

深圳大学实验报告

课程名称： 计算机网络

实验项目名称： Socket 编程

学 院： 计算机与软件学院

专 业： 计算机科学与技术

指导教师： 邹永攀

报告人： 刘睿辰 学号： 2018152051 班级： 数计班

实 验 时 间： 2021 年 3 月 30 日~2021 年 4 月 13 日

实验报告提交时间： 2021 年 4 月 27 日

一、实验目的：

1. 学习网络编程基本概念、InetAddress 的应用、URL 的应用、URLConnection 的应用；
2. 掌握 Socket 的 TCP 通信、Socket 的 UDP 通信。

二、实验环境：

1. 操作系统：Windows 10 操作系统；
2. 编程语言：Anaconda Python 3.6.3；
3. 电脑具有 Internet 连接；具有 WampServer 数据库以及 IDEA java 环境。

三、实验内容：

1. 获取本地机的名称和 IP 地址，并获取以下网站的 IP 地址，若存在多个 IP 地址，则要求全部返回。
网站 1：www.baidu.com
网站 2：www.csdn.net
网站 3：www.google.com
2. 下载深圳大学首页 <https://www.szu.edu.cn/>，并统计下载得到网页文件的大小。要求将深大主页上面的全部对象（图片、文本等）全部下载并保存到本地。
3. TCP 通信部分：
 - 1) 指令交互：客户端向服务器端发送 Time 命令，服务器端接受到该字符串后将服务器端当前时间返回给客户端；客户端向服务器端发送 Exit 命令，服务器端向客户端返回“Bye”后关闭连接退出程序；
 - 2) 文件传输：客户端向服务器端发送文件请求，服务器端接收到请求后，先向其传输文件名，当客户端接收后，再向客户端传输文件。客户端接收文件，并对其改名（文件名可自行定义）和存在本地。
4. UDP 通信部分：
 - 1) 编写完整程序包含一个服务器端程序和多个客户端程序（不同客户端代码几乎一样），使多个客户端之间可以彼此通信聊天（类同微信、QQ）；
 - 2) 客户端之间能够借助用户设计的 UI 界面显示聊天交互过程，并包含上述所有功能。

四、实验步骤：

1. 获取本地机的名称和 IP 地址，并获取网站的 IP 地址，若存在多个 IP 地址，则要求全部返回。
为了得到本地机名称，我们可以用 python 的 socket 库下的 gethostname 函数得到本地机名称。此外为了得到 IP 地址，我们可以使用该函数：gethostbyname(name)函数，参数为本地机名称。事实上，运行这个函数我们可以得到如图 1 所示的结果，可以看到只有一个返回结果。
为了得到全部的 IP 地址，我们可以使用 gethostbyname_ex 函数，这样可以得到全部

的 IP 地址，如图 2 所示。通过和 ipconfig 指令得到的结果来做对比，发现结果是一致的。其中 1/2 行返回的是 VMware Network Adapter VMnet1/8 的 IP 地址，第三行是本机以太网 IP 地址。

```
This computer is DESKTOP-FLKDJEV
Its IP addresses are:
192.168.191.1
```

图 1. gethostbyname 函数运行结果

```
This computer is DESKTOP-FLKDJEV
Its IP addresses are:
169.254.62.172
169.254.33.24
172.29.36.190
```

图 2. gethostbyname_ex 函数运行结果

代码如图 3 所示。由于 IP 地址位于元组的第三位，所以我们要访问的是 hostip[2]。

```
1 import socket
2
3 hostname = socket.gethostname() # 本地主机名
4 hostip = socket.gethostbyname_ex(hostname) # 原始主机名，域名列表，IP地址列表
5 print("This computer is "+str(hostname)+", and its IP address are below")
6
7 for iter in hostip[2]:
8     print(iter)
```

图 3. 得到本机所有 IP 地址的 python 程序

我们还可以用 gethostbyname_ex 方法来获取网站的 IP 地址。我们得到了网站的域名之后，通过该函数可以解析到网站的 IP 地址，如图 4-图 6 所示，我们分别得到了百度、csdn 以及谷歌的所有 IP 地址。

```
please input the domain name:www.baidu.com
The IP address of the domain www.baidu.com are below
14.215.177.39
14.215.177.38
```

图 4. 百度的所有 IP 地址

```
please input the domain name:www.csdn.net
The IP address of the domain www.csdn.net are below
47.95.164.112
```

图 5. csdn 的所有 IP 地址

```
please input the domain name:www.google.com
The IP address of the domain www.google.com are below
162.125.32.12
```

图 6. 谷歌的所有 IP 地址

2. 下载深圳大学首页 <https://www.szu.edu.cn/>，并统计下载得到网页文件的大小。要求将深大主页上面的全部对象（图片、文本等）全部下载并保存到本地。

方法：正则表达式匹配文件名称。

- 1) 首先，我们要明确网址 <https://www.szu.edu.cn/> 的网页文件都有什么，或者它们的类型都是什么样的。所以我们需要检查网页源代码，使用 Fn+F12 之后显示网站

源代码。在 Network 选项可以看到网站的文件格式。例如，在 JS 部分，我们可以看到 JavaScript 文件，如图 7 所示。

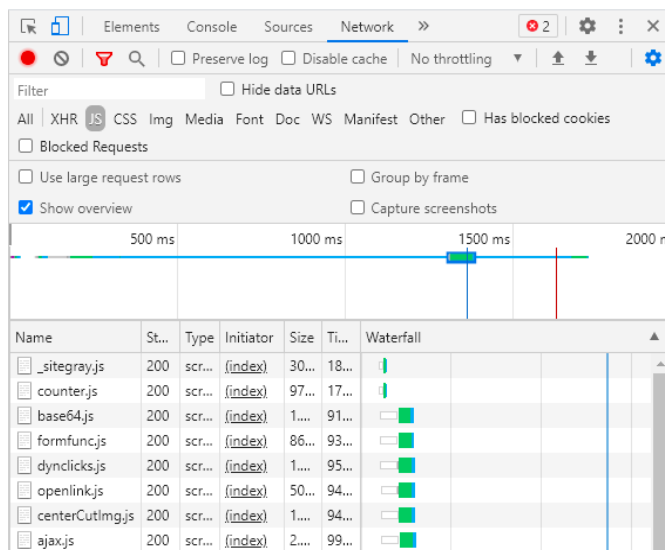


图 7. 网站的 JavaScript 文件

而在 Img 部分我们可以看到图片的文件格式，有 png 和 jpg 格式的图片。点击 CSS 可以看到所有的 css 文件，点击 Media 可以看到 MP4 文件（视频文件），如图 8 所示。

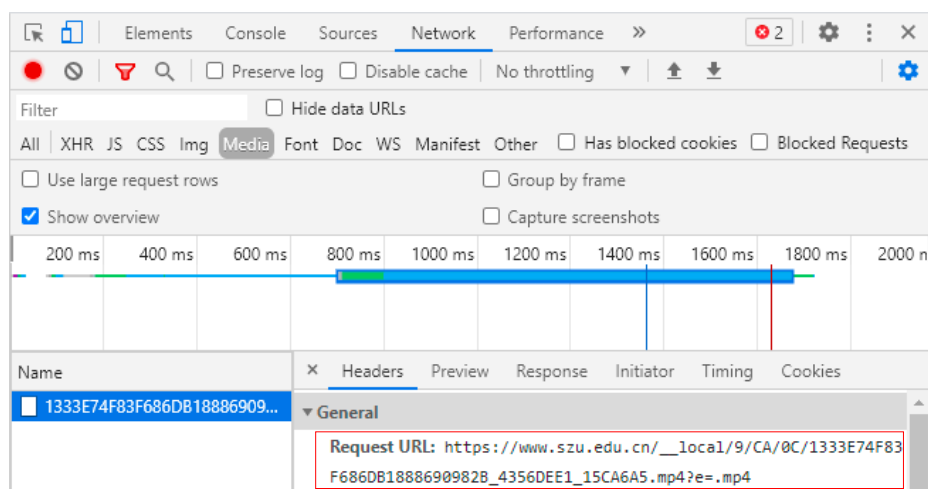


图 8. 网站的 MP4 视频文件

通过对网站的仔细检查，发现资源文件格式基为.jpg（图片）、.png（图片）、.css（层叠样式表）、.js（JavaScript 文件）以及.mp4（视频文件）。现在我们需要将所有的资源文件的 URL 路径保存下来方便我们爬取。如图 8 所示，以视频文件为例我们已经从 Request URL 中得知了 MP4 文件的文件路径，但是后面还有 e=.mp4，所以我们可以将文件格式进行规范化。

