Socket Programming Lab #1: Web Server

Full program for server.py provided below:

```python
# import socket module
from socket import *
import sys  # In order to terminate the program
def webServer(port=6789):
    serverSocket = socket(AF_INET, SOCK_STREAM)  # Prepare a server socket

    serverSocket.bind(('', port))  # Bind socket to specified port
    serverSocket.listen(1)  # Server listens for at most 1 TCP connection

    while True:
        # Establish the connection
        print('Ready to serve...')
        connectionSocket, addr = serverSocket.accept()  # Server accepts a connection and returns a new socket object
        try:
            message = connectionSocket.recv(1024).decode()  # Read bytes from socket
            filename = message.split()[1]
            f = open(filename[1:])
            outputdata = f.read()  # Retrieve specified file requested from client

            # Send one HTTP header line into socket
            connectionSocket.send('HTTP/1.1 200 OK\r\n\r\n'.encode())

            # Send the content of the requested file to the client
            for i in range(0, len(outputdata)):
                connectionSocket.send(outputdata[i].encode())

            connectionSocket.close()
        except IOError:
            # Send response message for file not found
            connectionSocket.send('HTTP/1.1 404 Not found\r\n\r\n'.encode())
            connectionSocket.send('File not found'.encode())

            # Close client socket
            connectionSocket.close()

    serverSocket.close()
    sys.exit()  # Terminate the program after sending the corresponding data

if __name__ == '__main__':
    webServer(6789)
```

After creating the server socket in line 5, we must set up the server socket to receive incoming TCP connections. In line 7 we bind the server socket to a specified port number, 6789. In line 8, we specify that the server will listen to at most 1 TCP connection before refusing any other incoming connections.

```
1  # import socket module
2  from socket import *
3  import sys  # In order to terminate the program
4  def webServer(port=6789):
5      serverSocket = socket(AF_INET, SOCK_STREAM)  # Prepare a server socket
6
7      serverSocket.bind(('', port))  # Bind socket to specified port
8      serverSocket.listen(1)  # Server listens for at most 1 TCP connection
```

In line 13, the server accepts a connection and, in return, creates a new socket object `connectionSocket` that can send and receive data on the connection. `addr` is the address bound to the socket on the other end of the connection.

```
10     while True:
11         # Establish the connection
12         print('Ready to serve...')
13         connectionSocket, addr = serverSocket.accept()
```

In line 15, we create a variable named `message` that will read in bytes of data received by `connectionSocket`.

```
14         try:
15             message = connectionSocket.recv(1024).decode()  #
```

In line 18, we create a variable named outputData to act as a buffer that will contain the contents of the file specified in the client request (from line 15).

```
16             filename = message.split()[1]
17             f = open(filename[1:])
18             outputdata = f.read()  # Retrieve
```

In line 21, we send a HTTP 200 OK header encoded as bytes from `connectionSocket` to the client.

```
20          # Send one HTTP header line into socket
21          connectionSocket.send('HTTP/1.1 200 OK\r\n\r\n'.encode())
22
23          # Send the content of the requested file to the client
24          for i in range(0, len(outputdata)):
25              connectionSocket.send(outputdata[i].encode())
26
27          connectionSocket.close()
```

If the client requests a file that does not exist in the server directory, then we want to return an error code. In line 30, we send a HTTP 404 header encoded as bytes from `connectionSocket` to the client. In line 31, we send a response message "File not found" from `connectionSocket` to the client.

```
28      except IOError:
29          # Send response message for file not found
30          connectionSocket.send('HTTP/1.1 404 Not found\r\n\r\n'.encode())
31          connectionSocket.send('File not found'.encode())
```

In line 34, after sending an error message if the client requests a file that does not exist, we close out `connectionSocket`.

```
33              # Close client socket
34              connectionSocket.close()
35
36      serverSocket.close()
37      sys.exit()  # Terminate the program after se
38
39  if __name__ == '__main__':
40      webServer(6789)
```