

# 第8章 网络编程

## 主要内容

- 网络通信模型
- **Socket**编程
- **Interne**编程
- 应用实例

## ===网络通信模型===

# 问题的引入

寄件人信息

收件人信息

**中国邮政 快递包裹**  
CHINA POST Express Parcel

客服电话: 11183  
网址: www.11183.com.cn

条形码: 9610092323762

**1 寄件人信息 Sender information**

寄件人 Sender 电话/手机 Tel  
寄件单位 From 客户代码 Customer Code  
地址 Address  
客户单号 Customer NO. 邮政编码 Post Code

**2 收件人信息 Consignee Information**

收件人 Consignee 电话/手机 Tel  
收件单位 To 客户代码 Customer Code  
地址 Address  
邮政编码 Post Code

**3 邮件详细说明 Shipment Information**

总件数 Amount of Pieces	实际重量(千克) Actual Weight	计费重量(总重量 千克) Charged Weight (kg)	总体积 Dimensions	长 X 宽 X 高 cm L W H

内件品名 Description of Goods

保价: ☐ 是 ☐ 否 声明价值: 万 仟 佰 拾 元 (¥ 元)  
请确定寄递物品单件价值不超过2万元, 贵重物品务必保价!  
Please ensure the value per shipment is no higher than 20,000 yuan. Valuables must be insured!

**4 附加服务 Additional Service**

☐ 实物返单 ☐ 电子返单  
☐ 其它 Other

**5 代收货款 C.O.D.**

万 仟 佰 拾 元

**6 寄递费用 Charge**

邮费: 元 保价费: 元  
封装费: 元 其他费用: 元  
费用合计: 元

**7 收件人付费 Payment of Consignee**

☐ 收件人付  
应收收件人寄递费 元

**8 揽投员信息 Courier Information**

收寄人员 Pickup by 投递人员 Delivered by

**9 目的地 Dest**

**10 寄件人签署 Shipper's Signature**

请仔细阅读背面契约条款! 承运人已尽说明义务, 您的签名意味着您理解并同意接受条款的一切内容。  
The shipper agrees to the terms & conditions on the reverse side of waybill!