以如下网址

`https://www.szu.edu.cn/__local/9/CA/0C/1333E74F83F686DB1888690982B_4356DEE1_15CA6A5.mp4?e=.mp4`

为例，我们想要规范化该网址，所以我们可以先用 python 的 split 方法分离出后面的相对路径

`1333E74F83F686DB1888690982B_4356DEE1_15CA6A5.mp4?e=.mp4`

然后用 python 的 find 方法找到“MP4”首次出现的位置，加上文件后缀名的长度进行截取，就可以得到

1333E74F83F686DB1888690982B_4356DEE1_15CA6A5.mp4

同理，别的文件名我们统一进行标准化，得到标准化代码如图 9 所示。

```
6 def filename_regularized(url):
7     filename = url.split('/')[-1]
8     if ".jpg" in filename:
9         filename = filename[:filename.find(".jpg") + len(".jpg")]
10    elif ".png" in filename:
11        filename = filename[:filename.find(".png") + len(".png")]
12    elif ".css" in filename:
13        filename = filename[:filename.find(".css") + len(".css")]
14    elif ".js" in filename:
15        filename = filename[:filename.find(".js") + len(".js")]
16    elif ".mp4" in filename:
17        filename = filename[:filename.find(".mp4") + len(".mp4")]
18    return filename
```

图 9. 文件名标准化

- 2) 现在我们知道了文件名标准化的方法，那么如何得到所有文件的 URL 地址呢？这里我们必须要用到 python 爬取网站库 requests 中的函数

reply = requests.get(url = url, headers = header)

这里返回一个名为 reply 的响应对象。我们可以从这个对象中获取所有我们想要的信息。url 自然就是我们访问主页的网址，header 是一个请求头，里面对 user-agent（中文名用户代理）进行伪造。从 User-agent 中服务器可以知道客户端的操作系统类型和版本，电脑 CPU 类型，浏览器种类版本，浏览器渲染引擎等等。这是爬虫当中最最重要的一个请求头参数，所以一定要伪造，甚至多个。如果不进行伪造，而直接使用各种爬虫框架中自定义的 user-agent，很容易被封禁。常见的用户代理如图 10 所示。

- User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:52.0) Gecko/20100101 Firefox/52.0
- User-Agent: Mozilla/5.0 (Windows NT 6.3; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/52.0.2743.116 Safari/537.36

图 10. 常见的用户代理

得到了 reply 之后意味着我们想要的资源都在这里。首先我们应该知道网站的编码，然后指定 reply 的编码方式为网站的编码方式。网站的编码方式在源代码 <meta charset>中可以查看，然后获取网页文本信息写入文件即可。

写入文件我们可以用 with open 实现，参数设置为‘w’代表打开一个文件只用于写入。还有很多其他的参数，例如‘r’代表只读，‘w+’代表读写等等。然后制定编码，用 write 函数实现写入什么内容，具体代码如图 11 所示。这里面我们将文本信息写入了 SZUpage.html 文件中，注意该文件保存的位置是与 python 文件一个文件夹的路径地址。

找到文件位置然后打开 SZUpage.html，可以看到结果如图 12 所示。可以看到所有的文字都被保存了下来，但是原网页的所有图片都消失了，因为我们仅仅爬取了网站的文本信息。

```
27 url = r'https://www.szu.edu.cn/'
28 # UserAgent代表使用浏览器内核，在爬取数据时不断切换浏览器内核可起到一定的伪装作用
29 header = {
30     'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML,
31 }
32 reply = requests.get(url=url, headers=header)
33 reply.encoding = 'utf-8'
34 reply = reply.text #获取网页文本信息
35 with open("SZUpage.html", 'w', encoding='utf-8') as f:
36     f.write(reply)
```

图 11. 爬取网站文本信息



图 12. 网站文本信息

网站文本信息得到了之后，我们再来看网站别的资源的 url 地址。我们之前看到网站文件比较多，尽管可以用列举的方法一一列出，但是对于大型网站资源比较多的情况下，列举就不是一个很好的办法，为此我们引入正则表达式(regular expression)的方法进行字符串匹配。

我们首先观察网页，网站文件一般储存在 src(source)和 href(Hypertext Reference, 超文本引用)中。在网页源代码中，src 是 source 的缩写，是指向外部资源的位置，指向的内部会迁入到文档中当前标签所在的位置；在请求 src 资源时会将其指向的资源下载并应用到当前文档中，例如 js 脚本，img 图片和 frame 等元素。href 用来建立当前元素和文档之间的链接。常用的有：link 和 a。

首先我们看一下正则表达式速查表，如图 13 所示。

正则表达式	描述
^	匹配字符串的开头
\$	匹配字符串的末尾。
.	匹配任意字符，除了换行符，当re.DOTALL标记被指定时，则可以匹配包括换行符的任意字符。
[...]	用来表示一组字符,单独列出: [amk] 匹配 'a', 'm' 或 'k'
[^...]	不在[]中的字符: [^abc] 匹配除了a,b,c之外的字符。
re*	匹配0个或多个的表达式。
re+	匹配1个或多个的表达式。
re?	匹配0个或1个由前面的正则表达式定义的片段，非贪婪方式

图 13. 正则表达式功能

我们先来看 src 里面的文件。检查网站代码，一般的 src 格式都是：

```
src = ".../.../xxx.format"
```

所以根据正则表达式表我们得到该正则表达式为

```
'src = "([^\"]*)"'
```

根据 href 中的内容，检查网页源代码发现，a 或 link 和 href 之间有时候有其他参数，对比一下如图 14 所示。所以我们需要将这部分考虑在内。

```
<link rel="shortcut icon" href="images/favicon.ico">
<link href="css/animate.css" rel="stylesheet">
```

图 14. href 需要判断多种情况

以上这些并不能说是全部的文件。例如，有一些图片被放进了 div 的 style 参数种，所以我们要将其提取出来。这个格式还是有不同，如图 15 所示，空格所在位置的不同导致我们要分门别类进行匹配。

```
<div class="row-a1" style="background-image: url(images/z-bg1.jpg);">
<div class="icon" style="background-image:url(images/icons01.png);">
<div class="bg" style=" background-image:url(images/m5.png);">
```

图 15. background-image 不同样式

综上所述，正则表达式匹配文件的代码如图 16 所示。

```
36  filelist = re.findall('src="([^\"]*)"', reply)
37  list1 = re.findall('<Link.*href="([^\"]*)"', reply)
38  list2 = re.findall('style="background-image:url\(((.*)\);"', reply)
39  list3 = re.findall('style=" background-image:url\(((.*)\);"', reply)
40  list4 = re.findall('style="background-image: url\(((.*)\);"', reply)
41  #list5 = re.findall('<a.*href="([^\"]*)"', reply)
42  filelist.extend(list1)
43  filelist.extend(list2)
44  filelist.extend(list3)
45  filelist.extend(list4)
```

图 16. 正则表达式匹配文件

得到了文件路径之后，我们可以用之前规范化的方法进行规范化，再加上主机名就是文件的所在位置。

- 3) 前两步完成了文件的提取，接下来我们需要按照它们原来的相对路径，在我们的工作目录下重新建立这些文件夹，以方便我们提取文件。我们的方法是，将得到的文件路径，按照/号进行分割，然后用 os.mkdir 逐层创建文件夹。代码如图 17 所示。

文件夹创建好之后就可以用写入(write)的方法来写入文件。在这里我们要注意一点，我们下载流文件的时候，我们需要将 requests 库中的 stream 设置为 True，然后分块进行传输。因为流文件一般比较大，一次传输可能会有数据丢失。我们设置 chunk(块)进行传输，大小设置为 1024 即可，代码如图 18 所示。

文件夹创建好之后就可以用写入(write)的方法来写入文件。在这里我们要注意一点，我们下载流文件的时候，我们需要将 requests 库中的 stream 设置为 True，

然后分块进行传输。因为流文件一般比较大，一次传输可能会有数据丢失。我们设置 chunk(块)进行传输，大小设置为 1024 即可，代码如图 18 所示。

然后我们统计各个格式文件的大小总和。

```
58     #构造初始文件相对路径
59     file_path = '.'
60     for i, folder in enumerate(file_url.split('/')):
61         if i != len(file_url.split('/'))-1 and i != 0:
62             file_path = file_path + '/' + folder
63             make_folder.append(file_path)
64     for folder in make_folder:
65         if not os.path.exists(folder):
66             os.mkdir(folder)
```

