全部课程 (/courses/) / 基于TCP的python聊天程序 (/courses/681) / TCP/IP协议简介与聊天程序的初步实现

在线实验,请到PC端体验

一、课程介绍

1. 内容简介

我们将实现基于TCP协议的python聊天程序。我们会使用python的Socket模块进行通讯。

编写简易实现后我们还会使用python的Tkinter进行GUI编程。

2. 课程知识点

通过通过本项目你将会了解到以下知识点:

- TCP/IP协议
- Python网络编程
- PythonGUI编程(Tkinter)

二、实验环境

操作系统: Ubuntu 14.04

所需Python包: Socket、Tkinter

三、实验原理

根据TCP协议,在在两进程之间建立通信。

利用多线程创建GUI与Socket通信模块。

TCP/IP协议

TCP/IP不是一个协议,而是一个协议族的统称。里面包括了IP协议,IMCP协议,TCP协议,以及我们更加熟悉的http、ftp、pop3协议等等。

TCP协议与UDP协议定义在传输层,IP协议定义在网络层。

在之上的应用层定义HTTP协议等。

TCP连接包括三个状态:连接创建、数据传送和连接终止。操作系统将TCP连接抽象为套接字的编程接口给程序使用,并且要经历一系列的状态改变。

下面我们将会使用python的套接字模块进行编程。

四、实验步骤

Python网络编程

传统的TCP编程都是服务器和客户端模式。

客户端只能连接一个服务器,而服务器可以连接多个客户端。

所以,不难猜出对于服务器需要多线程进行处理,分别与多个客户端通信。

下面我们看第一个TCP编程的例子

• 服务器

动手实践是学习 IT 技术最有效的方式!

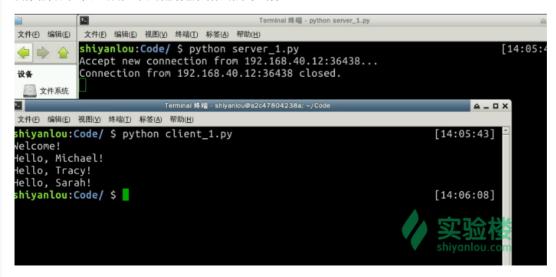
```
#server_1.py
import socket
import time
import threading
def tcplink(sock, addr):
   print 'Accept new connection from %s:%s...' % addr
   sock.send('Welcome!')
   while True:
       data = sock.recv(1024)
       time.sleep(1)
       if data == 'exit' or not data:
           break
       sock.send('Hello, %s!' % data)
   sock.close()
   print 'Connection from %s:%s closed.' % addr
host = socket.gethostname()
port = 12345
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.bind((host, port))
s.listen(5)
while True:
   # 接受一个新连接:
   sock, addr = s.accept()
   # 创建新线程来处理TCP连接:
   t = threading.Thread(target=tcplink, args=(sock, addr))
   t.start()
```

• 客户端

```
#cilent_1.py
import socket

host = socket.gethostname()
port = 12345
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
# 建立连接
s.connect((host, port))
# 接收欢迎消息:
print s.recv(1024)
for data in ['Michael', 'Tracy', 'Sarah']:
# 发送数据:
s.send(data)
print s.recv(1024)
s.send('exit')
s.close()
```

执行结果如下(这里说明一下,需服务器和客户端同时运行):



下面对上面代码进行简单的分析:

通过socket模块进行python网络编程对排标类键是学习学作程外最有效的严责!

server.pv

- i. 通过socket.socket(args)进行创建socket对象,socket.AF_INET表示服务器之间网络通信,socket.SOCK_STREAM表示 流式socket,用于 TCP(还有socket.SOCK_DGRAM是表示数据报式socket,用于 UDP)
- ii. s.bind(args)通过host(主机名/IP)和port(端口)确定绑定的地址
- iii. s.listen(args)开始进行监听连接
- iv. s.accept()接收客户端的连接,一旦接收连接,就返回客户端socket对象和地址用于通信
- v. 创建新线程进行处理,主线程仍然继续监听
- vi. s.send()用来发送消息, s.recv()用于接收消息

client.py

- i. 对于创建socket与服务器相同
- ii. s.connect(args)用于连接某地址的服务器,连接上就可用send和recv来发送和接受消息了
- iii. s.close() 用于关闭连接

下面我们尧根据需要对于上面代码进行改进,用于完成我们的聊天功能

下面命名为Server.py和Client.py,但是在客观上两者不存在服务器和客户端之分,应该均为客户端。

• 客户端一

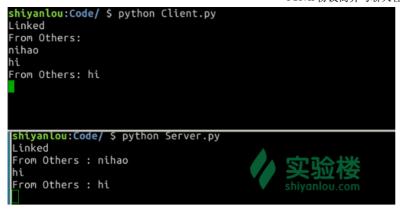
```
#Server.py
import socket
host = socket.gethostname()
port = 12345
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.bind((host, port))
s.listen(1)
sock, addr = s.accept()
print "Linked"
info = sock.recv(1024)
while info != "exit":
   print "From Others : "+info
   send_mes = raw_input("")
   sock.send(send_mes)
   if send_mes == "exit":
       break
   info = sock.recv(1024)
sock.close()
s.close()
```

• 客户端二

```
#Client.pv
import socket
s = socket.socket()
host = socket.gethostname()
port = 12345
s.connect((host, port))
print "Linked"
info = ""
while info != "exit":
   print "From Others: "+info
   send_mes = raw_input("")
   s.send(send_mes)
   if send mes == "exit":
       break
    info = s.recv(1024)
s.close()
```

执行结果如下:

动手实践是学习 IT 技术最有效的方式!



简单解释一下代码逻辑:

双方,一方进行监听,另一方进行连接。双方交互进行输入信息,并发送。当一方输入exit时,双方

的连接断开。程序结束。

而且,使用中会感到别扭,因为这种交流被代码固定。

所以接下来我们将用GUI方式进行完善我们的聊天程序。

思考与改进

尝试用 客户端-服务器-客户端 方式重写聊天程序

下一节 ➤ (/courses/681/labs/2220/document)

课程教师



AlbertWY 共发布过3门课程

查看老师的所有课程 > (/teacher/208579)



动手做实验,轻松学IT





公司

(http://weibo.com/shiyanlou2013)

关于我们 (/aboutus) 联系我们 (/contact)

加入我们 (http://www.simplecloud.cn/jobs.html)

技术博客 (https://blog.shiyanlou.com)

服务

企业版 (/saas)

实战训练营 (/bootcamp/)

会员服务 (/vip)

实验报告 (/courses/reports)

常见问题 (/questions/?

tag=%E5%B8%B8%E8%A7%81%E9%97%AE%E9%A2%98)

隐私条款 (/privacy)

合作

我要投稿 (/contribute) 教师合作 (/labs) 高校合作 (/edu/) 友情链接 (/friends) 开发者 (/developer) 学习路径

Python学习路径 (/paths/python) Linux学习路径 (/paths/linuxdev) 大数据学习路径 (/paths/bigdata) Java学习路径 (/paths/java) PHP学习路径 (/paths/php)

全部 (/paths/)

Copyright @2013-2017 实验楼在线教育 | 蜀ICP备13019762号 (http://www.miibeian.gov.cn/)

动手实践是学习 IT 技术最有效的方式!