签名 Signature  
年 月 日 时  
Y M D H

**11 收件人签收 Consignee's Signature**

签名 Signature  
证件号码 Consignee's ID  
年 月 日 时  
Y M D H

收寄局名 Office of Origin

备注 Remarks

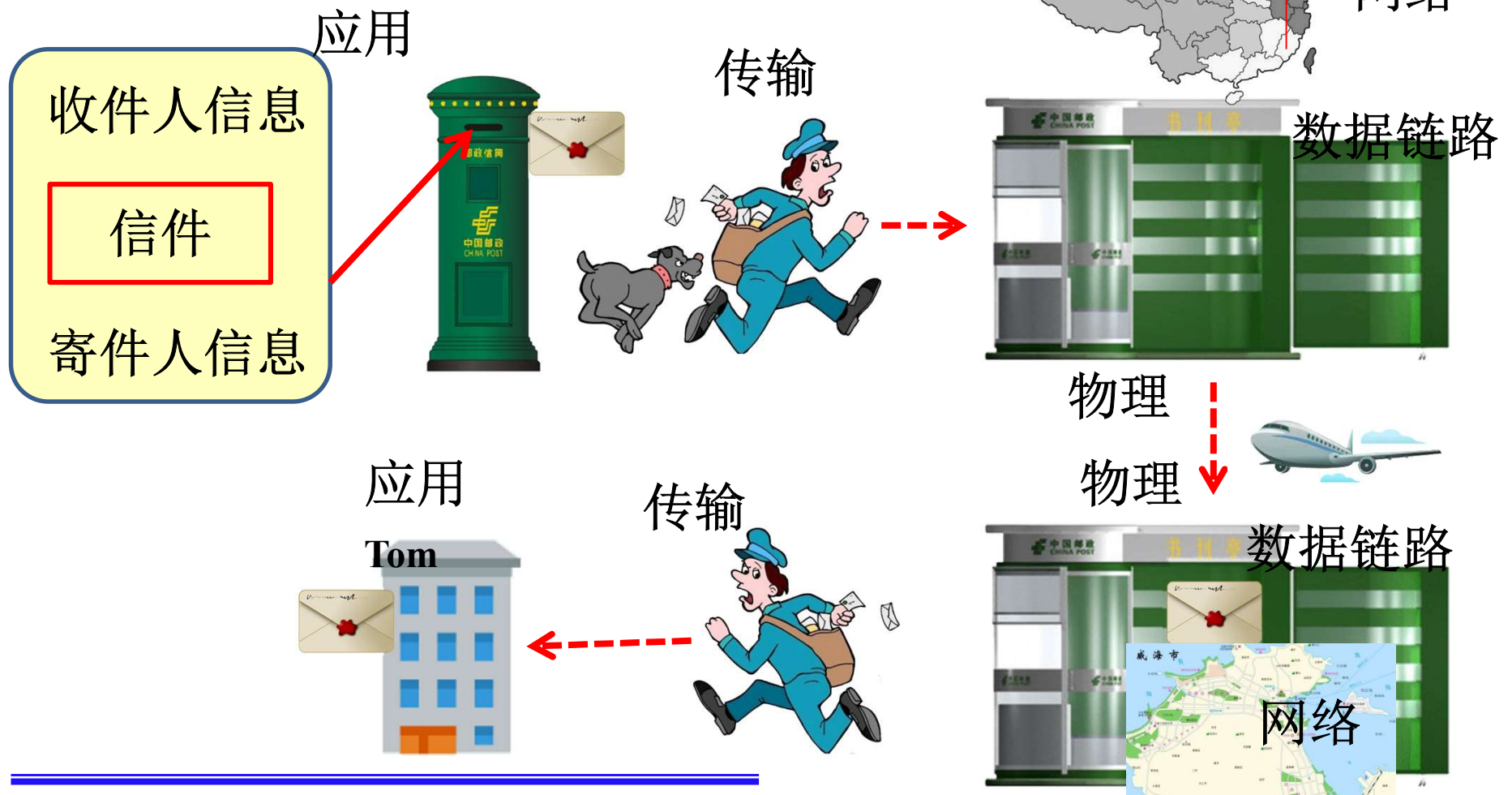
9610092323762

请用力填写! PRESS HARD!

包裹内容及其它信息

# ===网络通信模型===

## ➤ 问题的引入

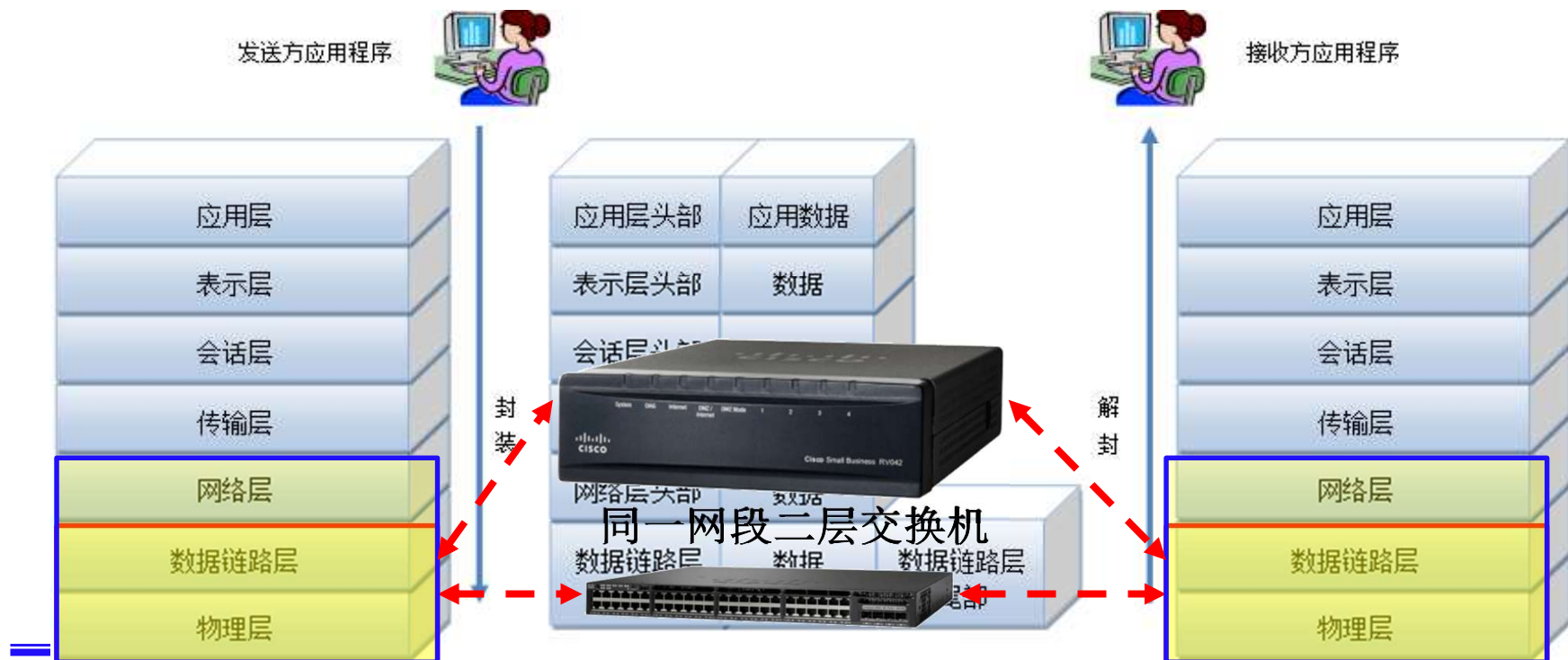


## ===网络通信模型===

## ➤ 计算机网络

将地理位置不同且具有独立功能的多台计算机及其外部设备通过通讯线路连接起来，并在网络操作系统、网络管理软件及网络通讯协议的管理和协调下，实现资源共享和信息传递的计算机系统。

