户端一旦输入了exit，则关闭客户端程序，并释放资源。

本案例实现了多发多收，那么是否可以同时接收多个客户端的消息？

- **不可以的。**
- **因为服务端现在只有一个线程，只能与一个客户端进行通信。**



1．本次多发多收是如何实现的

- **客户端使用循环反复地发送消息。**
- **服务端使用循环反复地接收消息。**

2．现在服务端为什么不可以同时接收多个客户端的消息。

- **目前服务端是单线程的，每次只能处理一个客户端的消息。**

目录

Contents

- 网络通信三要素
- UDP通信-快速入门
- UDP通信-广播、组播
- TCP通信-快速入门
- TCP通信-多发多收消息
- **TCP通信-同时接受多个客户端消息[重点]**
- TCP通信-使用线程池优化
- TCP通信实战案例-即时通信
- TCP通信实战案例-模拟BS系统

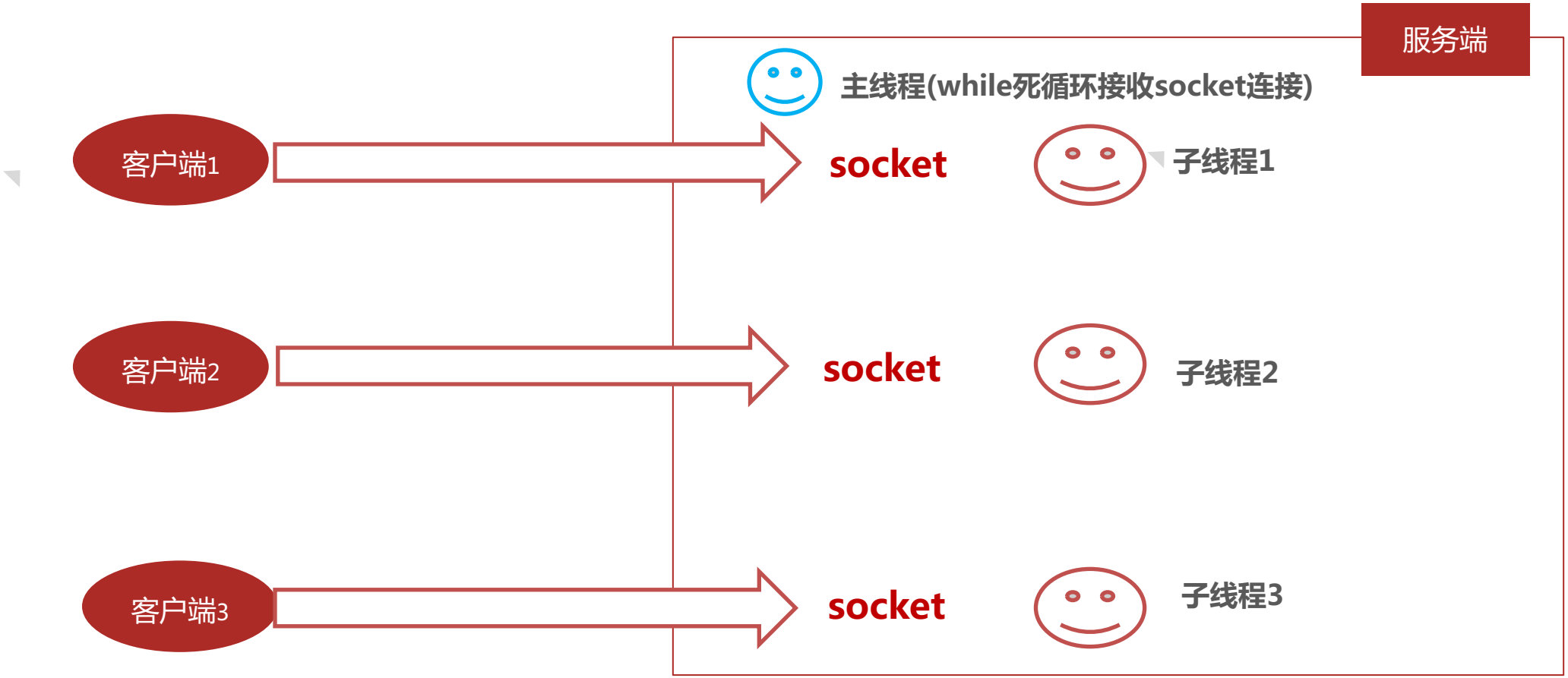