图 17. 创建文件夹

```
74     res = requests.get(url=path, headers=header, stream=True)
75     chunk_size = 1024
76     with open(file_path+'/' + file_name, 'wb') as f:
77         print(file_path+'/' + file_name)
78         for chunk in res.iter_content(chunk_size=chunk_size):
79             f.write(chunk)
```

图 18. 传输流文件

我们知道，python 的 `os.path.getsize` 方法可以得到文件的大小。但是，观察新建的文件夹结构，我们可以发现有以下文件夹里面嵌套了一些文件夹，所以直接用这个方法是不够的。所以我们需要进入文件夹内部，遍历所有文件来计算文件大小总和。对于文件夹内部的子文件夹也有可能文件夹，我们可以用递归的方式进行计算。这里介绍两个 `os` 库的函数。

`os.listdir()`用于返回一个由文件名和目录名组成的列表；

`os.path.isdir()`用于判断对象是否为一个目录；

`os.path.join()`.连接两个或更多的路径名组件。

结合以上三个函数以及我们叙述的递归方法，我们可以计算所有文件夹的大小。计算文件夹大小的代码如图 19 所示。

```
104 def calculate_total_size(folder_path):
105     total_size = 0
106     for file_name in os.listdir(folder_path):
107         path = os.path.join(folder_path, file_name)
108         if os.path.isdir(path): # 递归调用，计算子孙文件夹中所包含的文件总大小
109             total_size = total_size + calculate_total_size(path)
110         if os.path.isfile(path): # 直接计算
111             total_size = total_size + os.path.getsize(path)
112     return total_size
```

图 19. 计算文件夹大小

然后将我们提取出来的文件夹分别计算大小，即可得到所有文件夹的大小。

4) 拓展部分

在这部分我们完成对站长工具 <https://sc.chinaz.com/ppt/free.html> 里面 ppt 网页的爬取。在这里面我们使用 `etree` 以及 `xpath` 的方法进行爬取。

函数解释：

`etree.HTML()`可以用来解析字符串格式的 HTML 文档对象，将传进去的字符串转变成 `_Element` 对象。

xpath: 获取 html 源码中的内容并进行解析, 返回的结果是一个列表, 所以通常在取 xpath()方法结果的时候, 只取列表中的第一个元素。
首先通过 get 方法获取网页的文本信息, 然后观察网页。每一个 ppt 选项链接都是有如图 20 所示的结构。

```
<div class="bot-div">
  <a target="_blank" href="/ppt/210417047380.htm" class="name" title="市场营销计划书PPT模板">市场营销计划书PPT模板</a>
</div>
```

图 20. ppt 在文本中的路径

所以, 我们要找到 xpath 路径为 "//div[@class='bot-div']/a/@href", 这样就得到了 ppt 所在网页的路径。对于主页面的所有 ppt 链接, 我们都这样做, 既可以得到所有的链接地址。

然后, 针对一个链接我们来爬取。首先获取 ppt 名称。打开网页源代码, 点击网页中的 ppt 名称, 可以看到源代码中有了定位, 如图 21 所示。这样我们就得到了 ppt 名称的路径 "//h1[@class='title']/text()"

```
<h1 class="title">毕业论文开题报告PPT模板</h1> == $0
```

图 21. ppt 名称路径

然后我们开始爬取 ppt 资源。在网页中找到 ppt 下载的位置, 对应到网页源代码中如图 22 所示。

```
<div class="download-url">
  <a href="https://downsc.chinaz.net/Files/Download/moban/202104/zppt8456.rar">厦门电信下载</a> == $0
  <a href="https://downsc.chinaz.net/Files/Download/moban/202104/zppt8456.rar">福建移动下载</a>
  <a href="https://downsc.chinaz.net/Files/Download/moban/202104/zppt8456.rar">福建电信下载</a>
  <a href="https://downsc.chinaz.net/Files/Download/moban/202104/zppt8456.rar">厦门移动下载</a>
</div>
```

图 22. ppt 下载路径

这里我们选其中一个链接即可, 以第一个为例, 那么 ppt 的下载路径就是: "//div[@class='download-url']/a[1]/@href"

得到了下载路径之后就用 requests 方法来爬取即可。代码如图 23 所示。

```
10 page_text = requests.get(url=url, headers=header).text
11 etree_ppt = etree.HTML(page_text)
12 # 每个ppt的url
13 ppt_url = etree_ppt.xpath("//div[@class='bot-div']/a/@href")
14 print(ppt_url)
15
16 for each_url in ppt_url:
17     # 每个ppt的完整下载url
18     each_url = r'https://sc.chinaz.com' + each_url
19
20     ppt_text = urlopen(each_url, timeout = 10).read().decode('utf-8')
21     ppt_etree = etree.HTML(ppt_text)
22     # 每个ppt的名字
23     ppt_name = ppt_etree.xpath("//h1[@class='title']/text()")[0]
24     print(ppt_name)
25     # 每个ppt的下载链接
26     ppt_download_url = ppt_etree.xpath("//div[@class='download-url']/a[1]/@href")[0]
27     print(ppt_download_url)
28     # 下载获取二进制数据
29     ppt_data = requests.get(ppt_download_url, headers=header).content
30     with open("ppt_container/" + ppt_name + '.rar', 'wb') as f:
31         f.write(ppt_data)
```

图 23. 用 etree 和 xpath 来爬取网页 ppt 代码

3. TCP 通信部分:

- a) 指令交互: 客户端向服务器端发送 Time 命令, 服务器端接受到该字符串后将服务器端当前时间返回给客户端; 客户端向服务器端发送 Exit 命令, 服务器端向客户端返回“Bye”后关闭连接退出程序;

在任何类型的通信开始之前, 网络应用程序都必须创建套接字。有两种类型的套接字, 即基于文件的套接字和面向网络或者地址的套接字。在所有的地址家族之中, 目前 AF_INET 是使用得最广泛的。为了创建 TCP 套接字, 必须使用 SOCK_STREAM 作为套接字类型。

此外, 我们还需绑定地址关键字, AF_INET 下以元组的形式表示地址。常用 bind((host,port)), host 表示主机 IP 地址, port 表示端口号。这一部分的代码如图 24 所示。

```
6 serversocket = socket.socket(socket.AF_INET,socket.SOCK_STREAM)
7 host = socket.gethostname() #获取本地主机名
8 port = 9999 #固定端口号
9 #绑定地址关键字, AF_INET下以元组的形式表示地址。常用bind((host,port))
10 serversocket.bind((host,port))
```

图 24. 服务器设置 TCP 通信并绑定地址关键字

接下来, 我们需要被动接受 TCP 客户端的连接。这里 python 使用的是 accept 方法。这个方法返回一个新的套接字以及链接地址。具体方法为

clientsocket,addr = serversocket.accept()

服务器与 TCP 客户端连接之后, 就可以接受客户端发过来的信息了。python 中我们用 recv 函数来接收数据, recv 有一个参数, 指定数据缓冲区的大小, 默认设置为 1024, 单位为 byte。发送数据可以采用 send, 参数就是想要发送的字符串。send 前面是套接字, 这个用来确定发送到哪个客户端。在服务器中, 不管是接收数据还是发送数据, 都要用编码的方式。发送的时候, 将要传输的数据编码发送, 一般以 utf-8 为编码方式。发送的时候要进行 encode(加密), 接受的时候要 decode(解密)。客户端的方法亦然。

例如, 当客户端发送 Time 的时候, 服务器首先要接收数据, 然后进行解码得到 Time 指令。然后通过 datetime 库来获取当下时间。时间获取之后转成字符串, 通过 send 方法进行发送。代码如图 25 所示。

```
23 try:
24     data = clientsocket.recv(1024) #接收数据
25 except Exception:
26     print('断开的客户端: ',addr)
27     break
28 print('客户端发送内容: ',data.decode('utf-8'))
29 flag = 0
30 if data.decode('utf-8') == "Time":
31     cur = datetime.datetime.now().strftime('%F %T')
32     reply = '当前时间为: ' + cur
```

图 25. 服务器接收客户端"Time"指令

时间信息由服务器发送之后, 客户端接收信息并解码就可以输出当前的时间信息了。

这里面的 reply 是确定好了的, 如果需要我们手动输入 reply 的信息, 可以调用 input 函数, 然后让 reply 接受输入即可。

- b) 文件传输：客户端向服务器端发送文件请求，服务器端接收到请求后，先向其传输文件名，当客户端接收后，再向客户端传输文件。客户端接收文件，并对其改名（文件名可自行定义）和存在本地。