## ➤ OSI参考模型



## ===网络通信模型===

## ➤ PDU在OSI参考模型中的特定名称

- ◆ PDU(Protocol Data Unit) 对等协议之间交换的信息单元统称为协议数据单元

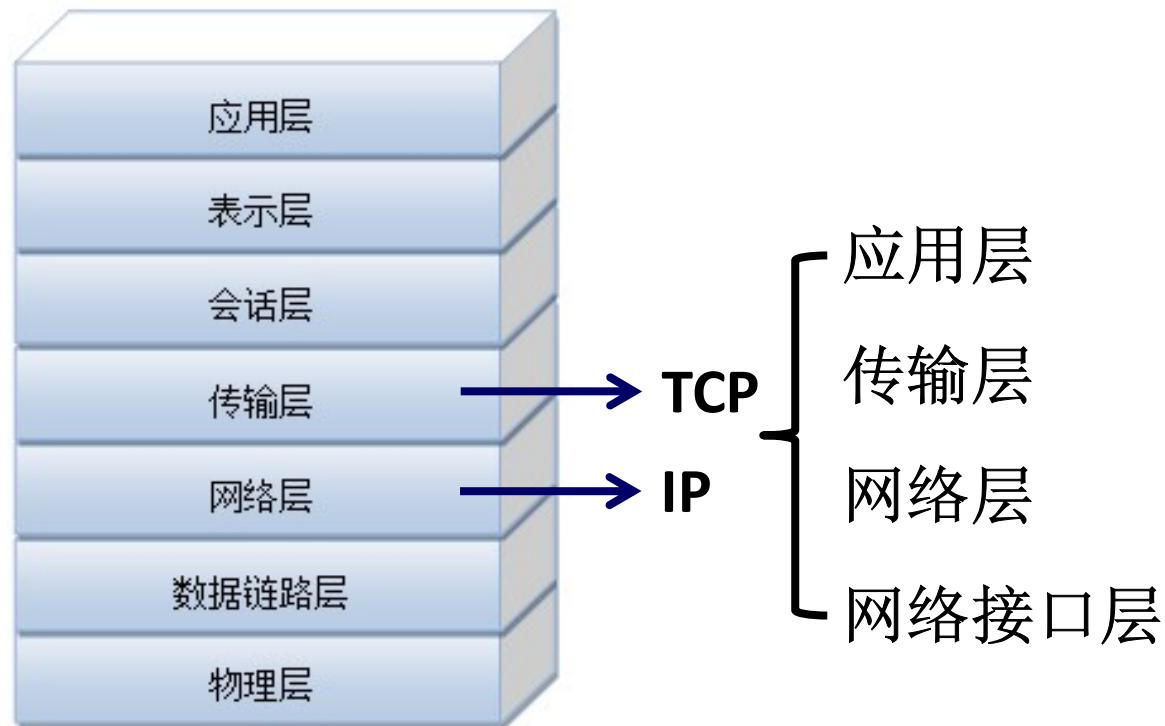
OSI参考模型中的层次	PDU的特定名称
传输层	数据段 (Segment)
网络层	数据包 (Packet)
数据链路层	数据帧 (Frame)
物理层	比特 (Bit)



## ===网络通信模型===

## ➤ TCP/IP协议簇体系结构

◆ TCP/IP是Internet的基础网络通信协议



## ===网络通信模型===

## ➤ TCP/IP协议簇体系结构

## ◆ TCP/IP是Internet的基础网络通信协议

OSI 参考模型		TCP/IP 协议簇	
应用层		应用层	FTP、Telnet SMTP、SNMP、NFS
表示层			
会话层			
传输层		传输层	TCP、UDP
网络层		网络层	IP、ICMP、ARP、RARP
数据链路层		网络接口层	Ethernet 802.3、Token Ring 802.5、X.25、Frame relay、 HDLC、PPP
物理层		未定义	

应用层	“你好，18级同学”			
传输层	“你好，18级同学”	TCP头部		
网络层	“你好，18级同学”	TCP头部	IP头部	
数据链路层	“你好，18级同学”	TCP头部	IP头部	帧头部