1、之前我们的通信是否可以同时与多个客户端通信，为什么？

- **不可以的**
- **单线程每次只能处理一个客户端的Socket通信**

2、如何才可以让服务端可以处理多个客户端的通信需求？

- **引入多线程。**

同时处理多个客户端消息



总结

1. 本次是如何实现服务端接收多个客户端的消息的。

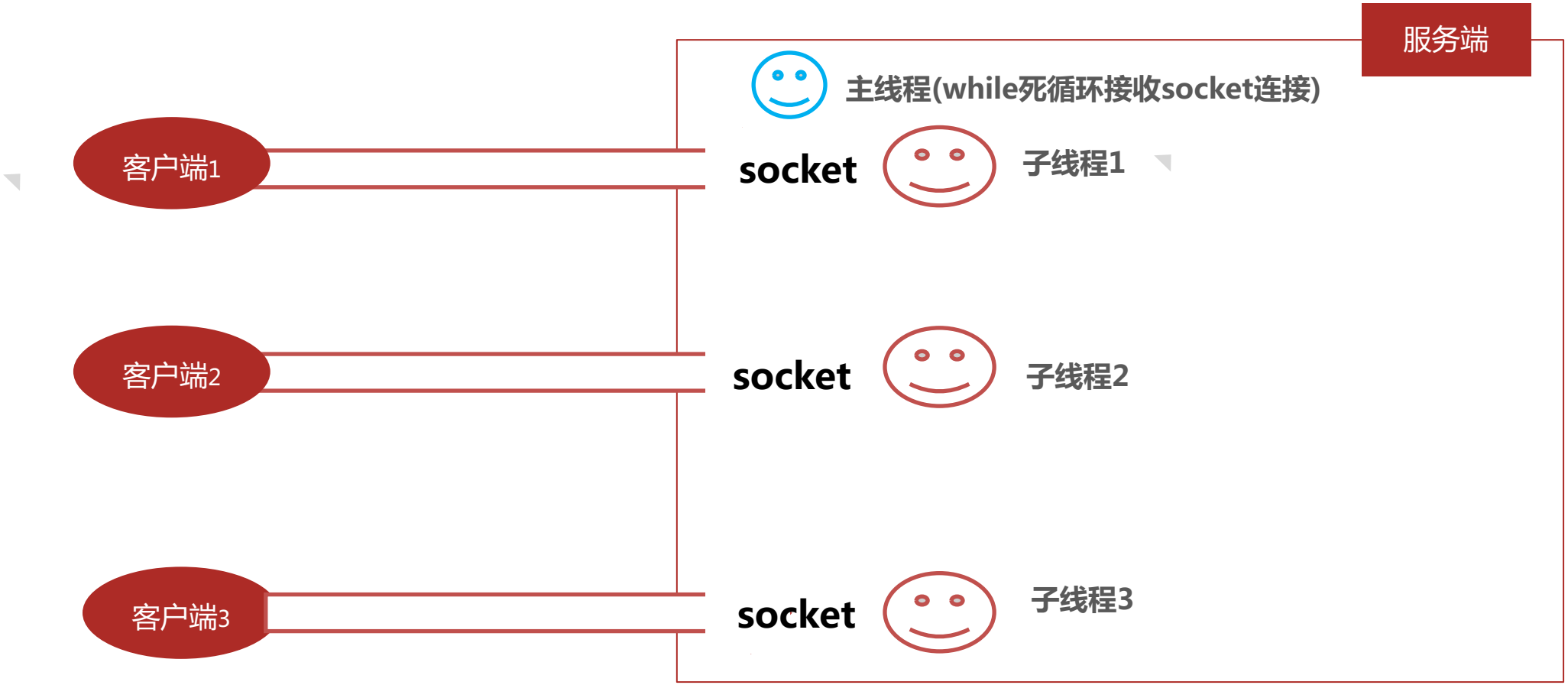
   - **主线程定义了循环负责接收客户端Socket管道连接**
   - **每接收到一个Socket通信管道后分配一个独立的线程负责处理它。**

目录

Contents

- 网络通信三要素
- UDP通信-快速入门
- UDP通信-广播、组播
- TCP通信-快速入门
- TCP通信-多发多收消息
- TCP通信-同时接受多个客户端消息
- TCP通信-使用线程池优化
- TCP通信实战案例-即时通信
- TCP通信实战案例-模拟BS系统