这部分和指令交互的区别是，我们需要读取文件。在文件存在的条件下，这里面我们要调用之前提及的 `with open`，只不过这里面调用 `'rb'` 参数来读取文件进行发送。值得注意的是，有一些文件可能比较大，如果一次性发送文件，那么客户端的 `recv` 缓冲区可能无法全部接收。所以我们可以每次发送 1024 bytes 的大小，直到全部发送完毕。

如果文件不存在，则返回 No such file 提示信息。

代码如图 26 所示。

```
4 def file_send(file_name, client_socket):
5
6     path = r'C:\Users\宋爽\.spyder-py3\socket\socket2.1'
7     ct = 0
8     for filename in os.listdir(path):
9         if filename == file_name:
10             filesize = os.path.getsize(file_name)
11             sfilesize = '%d'%filesize
12             client_socket.send(sfilesize.encode())
13             with open(file_name, "rb") as f:
14                 while True:
15                     file_content = f.read(1024)
16                     if file_content:
17                         client_socket.send(file_content)
18                     else:
19                         break
20             ct = 1
21             break
22     if ct == 0:
23         print("No such file")
24         client_socket.send(b'No such file')
```

图 26. 发送文件数据的函数

接下来从客户端角度出发来接收文件。客户端需要接输入想要下载的文件，然后服务器端搜索是否存在该文件。如果存在，那么则进行接收。接受的过程依然与发送过程类似，发送的时候是以 1024 bytes 为一块进行发送，接受的时候也要以 1024 bytes 为一块进行接收。由于服务端发送了文件的大小，所以客户端需要计算有多少块（不足向上取整），然后用 `with open` 进行写入。代码如图 27 所示。

下载完成之后，我们还可以调用 `os.path.getsize` 计算下载量。如果没有该文件，那么就输出 No such file，如果客户输入 Exit 可以直接退出。

用户细节：我们可以将服务器中所有的文件打印出来方便用户选择。可以调用之前提到过的 `os.listdir` 以及 `os.path.join` 来输出文件名。

```

45         filesize = mes.decode()
46         ifilesize = int(filesize)
47         count = int(ifilesize/1024)
48         if ifilesize % 1024 > 0:
49             count = count + 1
50         with open('./[new]' + file_name, 'wb') as f:
51             for i in range(0,count):
52                 mes = tcp_socket.recv(1024)
53                 f.write(mes)
54             f.close()

```

图 27. 将数据写入文件

- c) 拓展部分：在服务器端存储一个包含学生学号、姓名、性别信息的数据库，客户端每次发送一个学生学号给服务器，服务器则返回对应该学号的学生姓名和性别（提示：需要解决 TCP 粘包问题）；以及任意格式文件的传输
首先启动 WampServer 建立一个虚拟的学生数据库，字段数为 3，分别为学号、姓名以及性别，如图 28 所示。

+ 选项

	student_ID	student_name	student_gender
<input type="checkbox"/> 编辑 复制 删除	2021152001	Harry	Male
<input type="checkbox"/> 编辑 复制 删除	2021152002	Marie	Female
<input type="checkbox"/> 编辑 复制 删除	2021152003	Kane	Male
<input type="checkbox"/> 编辑 复制 删除	2021152004	Jason	Male
<input type="checkbox"/> 编辑 复制 删除	2021152005	Joe	Male
<input type="checkbox"/> 编辑 复制 删除	2021152006	Rose	Female
<input type="checkbox"/> 编辑 复制 删除	2021152007	Daniel	Male
<input type="checkbox"/> 编辑 复制 删除	2021152008	Mike	Male
<input type="checkbox"/> 编辑 复制 删除	2021152009	Maguire	Male
<input type="checkbox"/> 编辑 复制 删除	2021152010	David	Male

图 28. 建立学生信息数据库（模拟）

我们以指令交互的代码为基础来添加这部分功能。我们规定查找某个学生要以命令 *query*: 为起始以作区分。

为了实现 python 连接数据库，我们需要下载 pymysql 库，然后使用 connect 方法连接数据库，如图 29 所示。

```

12 db = pymysql.connect(host='localhost', user='root', password='',
13                       database='socket', cursorclass=pymysql.cursors.DictCursor)
14 cursor = db.cursor()

```

图 29. python 连接数据库

其中，host 和 user 是数据库参数，database 指明了是哪一个数据库，cursorclass=pymysql.cursors.DictCursor 设置数据库返回结果为字典类型。然后 cursor 是一个游标，方便我们访问数据库。

当我们的用户输入了一个学号之后，我们要进行查找。mysql 数据库中查找语句为 select，我们需要截取 *query*: 后面的学号信息，然后进行查找即可。查找语句

储存在 sql 中, 然后需要调用 execute 方法来执行。这里介绍两个 cursor 的属性:

cursor.rowcount: 返回符合查找条件的元组个数;

cursor.fetchall(): 返回结果集, 由于我们设置了以字典形式返回, 所以它也是一个字典型变量。

这里需要解决 TCP 粘包的问题。我们知道我们需要返回的是学生姓名以及学生性别, 这两个量是粘连在一起的。所以我们在发送给客户端的时候就用^号来进行分离, 然后客户端调用 split 函数来进行分离。

(split 函数: 分隔字符串。参数为一个字符, 该功能可以按照该字符来切割字符串, 以列表的形式返回。当然, split 函数也可以存在第二个参数, 表示分割的个数。)

服务器端代码如图 30 所示。

```
elif data.decode('utf-8')[0:6] == "query:":
    str1 = data.decode('utf-8')[6:]
    print("client wants to query the info of "+str1)
    sql = "select student_name,student_gender from student where student_ID = "+str1
    cursor.execute(sql)
    if cursor.rowcount == 0:
        reply = 'Sorry, no such student, pls check.'
    for iter in cursor.fetchall():
        reply = '%s^%s^%s' % (iter["student_name"],iter["student_gender"])
```

图 30. 查找学生--服务器端代码

客户端代码如图 31 所示。这里我们调用 split 函数来解决 TCP 粘包问题, 第二个参数设置为 2, 代表用^号将服务器发来的结果分割成 3 块。结合图 30 中的代码, 可以看到学生姓名在第二个, 学生性别在第三个, 按照这个顺序输出信息即可。

```
if msg_decode[-34:] == "Sorry, no such student, pls check.":
    print(msg_decode)
else:
    msg_divided = msg_decode.split("^",2)
    tag = msg_divided[0]
    msg1 = msg_divided[1]
    msg2 = msg_divided[2]
    print(tag+"The name of the student is "+msg1+", while the gender of the student is "+msg2)
```

图 31. 查找学生—客户端代码

此外, 关于任意格式文件的收取, 由于我们原来是按照二进制方法来读取文件, 而任何格式的文件实际上都是由二进制组成的, 所以我们的程序可以做到任意格式文件的接收。结果见实验结果。

4. UDP 通信部分:

- a) 编写完整程序包含一个服务器端程序和多个客户端程序 (不同客户端代码几乎一样), 使多个客户端之间可以彼此通信聊天 (类同微信、QQ);

首先我们实现 UDP 客户端与服务器之间的沟通。首先, 服务器需要确定 socket 类型。由于这里面是基于 UDP 协议的通信, 所以我们选择 SOCK_DGRAM。这是是无保障的面向消息的 socket, 主要用于在网络上发广播信息。它基于 UDP 协议, 专门用于局域网, 基于广播。然后我们需要绑定主机以及端口号。与 TCP 协议不同的是, 我们不需要 accept 方法以及设置监听个数, 直接就可以接收与服务器连接的客户端的信息。