# 第8章 网络编程

## 主要内容

- 网络通信模型
- **Socket**编程
- **Interne**编程
- 应用实例



## === Socket编程===

## ➤ Socket基本概念

## ◆ 定义

计算机之间进行网络通信的一套程序接口,网络编程的标准

## ◆ 出版单位

**Berkeley**

## ◆ 作用

Socket对象是网络通信的基础,相当于一个管道连接了发送端和接收端

## ◆ 基础协议

TCP和UDP



## === Socket编程===

## ➤ 基于UDP协议的网络编程

## ◆ 基础函数介绍

(1) `socket()`: 创建一个流式套接字, 返回套接字号。

**`socket(family,type[,protocol])`**

**Family:** 地址家族, 取值为AF\_INET(IPv4)、AF\_INET6(IPv6)。

**type:** 套接字类型, 取值为SOCK\_STREAM或SOCK\_DGRAM。

**SOCK\_STREAM:** TCP通讯的流式套接字

**SOCK\_DGRAM:** UDP通讯的数据报套接字。

**protocol:** 与特定的地址家族相关的协议。默认值为0。

(2) `bind()`: 将套接字绑定到一个已知的地址及端口号, 通常为本地IP地址。

**`bind(address)`**

(3) `sendto()`: 发送数据。

**`sendto(data,(addr, port))`**

## === Socket编程 ===

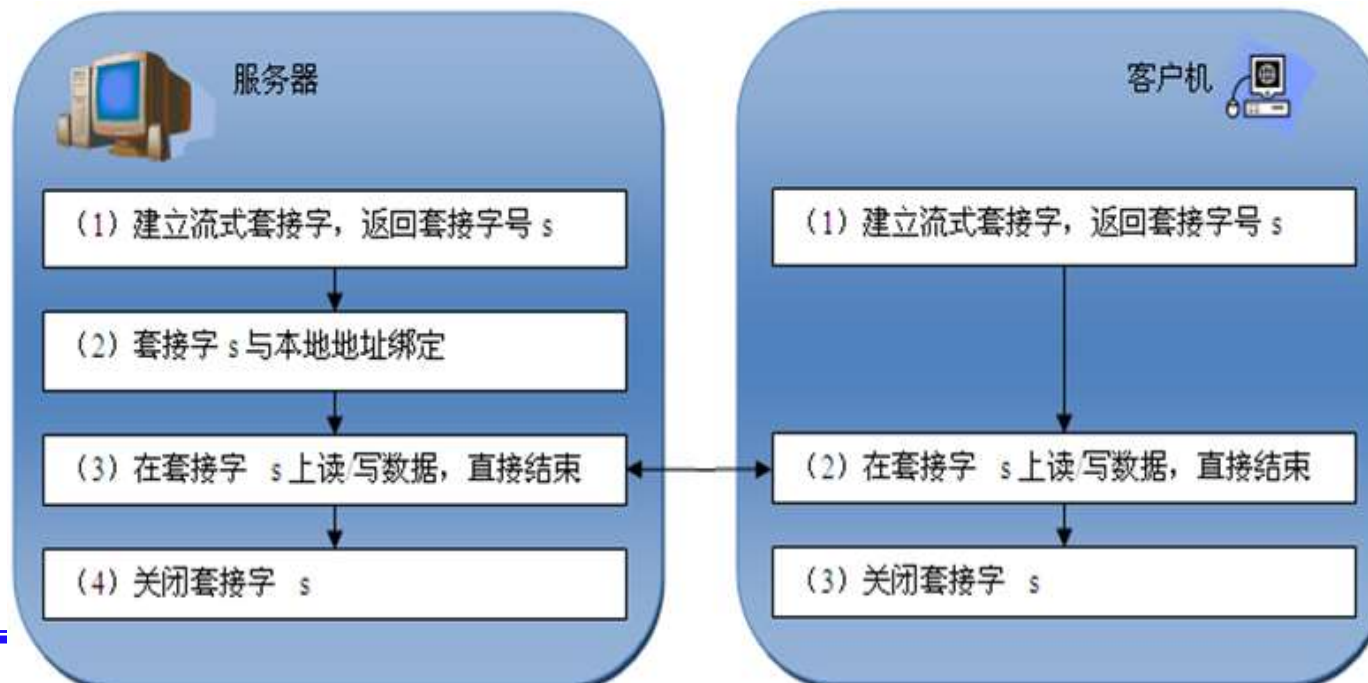
## ➤ 基于UDP协议的网络编程

## ◆ 基础函数介绍

(4) recvfrom(): 接收数据。

data, addr = s.recvfrom(bufsize)

【例8-1】编写一个基于UDP通讯程序，实现两台机器之间的信息通讯。



```
import socket
s=socket.socket(socket.AF_INET,
                 socket.SOCK_DGRAM)
s.bind(('127.0.0.1',5005))
while True:
    data,addr=s.recvfrom(1024)
    if not data:
        print('client has exited!')
        break
    print('received:',data,
          ' from ',addr)
s.close()
```

## ➤ 基于UDP协议的网络编程

```
import socket
s=socket.socket(socket.AF_INET,
                 socket.SOCK_DGRAM)

port=5005
host='127.0.0.1'
while True:
    msg=input()
    if not msg:
        break
    s.sendto(msg.encode("utf-8"),
             (host,port))
s.close()
```

## === Socket编程===

【例8-1】编写一个基于UDP通讯程序，实现两台机器之间的信息通讯。

```
from socket import *
s=socket(AF_INET,SOCK_DGRAM)
s.bind(('',5005))
try:
    while True:
        data,ipaddr=s.recvfrom(1024)
        if not data or data.decode()=='-1':
            print('break')
            break
        print(data.decode()) #解码
except Exception as e:
    print("发生异常，终止服务！ ",e) #捕获异常
s.close()#关闭socket对象
s.close()
```

## === Socket编程 ===

## ➤ 基于TCP协议的网络编程

## ◆ 基础函数介绍

## □ 服务器端

(1) `s.listen(backlog)`: 开始监听TCP连接,backlog为网络最大连接数。

(2) `s.accept()`:接受TCP连接并返回(conn, address), conn套接字对象, 用于接收和发送数据, address是连接客户端的地址

## □ 客户端

`s.connect(address)`: 连接到address处。Address=(host, port)