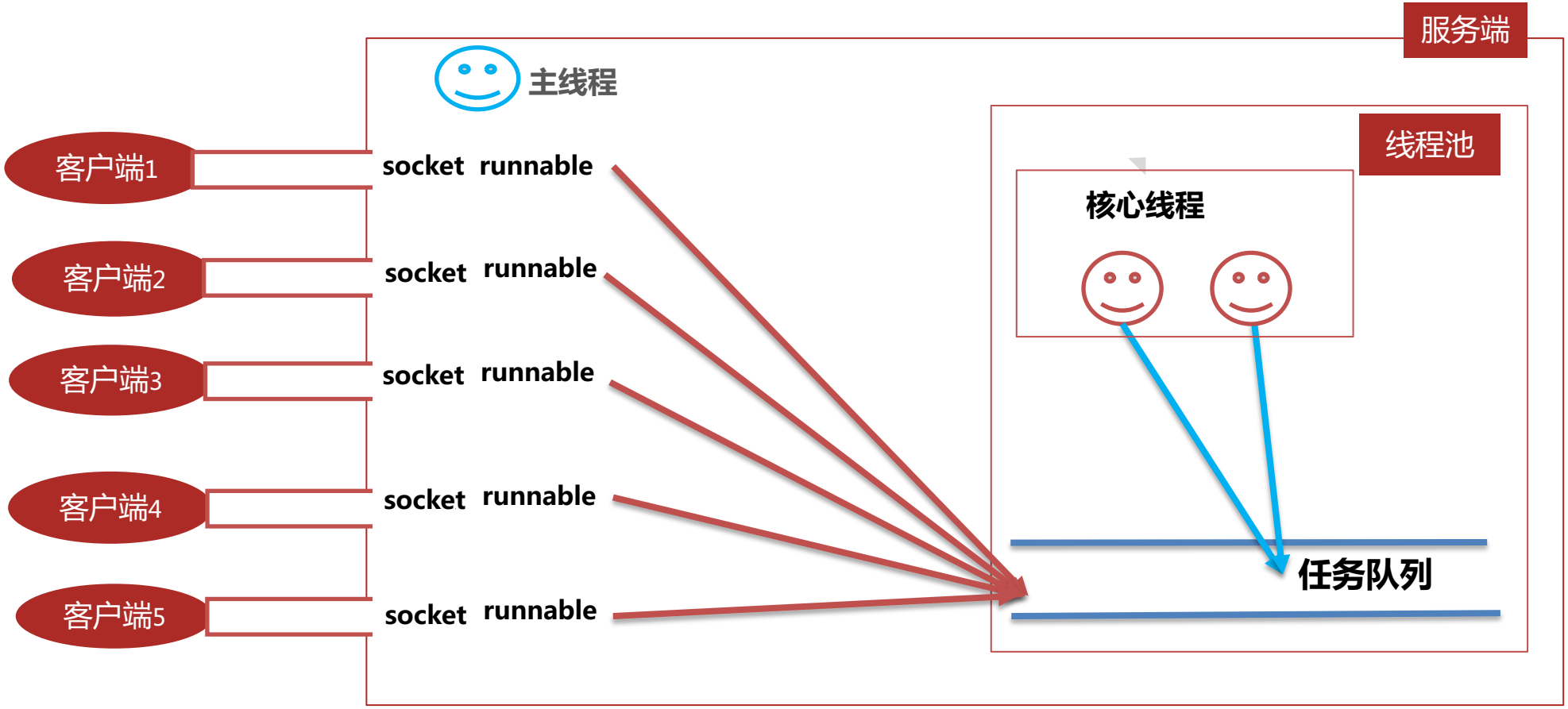
目前的通信架构模型

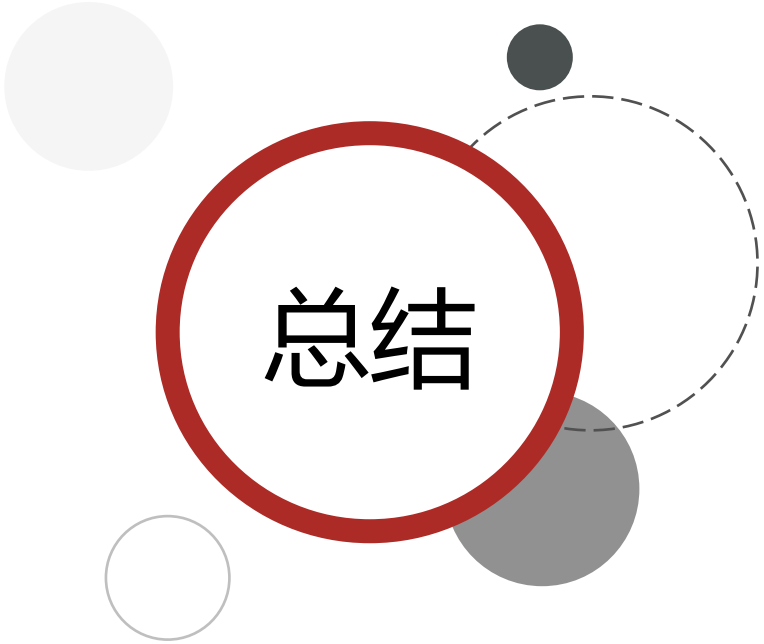

服务端
主线程(while死循环接收socket连接)
客户端1
socket
子线程1
客户端2
socket
子线程2
客户端3
socket
子线程3

1、目前的通信架构存在什么问题？

- **客户端与服务端的线程模型是： N-N的关系。**
- **客户端并发越多，系统瘫痪的越快。**

引入线程池处理多个客户端消息





1. 本次使用线程池的优势在哪里？

- **服务端可以复用线程处理多个客户端，可以避免系统瘫痪。**
- **适合客户端通信时长较短的场景。**

目录

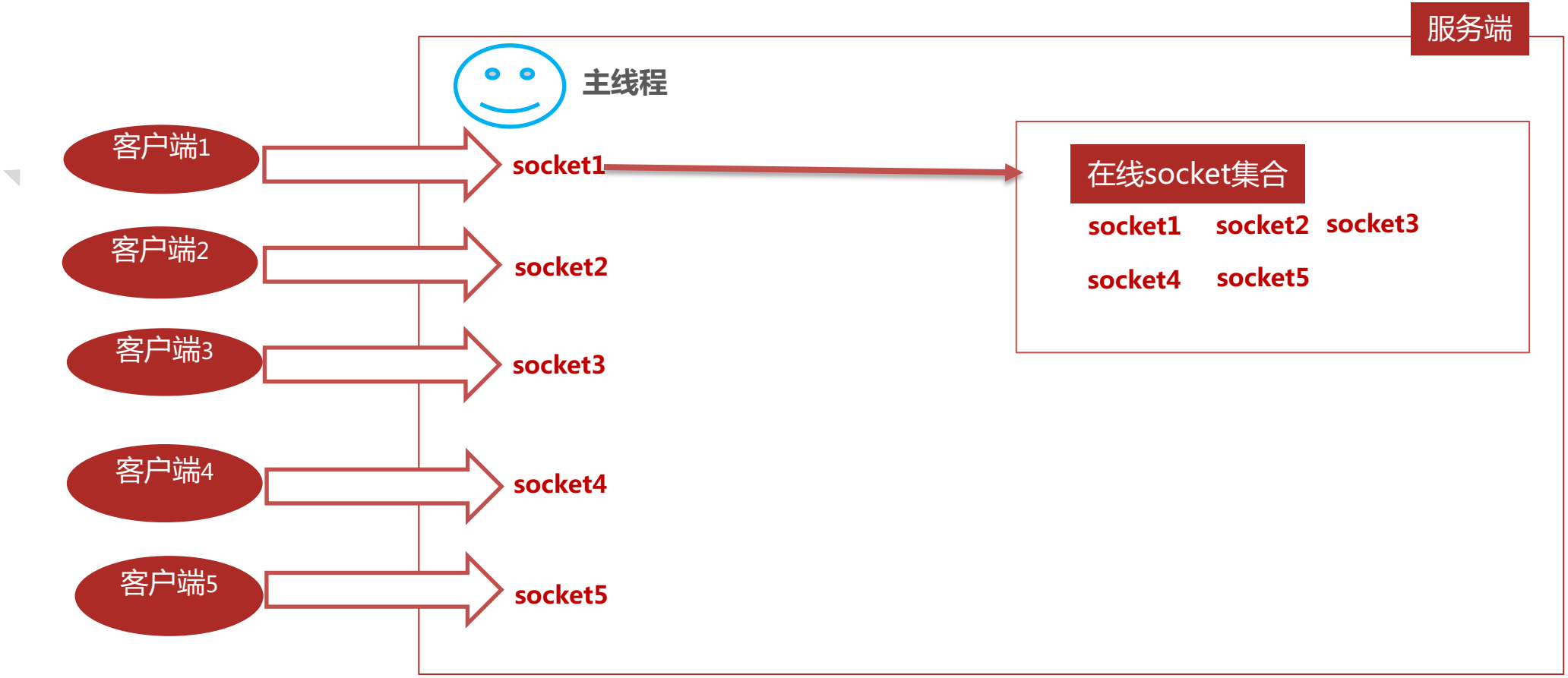
