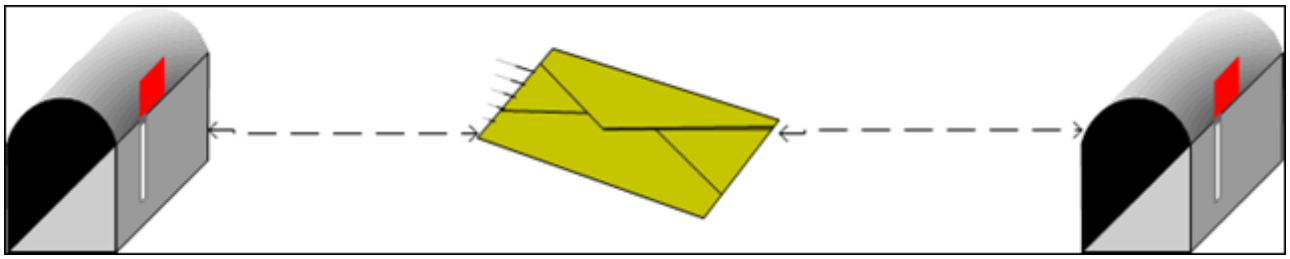
UDP

笔记本：	网络编程		
创建时间：	2018/5/10 10:21	**更新时间：**	2018/5/10 14:53
作者：	ly		
标签：	DGRAM, UDP		

UDP

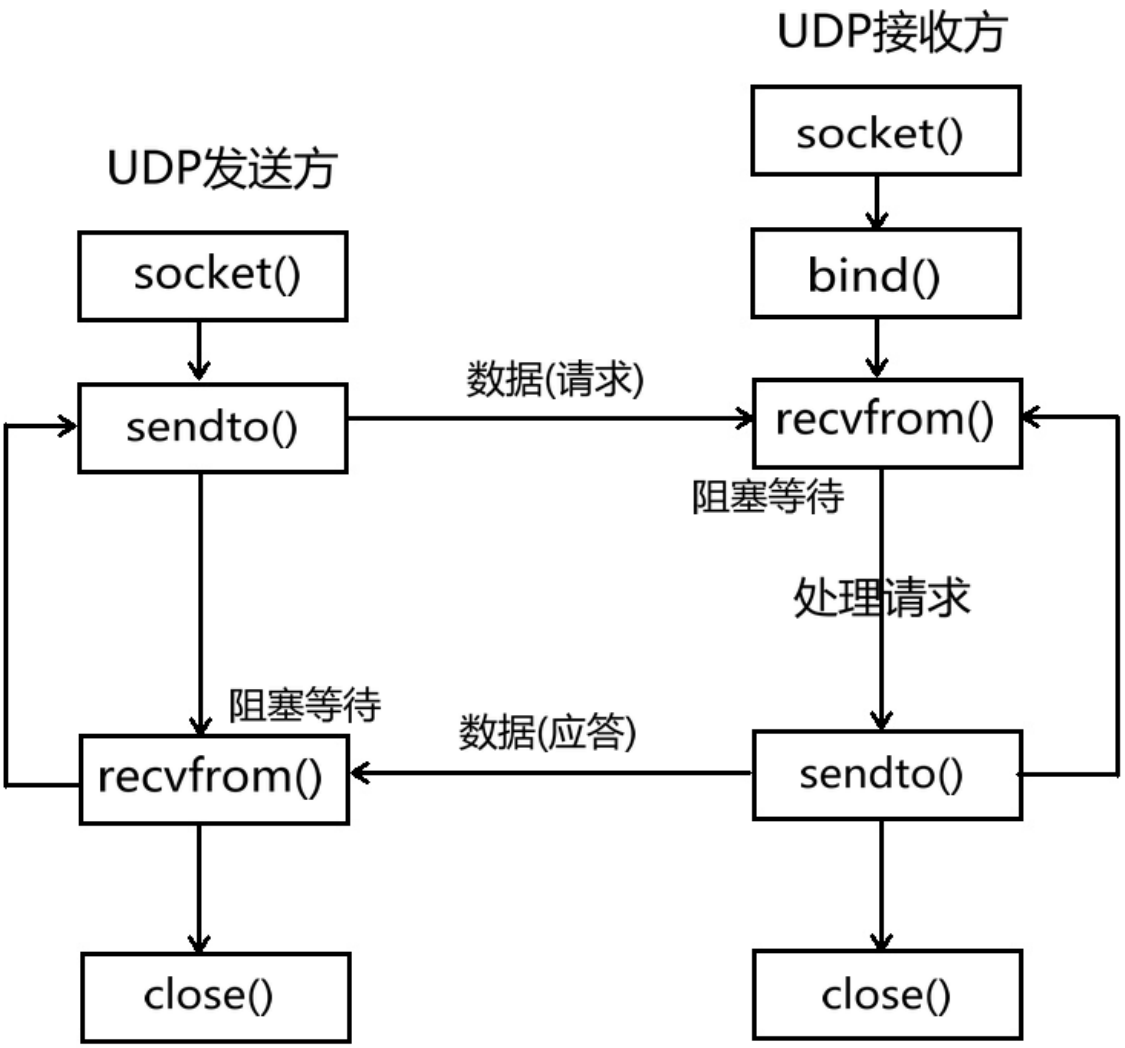
UDP介绍：

UDP 是 User Datagram Protocol 的简称， 中文名是用户数据报协议，是一个简单的面向数据报的运输层协议，在网络中用于处理数据包，是一种无连接的协议。



UDP 不提供可靠性的传输，它只是把应用程序传给 IP 层的数据报发送出去，但是并不能保证它们能到达目的地。由于 UDP 在传输数据报前不用在客户和服务器之间建立一个连接，且没有超时重发等机制，故而传输速度很快。