## □ 公有

`s.recv(bufsize[,flag])`:接收TCP数据, 以字符串形式返回, bufsize表示接收的最大数据量。

`s.sendall(bytes[,flag])`:完整发送TCP数据。

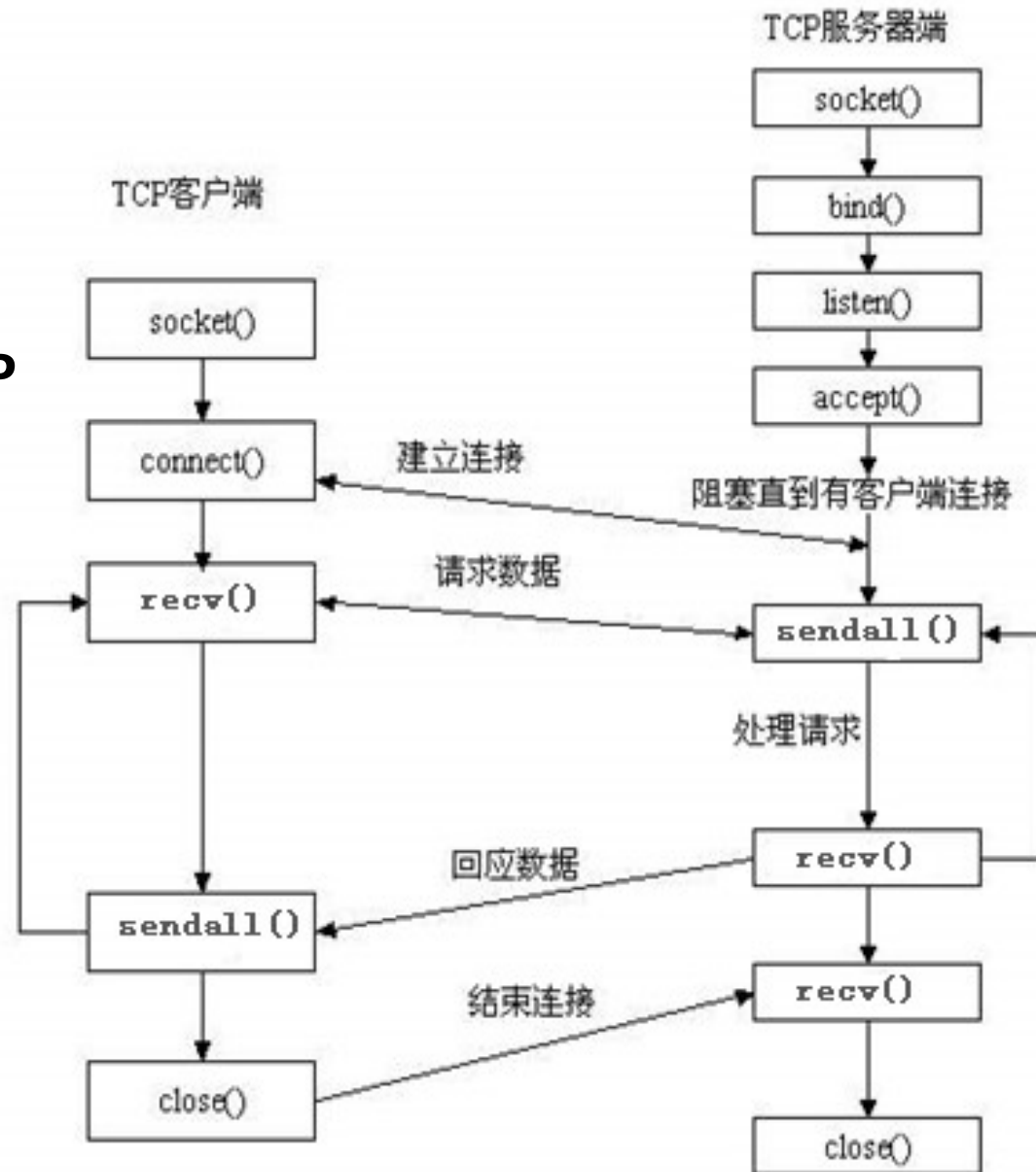
`s.settimeout(timeout)`:返回当前超时期的值, 单位是秒。



## === Socket编程 ===

## ➤ 基于TCP协议的I

## ◆ 基于TCP的网络通信

【例8-2】建立基于TCP  
协议的网络通信程序

## === Socket编程 ===

## ➤ 基于TCP协议的网络编程

## ◆ 基于TCP的网络通信编程框架

## 【例8-2】建立基于TCP协议的网络通信程序

```
from socket import *
from time import ctime
host=''
port=4700
Buf=1024
addr=(host,port)
s=socket(AF_INET,SOCK_STREAM,0)
s.bind(addr)
s.listen(20)
```

```
while True:
    print('等待客户连接...\r\n')
    cs,caddr=s.accept()
    print('...连接来自于:',caddr)
    data='欢迎你的到来! \r\n'
    cs.sendall(bytes(data,'utf-8'))
    data=cs.recv(Buf).decode('utf-8')
    if not data:
        break
    data='[%s] %s\r\n'%(ctime(),data)
    cs.sendall(bytes(data,'utf-8'))
    print(data)
    cs.close()
s.close
```

## === Socket编程===

➤ 基于TCP协议的网

◆ 基于TCP的网络通信

【例8-2】建立基于TCP协议的网络通信程序

```
from socket import *
buf=1024
addr=('127.0.0.1', 4700)
cs=socket(AF_INET,SOCK_STREAM,0)
cs.connect(addr)
data=cs.recv(buf).decode('utf-8')
if data:
    print(data)
while True:
    data=input()
    if data:
        cs.sendall(bytes(data,'utf-8'))
        data=cs.recv(buf).decode('utf-8')
        if data:
            print(data)
cs.close()
```