Contents

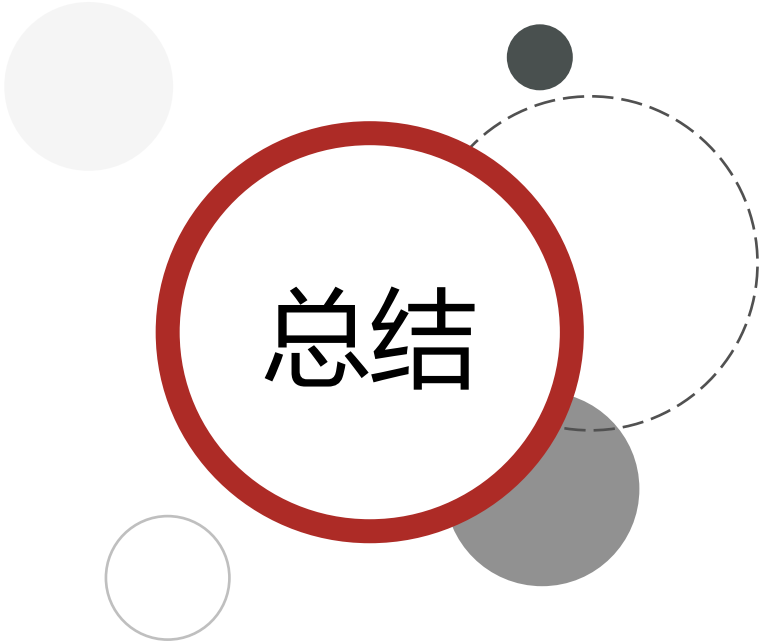
- 网络通信三要素
- UDP通信-快速入门
- UDP通信-广播、组播
- TCP通信-快速入门
- TCP通信-多发多收消息
- TCP通信-同时接受多个客户端消息
- TCP通信-使用线程池优化
- TCP通信实战案例-即时通信
- TCP通信实战案例-模拟BS系统

1、即时通信是什么含义，要实现怎么样的设计？

- **即时通信，是指一个客户端的消息发出去，其他客户端可以接收到。**
- **之前我们的消息都是发给服务端的。**
- **即时通信需要进行端口转发的设计思想。**

即时通信-端口转发


服务端
主线程
客户端1
客户端2
客户端3
客户端4
客户端5
socket1
socket2
socket3
socket4
socket5
在线socket集合
socket1
socket2
socket3
socket4
socket5



1. 即时通信是什么含义，要实现怎么样的设计？

- **即时通信，是指一个客户端的消息发出去，其他客户端可以接收到**
- **即时通信需要进行端口转发的设计思想。**
- **服务端需要把在线的Socket管道存储起来**
- **一旦收到一个消息要推送给其他管道**

目录

Contents

- 网络通信三要素
- UDP通信-快速入门
- UDP通信-广播、组播
- TCP通信-快速入门
- TCP通信-多发多收消息
- TCP通信-同时接受多个客户端消息
- TCP通信-使用线程池优化
- TCP通信实战案例-即时通信
- TCP通信实战案例-模拟BS系统[了解]