UDP通信模型



UDP编程：

发送方(sendto(数据，（ip, port)))

```
import socket   # 导入模块

udp = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)    # 创建套接字 udp是自己定义的变量名

senData = "hello"      # 发送的内容

add = ("192.168.25.44", 8080)   # 目的IP和端口    网络地址以元组的方式传入 ip为字符串  port为整型

udp.sendto(senData.encode("utf-8"), addr)   # udp.sendto(内容数据, ip地址和端口)  以元组的方式传送

udp.close()   # 关闭套接字
```

注意点：
在发送方的代码中，我们只设置了目的IP、目的端口
发送方的本地 ip、本地 port 是我们调用 sendto 的时候系统底层自动给客户端分配的。分配端口的方式为随机分配，即每次运行系统给的 port 不一样。

接收方（recvfrom(1024)）

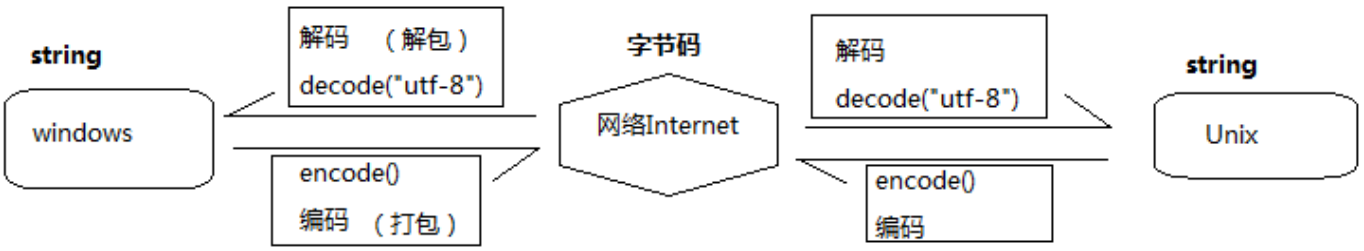
条件（本地）：确定的ip地址；确定的端口port.

接收方使用bind()函数，来完成地址和套接字的绑定，固定本地的ip、port，以便发送方直接指定接收的ip和port。

```
import socket    # 导入模块

udp = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)    # 创建套接字

localAddr = ("", 8080)   # 绑定本地对客户端开放端口，ip一般不写，代表本机的任何一个ip

udp.bind(localAddr)    # 绑定ip、port

# 接收数据，右边以元组的方式返回   (数据, (ip, port))
recvData = udp.recvfrom(1024)    # 1024表示本次接收的最大字节数

data, addr = recvData   # 元组解包

print("接收到的数据：", data, "ip和port：", addr)

udp.close()    # 关闭套接字
```

recvfrom 为**阻塞**式函数：没有客户端给它发送数据，程序会一直停在recvfrom等待。

Python转码问题



解释：
常用编码：utf-8国际编码、GBK中文、GBK2312简体中文编码。编码名不区分大小写，括号不写默认是utf-8的
字符串通过编码成为字节码，字节码通过解码编程字符串
网络上传输只能是字节码：发送数据时，必须转换为字节码；从网络上接收的数据，也是字节码

乱码：英文不存在乱码
中文乱码：Liunx和Unix需要用"utf-8"显示，windows需要用"GBK"显示。

应用：udp单次发送和接收

```
import socket    # 导入模块

udp = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)    # 创建套接字

sendData = "你好"    # 要发送的数据
```

```
udp.sendto(sendData.encode("GBK"), ("192.168.25.44", 8888))    # 发送数据给对方

data, addr = udp.recvfrom(1024)    # 等待接收对方数据和地址信息

str = data.decode("GBK")    # 解码

ip, port = addr    # 地址解包

print("对方发送的数据：", str)
print("对方地址：", ip, port)

udp.close()    # 关闭套接字
```

发送、接收应用：echo服务器 （回声：对方发送过来什么内容，就回复给对反什么内容）

```
import socket

udp = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)

udo.bind(("", 8081))    # 外括号为bind() 函数的调用括号， 内层括号为必须以元组的方式传入

i = 1
while True:
    recvData, recvaddr = udp.recvfrom(1024)
    udp.sendto(recvData, recvaddr)
    print("第%d次通讯的内容为%s" % (i,recvData.decode("utf-8")))
    i += 1

udp.close()
```

接收应用：聊天室（一直等待接收数据）

```
import socket

def main():

    udp = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)

    udp.bind(("", 8083))

    while True:

        recvInfo = udp.recvfrom(1024)

        print()

if __name__ == "__main__":
    main()
```

UDP案例：简单聊天室（根据需求输入要执行发送或者接收）

```
import socket


def sendMsg(udp):
    """发送数据"""
    sendIP = input("请输入要发送对方的IP:")
    sendPort = int(input("请输入要发送对方的port:"))
```

```
    sendbuf = input("请输入要发送的内容：")

    udp.sendto(sendbuf.encode("utf-8"), (sendIP, sendPort))

    print("发送成功")


def recvMsg(udp):
    """接收数据"""
    recvData,recvIP = udp.recvfrom(1024)
    ip, port = recvip
    print("接收到的数据为:", recvData.decode(), "<<<<<", ip, port)


def main():

    udp = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)

    port = 8888

    udp.bind(("", port))

    print("正在使用的端口是:8888")

    while True:
        print("1,发送数据")
        print("2,接收数据")
        num = input("请输入功能对应的数字:")

        if num == "1":
            sendMsg(udp)
        elif num == "2":
            recvMsg(udp)
        else:
            print("输入错误，请重新输入")
```