# 第8章 网络编程

## 主要内容

- 网络通信模型
- **Socket**编程
- **Interne**编程
- 应用实例

## === Internet编程 ===

## ➤ Web资源访问

## ◆ 网络协议

超文本传输协议(HTTP)

## ◆ 数据访问方式

通过统一资源定位符 (URL) 请求访问资源

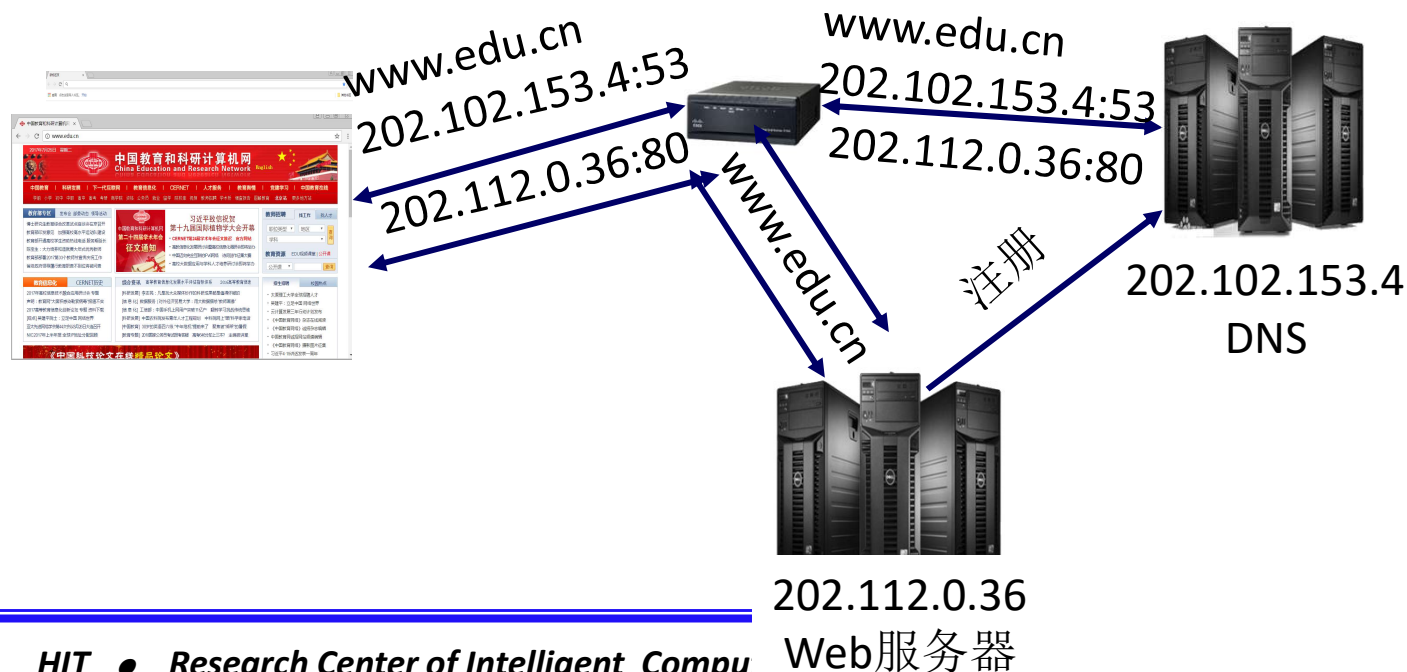
URL:http://www.edu.cn == IP:Port

## ◆ 主要内容

网页—数据展示

## ◆ 客户端软件

浏览器



## === Interne编程 ===

## ➤ Web资源访问

## ◆ 访问Web资源的方法

urllib模块:

urllib.request: 打开和读取url

urllib.parse: 解析URL

urllib.error: urllib.request引发的异常

urllib.robotparser: 解析robots文件

表示以post方式  
提交到url的数据

## (1) 获取资源

urlopen(): 创建一个表示远程URL的类文件对象

urllib.request.urlopen(url[,data[,proxies]])

设置代理

info(): 返回一个httplib.HTTPMessage对象, 表示远  
程服务器返回的头信息

getcode(): 返回HTTP状态码。200, 404

geturl(): 返回请求的url



## === Internet编程 ===

## ➤ Web资源访问

【例8-2】打开一个网站，显示其头部信息、访问状态、网址及网页上前30行信息。

```
import urllib.request
content=urllib.request.urlopen('http://www.edu.cn')
print("http header:",content.info())
print("http status:",content.getcode())
print("url:",content.geturl())
i=0
print('content:')
for line in content.readlines():
    print(line.decode('gb2312'))
    i=i+1
    if i>30:
```

## === Interne编程 ===

## ➤ Web资源访问

## ◆ 访问Web资源的方法

## (2) 保存web资源

```
urllib.request.urlretrieve(url[,filename[,reporthook[,data]]])
```

```
import urllib.request  
content=urllib.request.urlretrieve('http://www.edu.cn',  
'c:\\zxd1.htm')  
print(content)
```

对url进行编码、解码的有用方法

```
urllib.quote(string[,safe]), urllib.unquote(string)
```

```
urllib.quote_plus(string[,safe]),
```

```
urllib.unquote_plus(string),
```

.....