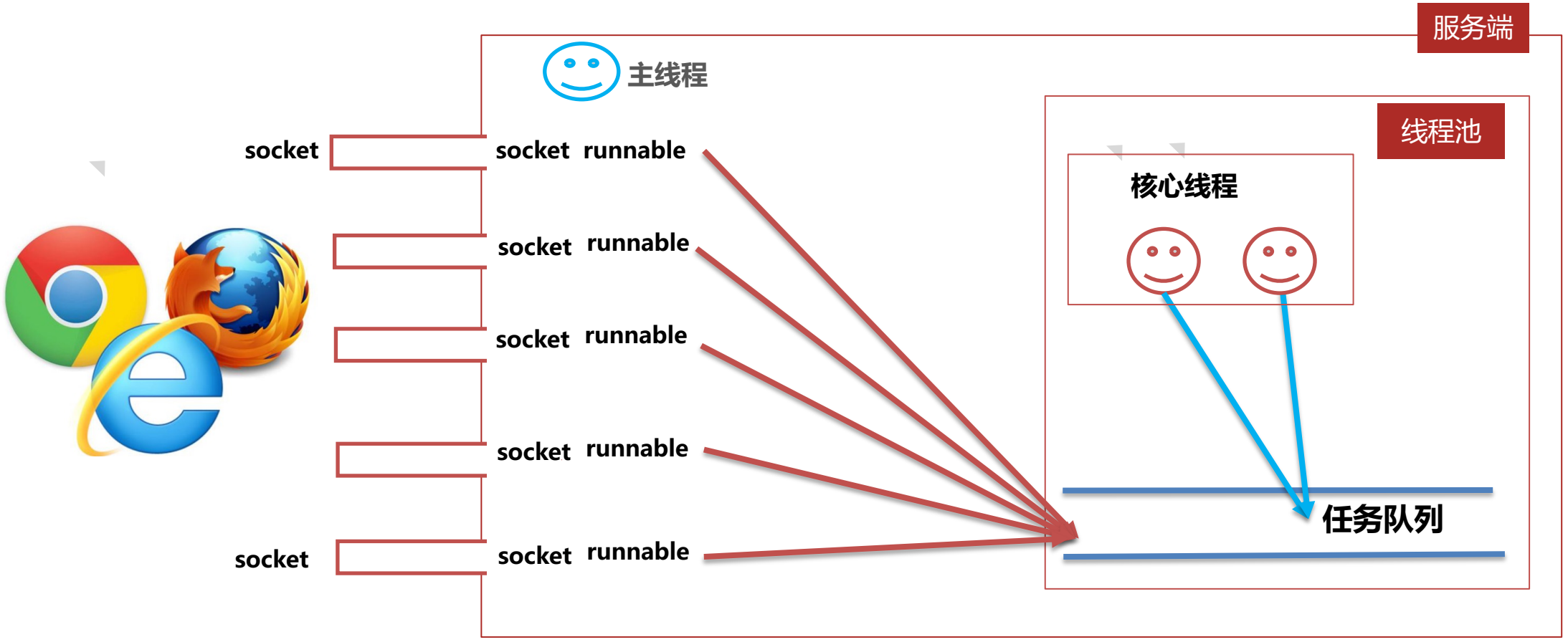
1、之前的客户端都是什么样的

- **其实就是CS架构，客户端实需要我们自己开发实现的。**

2、BS结构是什么样的，需要开发客户端吗？

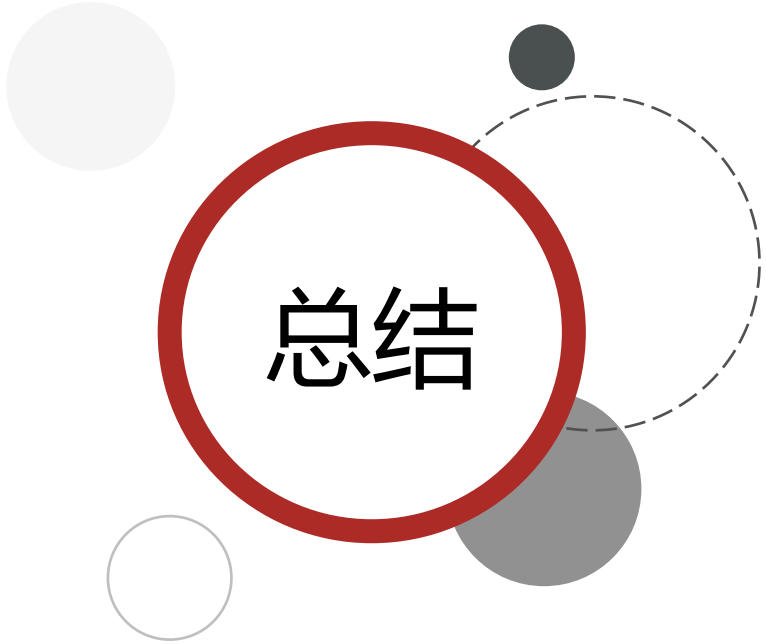
- **浏览器访问服务端，不需要开发客户端。**

实现BS开发



注意：服务器必须给浏览器响应HTTP协议格式的数据，否则浏览器不识别。

HTTP响应数据的协议格式：就是给浏览器显示的网页信息



1. TCP通信如何实现BS请求网页信息回来呢？
   - **客户端使用浏览器发起请求（不需要开发客户端）**
   - **服务端必须按照浏览器的协议规则响应数据。**
   - **浏览器使用什么协议规则呢？**
   - **HTTP协议（简单了解下）**

传智教育旗下高端IT教育品牌