一般情况下, 客户端和服务端都使用 sendto 方法来发送信息, 用 recvfrom 方法来接收信息。通信流程如图 32 所示。

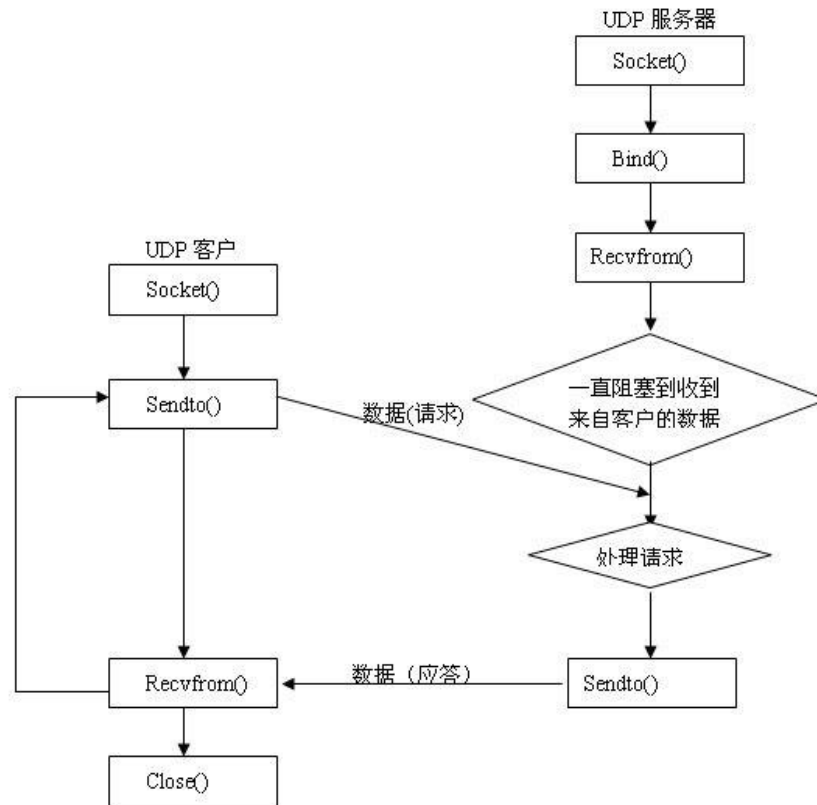


图 32. UDP 通信流程

接下来我们讨论客户端之间的通信问题。我们已经确定了客户端与服务器的通信，那么我们设想如果以服务器作为中转站，将服务器作为一个传话筒，这样就可以实现客户端之间的联系。传输过程如图 33 所示。

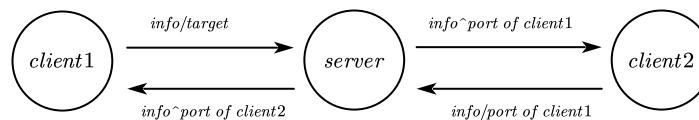


图 33. 客户端通过服务器彼此沟通

我们针对图 33 进行解释。假如 *client1* 是信息的发送方，

- 1) *client1* 将想要发送的信息，也就是 *info*，以及想要发送的目标 *target* 发送到服务器。我们可以用 / 将两部分信息分割开来；
- 2) 服务器接收到 *client1* 的请求信息之后，需要起到中转站的作用。我们需要将发起通话的发信人信息发送给 *client2*，同时发送 *client1* 的端口号，告诉收信人 *client2* 是谁在跟他通话。 *client2* 收到来自服务器的信息之后，就知道是谁在跟它通话、通话的内容是什么，这两个信息用 ^ 来分割；
- 3) *client2* 收到了 *client1* 的信息之后，可以输入想要回复的信息，然后告诉服务器这是回复信息，所以要告知服务器 *client1* 的端口号；
- 4) 服务器接收到回复信息之后，将信息返回给 *client1*，并将 *client2* 的端口号告知 *client1*。

以上就是两客户端通信的基本过程。

代码如图 34 所示。


```

28 data,address = server.recvfrom(1024)
29
30 data = data.decode('utf-8')
31
32 data_divided = data.split('/')
33
34 #data_divided[0]是信息，data_divided[1]是发信人的目标(1,2,3...), address[1]是发信人的端口号
35 send = data_divided[0] + "^" + str(address[1])
36
37 server.sendto(send.encode('utf-8'),(host,addr[data_divided[1][-1]]))

```

图 34. UDP 客户端通信服务器端代码

为了确定是谁先来发信息，我们要在客户端设置两个条件，一个是接收信息，一个是准备发送信息。这个判断由服务端确定，这里由服务端来指定发送信息的客户端是哪一个。

发送信息：如图 35 所示。

```

46 print("pls select a friend to communicate:")
47 selection = input('')
48 while True:
49     if selection != "1" and selection != "2" and selection != "3" and selection != "4":
50         print("error input, pls check")
51         selection = input('')
52     else:
53         break
54
55 send_data = input('>>').strip()
56 info = send_data + "/" + selection
57 client.sendto(info.encode('utf-8'),(host,9999))

```

图 35. UDP 客户端通信客户端代码--发送信息

这里我们输入一个客户端，这里所有的客户端由服务器发送给客户端，然后客户端选择其中一个客户端进行通话。这里我们要设置一个判断条件，就是选择的是哪一个，然后是否存在这个客户端。然后将发送的信息和目标一起发送到服务器。

接收信息：如图 36 所示。

```

30 re_divided = re_Data.split("^")
31
32 print("来点" + list(dic.keys())[list(dic.values()).index(int(re_divided[1]))] + "的信息:" + re_divided[0])
33 if re_divided[0] == "Exit":
34     send_data = "Bye"
35     print("Chat is over")
36     flag = 1
37 elif re_divided[0] == "Bye":
38     break
39 else:
40     send_data = input('>>').strip()
41     info = send_data + "/" + list(dic.keys())[list(dic.values()).index(int(re_divided[1]))]
42     client.sendto(info.encode('utf-8'),(host,9999))
43 if flag == 1:
44     break

```

图 36. UDP 客户端通信客户端代码—接收信息

接收信息的话，我们需要首先用^号先进行分割，得到了发信人以及通话的内容。现在我们得到了发信人以及会话内容。然后输入我们想要回复的内容，发送到服务器即可。

- b) 客户端之间能够借助用户设计的 UI 界面显示聊天交互过程，并包含上述所有功能。

这里面我们用 Java 来实现通话功能。

Java 通过 DatagramPacket 类和 DatagramSocket 类来使用 UDP 套接字，客户端和服务端都通过 DatagramSocket 的 send 方法和 receive 方法来发送和接收数据，用 DatagramPacket 来包装需要发送或者接收到的数据。

- 1) 发送信息时，Java 创建一个包含待发送信息的 DatagramPacket 实例，并将

其作为参数传递给 DatagramSocket 实例的 send () 方法;

- 2) 接收信息时, Java 程序首先创建一个 DatagramPacket 实例, 该实例预先分配了一些空间, 并将接收到的信息存放在该空间中, 然后把该实例作为参数传递给 DatagramSocket 实例的 receive 方法。

有关 UDP 通信部分, Java 的实现方法和 python 的实现方法是类似的, 用法区别在于以上两点。现在叙述界面的构造。

我们使用的是 JPanel, 它是 Java 图形用户界面(GUI)工具包 swing 中的面板容器类。我们用到类型有: JTextArea(文本域组件)、JButton(按钮组件)等等。同时我们注意到, 当一个容器内放置了许多组件, 而容器的显示区域不足以同时显示所有组件时, 如果让容器带滚动条, 通过移动滚动条的滑块, 容器中位置上的组件就能看到。滚动面板 JScrollPane 能实现这样的要求, JScrollPane 是带有滚动条的面板。JScrollPane 是 Container 类的子类, 也是一种容器, 但是只能添加一个组件。

我们首先设想我们的界面的大致形状, 如图 37 所示。

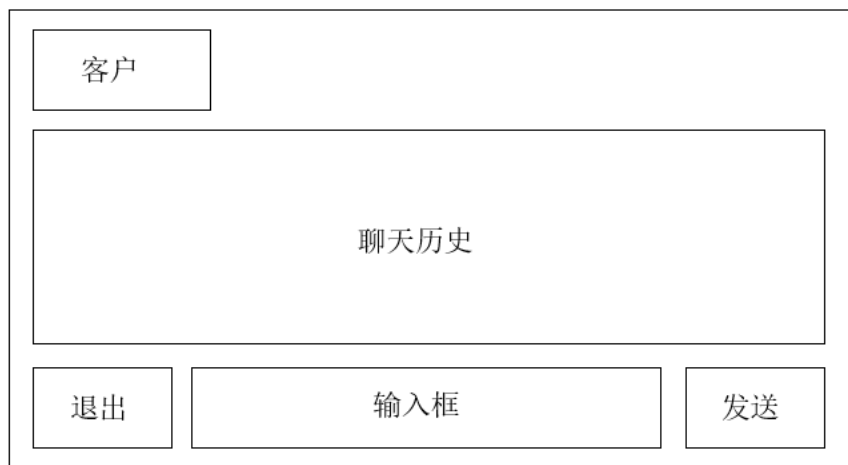


图 37. UDP 通信界面

然后定义组件, 并设置布局。代码如图 38 所示。

```
super( title: "client B");
Container contain=getContentPane();
contain.setLayout(new BorderLayout( hgap: 3, vgap: 3));
//历史记录部分
mainArea=new JTextArea();
mainArea.setEditable(false);
//历史记录部分的滚动条功能实现
JScrollPane Scroll=new JScrollPane(mainArea);
Scroll.setBorder(BorderFactory.createTitledBorder("chat history"));
//输入框部分
JPanel panel=new JPanel();
panel.setLayout(new BorderLayout());
//设置大小
sendArea=new JTextArea( rows: 3, columns: 8);
//输入框部分需要滚动条功能
JScrollPane sendScroll=new JScrollPane(sendArea);
UserChat=new BUserChat( ui: this);
```