## === Interne编程 ===

## ➤ Web资源访问

## ◆ 访问Web资源的方法

(2) 向Web服务器传递参数（修改版本在小图，加入了头部）

如http://search.jd.com/Search?keyword=平板电脑&enc=utf-8

```
import urllib.parse
import urllib.request
url='http://search.jd.com/Search'
values={'enc':'utf-8','keyword':'平板电脑'}
data=urllib.parse.urlencode(values)
content=urllib.request.urlopen(url+'?' + data)
i=0
for line in content:
    print(line.decode('utf-8'))
    i=i+1
    if i>60:
        break
content=urllib.request.urlretrieve(url+'?' + data,'zxd2.htm')
```

## ➤ Web资源访问

## ◆ 访问Web资源的方法

(2) 向Web服务器传递参数

如http://search.jd.com/Search?keyword=平板电脑&enc=utf-8

```
import urllib.parse
import urllib.request
url='http://search.jd.com/Search'
values={'enc':'utf-8','keyword':'平板电脑'}
data=urllib.parse.urlencode(values)
kv = {'user-agent':'Mozilla/5.0'}
req = urllib.request.Request(url= url+'?' + data, headers=kv)
content=urllib.request.urlopen(req)
i=0
for line in content:
    print(line.decode('utf-8'))
    i=i+1
    if i>60:
        break
content=urllib.request.urlretrieve(url+'?' + data,'zxd2.htm')
```

## ➤ 邮件客户端编程

◆ POP3(Post Office Protocol 3): 负责从邮件服务器中检索电子邮件。支持多用户互联网邮件扩展, 允许在电子邮件上附带二进制文件。是因特网电子邮件的第一个离线协议标准。

POP3(host, post=POP3\_PORT[,timeout]): 构造方法, 连接host邮件服务器。POP3\_PORT=110

user(username): 发送邮件的账号(登录用户名)

pass\_(password): 发送用户密码

getwelcome(): 返回欢迎信息

stat(): 返回邮箱状态, 结果为元组 (邮件数量, 邮件字节数)

list(): 返回邮件列表, 结果是一个三元组

retr(i): 返回第i封邮件(从最早的开始计数), 三元组

dele(i): 设置邮件删除标志, 调用quit()时从服务器删除

quit(): 注销, 释放连接

## === Interne编程 ===

## ➤ 邮件客户端编程

【例8-5】从mail.hit.edu.cn

```
import poplib
user='zxd@hit.edu.cn'
pwd=input('输入密码')
host='mail.hit.edu.cn'
M=poplib.POP3(host)
M.user(user)
M.pass_(pwd)
mboxstat=M.stat()
print("邮件数量",mboxstat[0])
mylist=M.list()
```

```
for i in range(3):
    print(mylist[1][i])
numMessages=len(M.list()[1])
print("邮件数量",numMessages)
k=75
mailx=M.retr(k)[1]
i=0
for line in mailx:
    print(line)
    i=i+1
    if i>=5:
        break
M.quit()
```

## === Internet编程 ===

## ➤ 邮件客户端编程

【例8-6】从mail.hit.edu.cn升级版





## 【例8-6】从mail.hit.edu.cn升级版

题目\发件人\收件人

```
import poplib
from email.parser import Parser
from email.header import decode_header
from email.utils import parseaddr
def decode_str(s):
    value,charset=decode_header(s)[0]
    if charset:
        value=value.decode(charset)
    return value
```

```
def print_info(msg):
    for header in ['From','To','Subject']:
        value=msg.get(header,"")
        if value:
            if header=='Subject':
                value=decode_str(value)
            else:
                hdr,addr=parseaddr(value)
                name=decode_str(addr)
                value=name+'<'+addr+'>'
        print(header+':'+value)
```

编码

```
def guess_charset(msg):
    charset=msg.get_charset()
    if charset is None:
        content_type=msg.get('Content-Type','').lower()
        pos=content_type.find('charset=')
        if pos>=0:
            charset=content_type[pos+8:].strip()
    return charset
```

## 【例8-6】从mail.hit.edu.cn升级版

```
i=0
for part in msg.walk():
    filename=part.get_filename()
    content_type=part.get_content_type()
    print('====',content_type,'====')
    charset=guess_charset(part)
    if filename:
        filename=decode_str(filename)
        data=part.get_payload(decode=True)
        if filename!=None or filename!="":
            print('Accessory:' + filename)
            savefile(filename,data,mypath)
    else:
        email_content_type=""
        content=""
        if content_type=='text/plain':
            email_content_type='text'
        elif content_type=='text/html':
            email_content_type='html'
        if charset:
            content=part.get_payload(decode=
                True).decode(charset)
        print(email_content_type+' '+content)
```

```
email='zxd@hit.edu.cn'
pwd=input("input password")
pop3_server='mail.hit.edu.cn'
mypath='D://pythonemail/'
server = poplib.POP3(pop3_server,110)
print(server.getwelcome())
server.user(email)
server.pass_(pwd)
print('Message:%s. Size:%s'%server.stat())
resp,mails,objects=server.list()
#index=len(mails)
resp,lines,octets=server.retr(76)
lists=[]
for e in lines:
    lists.append(e.decode())
msg_content='\r\n'.join(lists)
msg=Parser().parsestr(msg_content)
print_info(msg)
server.quit()
```

