SOCKET PROGRAMMING

2023005439-Ronanki Trivikram

GITAM

Abstract

This program demonstrates basic client-server communication using Python's socket library. It comprises two components: a server and a client. The server listens for incoming connections on a specified port, accepts a single client connection, sends a predefined greeting message, and then closes the connection. The client connects to the server, receives the greeting message, and then terminates. This simple example illustrates the fundamental principles of socket programming, including socket creation, binding, listening, accepting, connecting, sending, and receiving data over a network. It serves as a foundational example for understanding more complex network applications.

## Program

A computer screen shot of a program

AI-generated content may be incorrect.

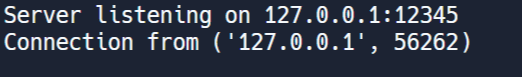
A blue and white background

AI-generated content may be incorrect.

## Client program

## A computer screen shot of a program AI-generated content may be incorrect.

## Output



A blue background with white text

AI-generated content may be incorrect.

## Conclusion

The provided Python program effectively demonstrates a basic client-server communication model using sockets. The start\_server() function initializes a server socket, binds it to a specified address and port, and listens for incoming connections. Upon accepting a client connection, it sends a simple "Hello from the server!" message and then closes both the client and server sockets. Conversely, the start\_client() function creates a client socket, connects to the server's address and port, receives the server's message, and closes the client socket.

**Function Breakdown:**

* **start\_server():**
  + Creates a server socket using socket.socket(socket.AF\_INET, socket.SOCK\_STREAM), specifying IPv4 and TCP.
  + Binds the socket to a local address and port using server\_socket.bind().
  + Enables the server to listen for connections using server\_socket.listen(1), allowing one queued connection.
  + Accepts an incoming connection using server\_socket.accept(), returning a client socket and address.
  + Sends a message to the connected client using client\_socket.send(message.encode('utf-8')), encoding the string to bytes.
  + Closes both the client and server sockets using client\_socket.close() and server\_socket.close().
* **start\_client():**
  + Creates a client socket using socket.socket(socket.AF\_INET, socket.SOCK\_STREAM).
  + Connects to the server using client\_socket.connect().
  + Receives data from the server using client\_socket.recv(1024), decoding the received bytes back to a string using .decode('utf-8').
  + Closes the client socket using client\_socket.close().

This program provides a fundamental understanding of how client-server interactions are established and managed through sockets, highlighting the core functions involved in network communication.

Thank you.