图 38. 定义控件以及进行布局

然后我们需要将设置 send 按钮，并为其设置监听事件，实现点击按钮可以进行发送功能。代码如图 39 所示。exit 按钮则监听 system.exit(0)事件。

```
//开始发送信息
UserChat.start();
sendBtn=new JButton( text: "send");
//事件处理
sendBtn.addActionListener(new ActionListener(){
    public void actionPerformed(ActionEvent e){
        UserChat.sendMsg(sendArea.getText().trim());
        mainArea.append("[clientB]" + sendArea.getText().trim() + "\n");
        sendArea.setText("");
    }
});
```

图 39. 定义按钮以及监听事件

然后添加内容即可。然后还要设置 setVisible(true)，这是允许 JVM 可以根据数据模型执行 paint 方法开始画图并显示到屏幕上了，并不是显示图形，而是可以运行开始画图了。然后设置一个默认的关闭操作，结束进程。代码如图 40 所示。

```
JPanel tmpPanel=new JPanel();
tmpPanel.add(sendBtn);
panel.add(tmpPanel,BorderLayout.EAST);
panel.add(sendScroll,BorderLayout.CENTER);
contain.add(Scroll,BorderLayout.CENTER);
contain.add(panel,BorderLayout.SOUTH);
setSize( width: 500, height: 300);
setVisible(true);
setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
```

图 40. 添加组件、可视化以及默认关闭操作

这样，一个聊天界面就制作完毕了。两个客户端的代码是类似的。相关结果位于实验结果部分。

拓展部分：这部分我们主要是 UI 界面的美化，给聊天界面添加背景图。

对于 JPanel 来说，添加背景图片有很多方法，例如重写 panel 方法等等。这里面我们采取相对容易一点的方法，直接在 JTextArea 部分添加背景图。首先声明 Image 型变量，接收资源图片。然后写 paint 函数，打印图片并设置图片随聊天框大小改变而改变。要用到的工具如下：

paintComponent 是绘制容器中的组件；应该绘制组件的内容时调用此方法；例如首次显示组件或者组件已损坏并需要修复时。Graphics 参数中的矩形框设置为需要绘制的区域；

getParent 方法：该函数获得一个指定子窗口的父窗口句柄。例如，通过 getSize 方法得到窗口大小，这里为了实现图片大小随着窗口大小改变而改变，应该通过这个函数获得窗口当前大小，然后相应地进行绘制；

drawImage 方法：绘制指定图像中已缩放到适合指定矩形内部的图像。

函数原型：drawImage (image, x, y, width, height, observer).参数作用都比较显然，其中 observer 代表当转换了更多图像时要通知的对象，这里不需要通知任何对象，设置为 null 即可。

代码如图 41 所示。

```
//背景图片
Image image1 = new ImageIcon( filename: "src/sok.jpg").getImage();
mainArea=new JTextArea() {
    Image image2 = image1;
    Image grayImage = GrayFilter.createDisabledImage(image2);
    {
        setOpaque(false);
    }

    public void paint(Graphics g) {
        super.paintComponent(g);
        Dimension size=this.getParent().getSize();
        g.drawImage(image2, x: 0, y: 0,size.width,size.height, observer: null);
        //g.drawImage(image2, 0, 0, this);
        super.paint(g);
    }
}
```

图 41. 界面美化—添加背景图片

五、实验结果：

1. 实验 4.1 部分关于爬取深大主页的结果。

在 python 文件的工作目录中，我们看到了如图 42 所示的文件夹。

名称	修改日期	类型	大小
^			
__local	2021/4/10 0:12	文件夹	
_sitegray	2021/4/10 0:12	文件夹	
css	2021/4/10 0:12	文件夹	
images	2021/4/10 0:12	文件夹	
info	2021/4/17 23:41	文件夹	
js	2021/4/10 0:12	文件夹	
jsrk	2021/4/17 23:41	文件夹	
system	2021/4/10 0:11	文件夹	
index.vsb.css	2021/4/18 0:37	层叠样式表文档	2 KB
SZUpage.html	2021/4/17 23:56	Chrome HTML D...	94 KB

图 42. 爬取深大主页之后得到的文件夹

为了验证爬取下来的文件夹是否与网站的文件相对路径吻合，我们以 src 为例观察一下网站的 src，如图 43 所示。可以看到，文件夹与相对路径是一样的，点击进文件夹来看也是一样的结果。

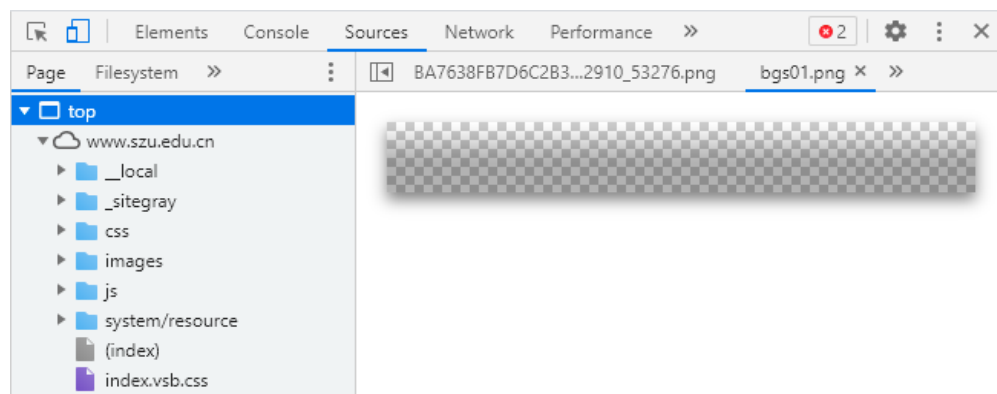


图 43. 网站文件夹

以其中一个文件夹 js 为例，可以看到如图 44 所示，和网站的 js 文件夹是一样的。



图 44. js 文件夹文件内容

打开其中一个 js 文件，如图 45 所示可以看到其内容并没有出现乱码的情况，说明文件编码是正确的。

```
1 $(document).ready(function() {  
2     // 手机导航  
3     $('.menuBtn').append('<b></b><b></b><b></b>');  
4     $('.menuBtn').click(function(event) {  
5         $(this).toggleClass('open');  
6         $('.nav').stop().slideToggle();  
7     });  
8  
9     var _winw = $(window).width();  
10    if (_winw > 959) {  
11        $('.nav li').hover(function() {  
12            $(this).find('.sub').stop().slideDown();  
13            if ($(this).find('.sub').length) {  
14                $(this).addClass('ok');  
15                // $('.header').addClass('on');  
16            } else {  
17                // $('.header').removeClass('on');
```

图 45. JavaScript 文件正常

接下来我们展示一下统计数据，如图 46 所示，所有文件夹的大小都被统计了出来。

```
The size of the folder/document __local is 32357.23828125 KB  
The size of the folder/document js is 260.90625 KB  
The size of the folder/document images is 3004.294921875 KB  
The size of the folder/document system is 24.783203125 KB  
The size of the folder/document css is 186.798828125 KB  
The size of the folder/document index.vsb.css is 1.111328125 KB  
The size of the folder/document _sitegray is 0.1123046875 KB
```

图 46. 各文件夹大小统计结果

实验 4.1 拓展部分实验结果

如图 47 所示，所有的 ppt 下载相对地址都被打印了出来。

```
In [3]: runfile('C:/Users/宋爽/Desktop/untitled0.py', wdir='C:/Users/宋爽/Desktop')  
['/ppt/210412427140.htm', '/ppt/210412457701.htm', '/ppt/210412174390.htm', '/ppt/  
210412203381.htm', '/ppt/210412163450.htm', '/ppt/210412228111.htm', '/ppt/  
210411436940.htm', '/ppt/210411465431.htm', '/ppt/210411309610.htm', '/ppt/  
210411354591.htm', '/ppt/210410245020.htm', '/ppt/210410278491.htm', '/ppt/  
210410142910.htm', '/ppt/210410176201.htm', '/ppt/210409464840.htm', '/ppt/  
210409529311.htm', '/ppt/210409574750.htm', '/ppt/210409013901.htm', '/ppt/  
210409578970.htm', '/ppt/210409007781.htm']
```

