RAGULRAAJAN / SocketStudy

Code Pull requests Actions Projects Wiki Security Insights Settings

0 stars 565 forks 0 watching 1 Branch 0 Tags Activity

Public repository · Forked from NaliniG007/SocketStudy

1 Branch 0 Tags Go to file t Go to file + Add file Code ...

This branch is up to date with `NaliniG007/SocketStudy:main`. Contribute Sync fork

NaliniG007 Update README.md d2c61f5 · 7 months ago

README.md	Update README.md	7 months ago

README

Ex.No:1a Study of Socket Programming

Aim:

To perform a study on Socket Programming

Introduction:

```
Socket programming is a crucial aspect of network communication, allowing for data exchange
between computers over a network. It forms the backbone of various networked applications,
enabling communication between clients and servers. This study explores the fundamental
concepts of socket programming, its use cases, and provides a practical example to
demonstrate its implementation.
```

Understanding Socket Programming:

```
Socket programming involves the use of sockets, which serve as endpoints for communication.
A socket is identified by an IP address and a port number, and it facilitates data transfer
between a client and a server. The two main types of sockets are Stream Sockets, which
provide a reliable, connection-oriented communication, and Datagram Sockets, which are
connectionless and suitable for scenarios where reliability is less critical.
```

Key Concepts in Socket Programming:

1.Sockets • A socket is a software representation of a communication endpoint in a network. • It is identified by an IP address and a port number. • Sockets can be classified into two main types: Stream Sockets and Datagram Sockets. • Stream Sockets provide a reliable, connection-oriented communication, while Datagram Sockets are connectionless and operate in a best-effort mode.

2. Client-Server Model

• Socket programming typically follows the client-server model. • The server listens for incoming connections from clients, while clients initiate connections to the server. • Servers are passive, waiting for connection requests, and clients are active, initiating communication.

3, TCP/IP Protocol:

• Transmission Control Protocol (TCP) and Internet Protocol (IP) are the foundational protocols for socket programming. • TCP provides reliable, connection-oriented communication, ensuring data integrity and order. • IP facilitates the routing of data between devices in a network.

4.Basic Socket Functions:

• Socket programming involves a set of functions provided by the operating system or programming language to create, bind, listen, accept, connect, send, and receive data through sockets. • Examples of functions include socket(), bind(), listen(), accept(), connect(), send(), and recv().

Server-Side Operations:

• Servers create a socket using socket() and bind it to a specific IP address and port using bind(). • They then listen for incoming connections with listen() and accept connections with accept(). • Once a connection is establi • shed, servers can send and receive data using send() and recv().

Client –Server Operations

Clients create a socket using socket() and connect to a server using connect(). After establishing a connection, clients can send and receive data using send() and recv().

Use Cases of Socket Programming:

Socket programming finds applications in various domains, including web development, file transfer protocols, online gaming, and real-time communication. It is the foundation for protocols like HTTP, FTP, and SMTP, which power the internet. Socket programming enables the development of both server and client applications, facilitating the exchange of information between devices in a networked environment.

Example Use Cases:

1. Web servers: Web servers use socket programming to handle incoming HTTP requests from clients, serving web pages and content.
2. Chat Application: Instant messaging and chat applications use sockets to enable real-time communication between users.
3. File Transfer Protocol: Protocols like FTP (File Transfer Protocol) utilize socket programming for transferring files between a client and a server.

4. Networked Games: Online multiplayer games rely on socket programming to facilitate communication between game clients and servers.
5. RPC mechanisms: which allow processes to execute code on a remote server, often use socket programming for communication

Releases

No releases published
Create a new release

Packages

No packages published
Publish your first package