=== **Interne**编程===

## ➤ 邮件客户端编程

## ◆ 使用SMTP协议发送邮件

**SMTP(host="",port=0,local\_h=None[,timeout],src\_addr=Noe),**  
构造函数，其主要参数**host**是发送邮件的服务器

**login(user,password):**登录服务器

**sendmail(from\_addr, to\_addr, msg, mail\_option=[],**  
**rept\_option=[]):** 发送邮件

**send\_message(msg, from\_addr, to\_addr):** 以邮件对象的形式发送邮件。

**quit():**注销，释放连接

=== Interne编程 ===

## 【例8-7】编写一发送邮件的程序

```
import smtplib
import email
def prompt(prompt):
    return input(prompt).strip()
fromaddr=prompt("From:")
toaddrs=prompt("To:")
subject=prompt("Subject:")
print("Enter message, end with continusly enter:")
text=""
while True:
    try:
        line=input()
    except EOFError:
        break
    if len(line)==0:
        break
    text=text+line+"\r\n"
```

## === Internet编程 ===

## ➤ 邮件客户端编程

## 【例8-7】编写一发送邮件的程序

```
msg=email.message_from_string(text)
msg['Subject']=subject
msg['From']=fromaddr
msg['To']=toaddrs
msg.set_charset('GB2312')
server=smtplib.SMTP('123.125.50.133')#202.118.224
.233
user=fromaddr
pwd=input('请输入邮箱密码')
server.login(user,pwd)
server.send_message(msg,fromaddr,toaddrs)
print("Send successfully")
server.quit()
```

# 第8章 网络编程

## 主要内容

- 网络通信模型
- **Socket**编程
- **Interne**编程
- 应用实例



## ➤ 服务器与客户端的交互计算

**【例8-7】** 使用基于TCP的socket套接字模块编写一个网络通信程序，实现客户与服务器的交互计算。其中，客户端将三角形的三边长发给服务器，服务器计算出三角形的面积再把结果返回给客户端。

### □ 问题分析：

- (1) 建立服务器socket对象，等待用户发送请求
- (2) 服务器接收到客户端数据，调用函数进行三角形面积计算，并把计算结果返回给客户端
- (3) 建立客户端socket对象，收集用户输入信息，发送给服务器端

## 【例8-7】基于TCP的socket编写一个网络通信程序

```
from socket import *
from time import ctime
from math import *
def triangle(a,b,c):
    s=(a+b+c)/2
    return sqrt(s*(s-a)*(s-b)*(s-c))
addr=("",5005)
ss=socket(AF_INET,SOCK_STREAM,0)
ss.bind(addr)
ss.listen(20)
```

## 【例8-7】基于TCP的socket编写一个网络通信程序

```
while True:
    print('等待客户连接...\r\n');
    cs,caddr=ss.accept()
    print('连接来自于: ',caddr)
    data='欢迎你的到来! \r\n'
    cs.sendall(bytes(data,'utf-8'))
    data=cs.recv(1024).decode('utf-8')
    if not data:
        break
    print('三角形三边长为:',data)
    sides=data.split(',')
    cs.sendall(bytes(''+str(triangle(float(sides[0]),
                                   float(sides[1]),float(sides[2]))),'utf-8'))
    cs.close()
ss.close()
```

```
from socket import *
s=socket(AF_INET,SOCK_STREAM,0)
s.connect(('127.0.0.1',5005))
data=s.recv(1024).decode('utf-8')
if data:
    print(data)
a=input("请输入边长a:")
b=input("请输入边长b:")
c=input("请输入边长c:")
data=a+','+b+','+c
if data:
    s.sendall(bytes(data,'utf-8'))
    data=s.recv(1024).decode('utf-8')
    if data:
        print('三角形面积为: ',data)
s.close()
```

## ➤ 服务器与客户端的交互计算

【例8-8】使用urllib模块访问某一个网址并判断得到的网页存在多少个网页链接和图片链接。

### □ 问题分析：

- (1) 使用input输入任意一个网址
- (2) 使用urlopen函数打网址，建立应答对象res
- (3) 使用res的read读取网页资源资源
- (4) 使用re模块的compile建立网页链接的正则表达式 “a href=\"(.+?)\”” 的对象linkPattern.
- (5) 使用linkPattern的findall函数获得全部网页链接的列表links
- (6) 使用links的len函数获得网页链接的数量并显示。
- (7) 同样建立 “imgsrc=\"(.+?)\” 对象imagePattern并获得全部链接的images，显示数量

## === Interne编程 ===

【例8-8】使用urllib模块访问某一个网址并判断得到的网页存在多少个网页链接和图片链接。

```
import sys
import urllib.request
import re
url=input("input url:")
res=urllib.request.urlopen(url)
htm=res.read()
details=htm.decode('utf-8')#gb2312
linkPattern=re.compile("a href=\"(.+?)\"")
links=linkPattern.findall(details)
print("网页链接总数=",len(links))
imagePattern=re.compile("img src=\"(.+?)\"")
images=imagePattern.findall(details)
print("图片总数=",len(images))
```

```
import chardet
#用chardet进行内容分析
chardit1 = chardet.detect(htm)
print(chardit1['encoding'])
```

# 本章小结

- 网络通信模型
- **Socket**编程
- **Interne**编程
- 应用实例