图 47. 所有 ppt 文件链接地址

再来看文件名称的打印，如图 48 所示，我们截取了一部分文件名称以及 ppt 的名称，可以看到文件名称以及 ppt 的名称都是正确的。

```
部门工作汇报PPT
https://downsc.chinaz.net/Files/DownLoad/moban/202104/zppt8386.rar
企业内训师培训PPT模板
https://downsc.chinaz.net/Files/DownLoad/moban/202104/zppt8412.rar
部门半年工作总结PPT模板
https://downsc.chinaz.net/Files/DownLoad/moban/202104/zppt8385.rar
简约大气项目策划书PPT模板
https://downsc.chinaz.net/Files/DownLoad/moban/202104/zppt8406.rar
管理者经验分享PPT模板
https://downsc.chinaz.net/Files/DownLoad/moban/202104/zppt8402.rar
消防知识培训课程PPT模板
https://downsc.chinaz.net/Files/DownLoad/moban/202104/zppt8432.rar
```

图 48. 文件名称以及 ppt 名称的打印

再来到 python 工作目录，可以看到所有的下载好的 rar 文件，如图 49 所示。






















	部门半年工作总结PPT模板.rar	2021/4/13 0:43	RAR 压缩文件	2,899 KB
	部门工作汇报PPT.rar	2021/4/13 0:42	RAR 压缩文件	1,494 KB
	管理者经验分享PPT模板.rar	2021/4/13 0:43	RAR 压缩文件	4,228 KB
	简约大气项目策划书PPT模板.rar	2021/4/13 0:43	RAR 压缩文件	6,003 KB
	企业内训师培训PPT模板.rar	2021/4/13 0:42	RAR 压缩文件	3,004 KB
	企业行政管理培训PPT模板.rar	2021/4/13 0:43	RAR 压缩文件	2,470 KB
	人力资源管理培训PPT模板.rar	2021/4/13 0:43	RAR 压缩文件	1,600 KB
	软件测试报告总结PPT模板.rar	2021/4/13 0:43	RAR 压缩文件	980 KB
	软件功能测试总结归纳PPT模板.rar	2021/4/13 0:43	RAR 压缩文件	1,339 KB
	商务风新品发布会PPT模板.rar	2021/4/13 0:43	RAR 压缩文件	1,822 KB
	商务月报总结PPT模板.rar	2021/4/13 0:43	RAR 压缩文件	2,209 KB
	上半年运营总结PPT模板.rar	2021/4/13 0:43	RAR 压缩文件	1,497 KB
	市场营销计划书PPT模板.rar	2021/4/17 23:02	RAR 压缩文件	2,156 KB
	项目投资计划书PPT模板.rar	2021/4/13 0:43	RAR 压缩文件	6,781 KB
	消防知识培训课程PPT模板.rar	2021/4/13 0:43	RAR 压缩文件	681 KB
	销售部门半年总结PPT模板.rar	2021/4/13 0:43	RAR 压缩文件	4,670 KB
	销售人员沟通技巧PPT模板.rar	2021/4/13 0:44	RAR 压缩文件	5,795 KB
	小清新春季主题PPT模板.rar	2021/4/13 0:43	RAR 压缩文件	3 KB
	一起去旅行相册PPT模板.rar	2021/4/13 0:43	RAR 压缩文件	1,605 KB
	职业生涯规划报告PPT模板.rar	2021/4/13 0:43	RAR 压缩文件	1,995 KB
	转正工作汇报PPT模板.rar	2021/4/13 0:43	RAR 压缩文件	3,785 KB

图 49. 所有下载好的文件

文件打开之后，可以看到文件都是正常的，如图 50 所示。

名称	压缩后大小	原始大小	类型	修改日期
说明.htm	1,470	3,177	Chrome HTML Docu...	2016/11/3 13:54:27
zppt8385_s.jpg	36,186	36,903	JPG 文件	2021/4/1 17:22:28
zppt8385.ppt	2,930,449	3,488,256	Microsoft PowerPoint...	2021/4/1 10:24:33

图 50. rar 文件都是正常的

2. TCP 通信部分结果

2.1 TCP 指令交互

1) 输入 Time 指令之后返回了系统时间，如图 51 所示。

```
>>Time
服务端回应: [2021-04-19 00:19:17]:当前时间为: 2021-04-19 00:19:17
```

图 51. Time 指令返回结果

2) 输入一般的语句, 可以进行通信, 如图 52 所示。

```
>>Hello and this is client. Can u hear me?  
服务端回应: [2021-04-19 00:25:41]:Yep this is server and we can hear from u.  
  
>>你好, 这里是客户端。可以收到消息吗?  
服务端回应: [2021-04-19 00:26:19]:这里是服务器, 收到了消息。
```

图 52. TCP 聊天结果

3) 输入 Exit, 可以退出程序。如图 53 所示。

```
>>Exit  
Bye
```

图 53. Exit 退出程序

2.2 TCP 文件传输

我们尝试 ls 操作, 可以得到如图 54 所示的结果。

```
ls  
C:\Users\未央\.spyder-py3\socket\socket2.1\client.py  
C:\Users\未央\.spyder-py3\socket\socket2.1\ppt模板.pptx  
C:\Users\未央\.spyder-py3\socket\socket2.1\server.py  
C:\Users\未央\.spyder-py3\socket\socket2.1\server_file.py  
C:\Users\未央\.spyder-py3\socket\socket2.1\样例.txt  
C:\Users\未央\.spyder-py3\socket\socket2.1\样例照片1.JPG
```

图 54. 列举服务器所有文件

然后调用 download 操作, 可以进行下载, 其中包括对违法情况的判断。然后调用 Exit 指令, 退出程序, 如图 55 所示。

```
download  
请输入要下载的文件名: 样例.txt  
文件下载成功!  
您所要下载的文件是 样例.txt  
本次下载共计0.0234375KB  
Pls select an opeation:  
  
download  
请输入要下载的文件名: 样例照片1.JPG  
文件下载成功!  
您所要下载的文件是 样例照片1.JPG  
本次下载共计44.4072265625KB  
Pls select an opeation:  
  
download  
请输入要下载的文件名: ppt模板.pptx  
文件下载成功!  
您所要下载的文件是 ppt模板.pptx  
本次下载共计1481.0478515625KB  
Pls select an opeation:  
  
download  
请输入要下载的文件名: illegal.png  
No such file  
Pls select an opeation:  
  
Exit  
Bye
```

图 55. download 操作下载文件

进入桌面，看到下载的文件，如图 56 所示。下载文件大小与统计下载结果对比来说是一样的，可以说明下载是正常的。打开了文件之后，发现文件都是正常的。





 [new]ppt模板.pptx	2021/4/19 0:31	Microsoft Power...	1,482 KB
 [new]server.py	2021/4/19 0:31	Notepad++ Doc...	2 KB
 [new]样例.txt	2021/4/19 0:30	文本文档	1 KB
 [new]样例照片1.JPG	2021/4/19 0:31	JPG 文件	45 KB

图 56. download 操作下载文件显示在文件夹中

2.3 TCP 与数据库的结合与任意格式文件的下载

如图 57 所示，我们看到了查询数据库的结果，这与数据库中存储的数据是吻合的。此外还包括为违法情况的判断。

```
>>query:2021152001
[2021-04-19 01:21:04]:The name of the student is Harry, while the gender of the student is Male

>>query:2021152007
[2021-04-19 01:21:15]:The name of the student is Daniel, while the gender of the student is Male

>>query:2021152006
[2021-04-19 01:21:41]:The name of the student is Rose, while the gender of the student is Female

>>query:2018152051
[2021-04-19 01:21:52]:Sorry, no such student, pls check.
```

图 57. 连接数据库并查询学生信息结果

任意格式文件的下载：图片我们之前下载过，这里我们以视频、音频为例。首先可以看到，MPEG-4 音频文件大小为 3.37MB，视频文件大小为 2.89MB，如图 58 所示。



 样例音频.m4a	 样例视频.mp4
文件类型: M4A 文件 (.m4a)	文件类型: MP4 文件 (.mp4)
打开方式: Groove 音乐 <button>更改(C)...</button>	打开方式: 电影和电视 <button>更改(C)...</button>
位置: C:\Users\宋央\spyder-py3\socket\socket2.1	位置: C:\Users\宋央\spyder-py3\socket\socket2.1
大小: 3.37 MB (3,541,576 字节)	大小: 2.89 MB (3,040,629 字节)

图 58. 下载其他格式文件的大小

然后用我们的程序进行下载，如图 59 所示。可以看到文件大小是接近的。为此我们检查一下文件夹中的文件，发现播放都是正常的，所以文件格式没有问题。

<pre>Pls select an opreation: download 请输入要下载的文件名:样例音频.m4a 文件下载成功! 您所要下载的文件是 样例音频.m4a 本次下载共计3458.5703125KB</pre>	<pre>Pls select an opreation: download 请输入要下载的文件名:样例视频.mp4 文件下载成功! 您所要下载的文件是 样例视频.mp4 本次下载共计2969.3642578125KB</pre>
--	---

图 59. 下载其他格式文件正常

此外，我们通过优化代码，提供重新命名服务。以其中一个 python 文件为例，我们将其改名后可以看到 client 所在的文件夹已经有了新命名的文件，如图 60 和图 61 所示。

```
download
请输入要下载的文件名:server.py
你想使用默认文件名称吗? y/n
n
请输入文件名称:server_dup.py
文件下载成功!
您所下载的文件是 server.py
本次下载共计1.9541015625KB
```

图 60. 文件改名






	client_file.py	2021/5/9 19:15	Notepad++ Doc...	3 KB
	formula.afx	2021/5/8 19:21	Equation.AxMath	177 KB
	Microsoft Office 365 支持和恢复助手	2019/3/26 23:46	Application Refe...	1 KB
	ppt模板.pptx	2020/7/2 20:50	Microsoft Power...	1,482 KB
	server_dup.py	2021/5/9 22:52	Notepad++ Doc...	2 KB

图 61. 文件改名成功

3. UDP 通信部分结果

3.1 UDP 通信

这里面我们提供 1 个服务器和 4 个客户端，服务器为 server，客户端分别命名为 client 1、client 2、client 3 以及 client 4。

启动服务器以及所有客户端，将会出现如图 62 所示的结果。所有客户端的信息都被打印了出来，这就相当于是一个好友列表。然后选择一个好友进行通讯。

```
client 1, his port is 49372
client 2, his port is 49374
client 3, his port is 49375
client 4, his port is 49376
pls select a friend to communicate:
```

图 62. 客户端收到好友列表并选择通讯对象

这里我们以 client 3 联系 client 1，client 4 联系 client 2 为例，先来看 client 3 联系 client 1 的结果，如图 63 所示。图 63(a)是 client 3 端的通话信息，图 63(b)是 client 1 端的通话信息。通话的最后，client 3 发送 Exit 指令，然后双方退出通话。这里包括了对违法情况的判断。

```
pls select a friend to communicate:

Hi client 1, this is client 3. Can you hear me?
error input, pls check
1
>>Hi client 1, this is client 3. Can you hear me?
来自client 1的信息:Hi client 3, this is client 1. I can hear you. Can you hear me?
>>Yes i can.
来自client 1的信息:Exit
Chat is over
```

图 63(a). client 3 向 client 1 答复信息

```

来自client 3的信息:Hi client 1, this is client 3. Can you hear me?

>>Hi client 3, this is client 1. I can hear you. Can you hear me?
来自client 3的信息:Yes i can.

>>Exit
来自client 3的信息:Bye

```

图 63(b). client 1 向 client 3 答复信息

我们看到服务器端显示了聊天的过程，如图 64 所示。

```

We server receive a message from client 3, and the content is 'Hi client 1, this is client 3. Can you hear me?', the terminal is 1
We server receive a message from client 1, and the content is 'Hi client 3, this is client 1. I can hear you. Can you hear me?', the terminal is client 3
We server receive a message from client 3, and the content is 'Yes i can.', the terminal is client 1
We server receive a message from client 1, and the content is 'Exit', the terminal is client 3
We server receive a message from client 3, and the content is 'Bye', the terminal is client 1

```

图 64. client 1 和 client 3 沟通过程

同时，client 4 向 client 2 发送信息，然后 client 2 向 client 4 答复信息，如图 65 所示。图 65(a)是 client 4 端的通话信息，图 65(b)是 client 2 端的通话信息。

```

pls select a friend to communicate:

2

>>Hi client 2, this is client 4. Can you hear me?
来自client 2的信息:Hi client 4, this is client 2. I can hear you.

>>I can hear you too.
来自client 2的信息:Exit
Chat is over

```

图 65(a). client 4 向 client2 发送信息

```

来自client 4的信息:Hi client 2, this is client 4. Can you hear me?

>>Hi client 4, this is client 2. I can hear you.
来自client 4的信息:I can hear you too.

>>Exit
来自client 4的信息:Bye

```

图 65(b). client 2 向 client4 答复信息

我们看到服务器端显示了聊天的过程，如图 66 所示。

```

{'1': 62035, '2': 62036, '3': 62037, '4': 62038}
We server receive a message from client 4, and the content is 'Hi client 2, this is client 4. Can you hear me?', the terminal is 2
We server receive a message from client 2, and the content is 'Hi client 4, this is client 2. I can hear you.', the terminal is client 4
We server receive a message from client 4, and the content is 'I can hear you too.', the terminal is client 2
We server receive a message from client 2, and the content is 'Exit', the terminal is client 4
We server receive a message from client 4, and the content is 'Bye', the terminal is client 2

```

图 66. 服务器端体现交互信息

3.2 UDP 通信 Java 界面，如图 67 所示。

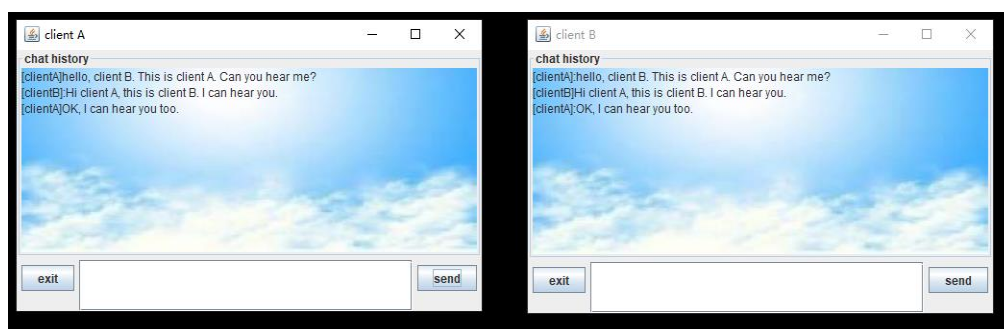


图 67. Java 实现界面体现交互信息

点击 send 可以发送信息，点击 exit 可以关闭窗口。

六、实验小结：

1. 通过 python 编程的方法，掌握了如何获取本机 IP 以及获取网站 IP 的方法。同时安装了 requests 库来进行爬虫功能的实现。爬虫方法有很多，类似正则表达式、xpath 以及 BeautifulSoup 等等，每一种方法都有其独到之处。但是，相比之下，正则表达式是一种更加万能的方法；
2. 本次实验利用 python 实现，实现了 TCP 指令交互、文件传输以及结合数据库，用 TCP 来实现查询的功能。此外，我们用 python 实现了客户端之间的 UDP 通信，并用 Java 图形用户界面(GUI)工具包 swing 中的面板容器类 JPanel 来进行了可视化功能；
3. 本次实验内容虽然很多，但是加强了 python 编程能力、加深了对 TCP/UDP 协议的理解，同时学会了用多种方法进行爬虫来爬取网页资源。

七、实验附件：

1. Socket 是整个 python 代码文件夹，其中包括 socket1.1、socket1.2、socket2.1、socket2.2：
 - 1) socket1.1 包含 getPCIP.py，用于获取本机 IP；getIP.py 用于获取网站 IP；SZU_fetcher.py 用于抓取深大主页文件。同时附加的文件夹都是抓好的深大主页文件，具体的在实验内容已经详细阐述；
 - 2) socket1.1 包含 fetch_ppt.py，这个是实验 4.1 的附加部分，用于抓取 ppt 文件；附加的文件夹是抓好的 ppt 文件压缩包；
 - 3) socket2.1 包含 client.py 和 server.py，这个是关于 TCP 指令交互的；client_file.py 和 server_file.py，这个是关于 TCP 文件交互的；
 - 4) socket2.2 包含 server_UDP.py、client_UDP.py、client_UDP_1.py、client_UDP_2.py 以及 client_UDP_3.py 这是一个 UDP 服务器与 4 个 UDP 客户端。分别运行就可以进行 UDP 通信。
2. UDPconnect 是一个 Java 工程文件，里面 src 文件夹中两个 java 文件模拟的是 UDP 两个客户端之间的沟通。

指导教师批阅意见：

成绩评定：

指导教师签字：邹永攀

年 月 日

备注：