Day5 과제 2

20기 예비 김도현

UDP (User Datagram Protocol)

communication protocol

Internet protocol suite의 구성요소

Datagram의 packet을 통해 데이터를 전송한다.

OSI (Open Systems Interconnection) model – Transport layer

컴퓨터 네트워크의 계층 구조의 4단계 구성요소.

Application의 end-to-end communication을 제공한다.

Transport layer의 기능:

연결 지향 통신: 기본 연결을 제공하는 모델에서 연결을 데이터 스트림으로 해석한다.

동일 순서 전달: 데이터 패킷의 전송 순서를 보장하지 않기 때문에 데이터 패킷에 번호를 지정하여 수신자가 송신자가 지정한 순서에 맞춰 패킷을 해석할 수 있다.

신뢰성: 전송 중에 패킷이 손실되었을 때, 체크 섬과 같은 오류 검출 코드를 통해 데이터가 손상되었는지 확인하고, 송신자에게 올바른 수신을 확인하거나 자동 반복 요청 체계를 통해 손실되거나 손상된 데이터를 재전송한다.

흐름 제어: 송신자가 수신자의 데이터 버퍼를 초과하는 데이터를 전송할 때 발생하는 버퍼 오버런이나 버퍼 언더런을 방지한다.

혼잡 회피: 네트워크의 과도한 구독을 피하고 혼잡이 발생 시 자원 감소 단계를 통해 혼잡 붕괴를 피하고 트래픽 진입을 제어한다.

다중화: 단일 노드에서 여러 엔드 포인트를 제공할 수 있다.

UDP 는 다중화를 제외한 transport layer 의 기능을 지원하지 않기 때문에 규약비연결성과 비신뢰성을 가진다.

UDP 는 전송과정에서 연결과정이 없기 때문에 수신자의 수신여부와 관계없이 송신할 수 있다.

Checksum 을 제외한 오류 검출 과정이 없기 때문에 신뢰성을 보장하지 않는다.

8byte 의 고정 헤더를 가진다.

영상 스트리밍, 온라인 게임, 센서 데이터 전송 등 실시간성이 중요한 경우나 다음프레임으로 손실을 대체하는 등의 방법으로 데이터의 손실의 위험성이 낮은 않은 경우 적합하다.

빠른 전송이 필요할 때는 UDP 를 사용하고 손실이 치명적인 상황에서만 CRC8 또는 TCP 와 병행하여 전송하는 방법으로 보완할 수 있다.

UDP 는 0~65535 의 port 를 가진다.

0~1023: Well-known Port

1024~4915: Registered Port

49152~65535: Dynamic/Private Port

QUdpSocket

Qt에서 UDP 통신을 사용하기 위해 사용되는 라이브러리.

#include <QUdpSocket>

QUdpSocket(QObject *parent = nullptr)

생성자

~QUdpSocket()

소멸자

bool hasPendingDatagrams()

읽지 않은 datagram의 존재 여부를 반환.

bool joinMulticastGroup(const QHostAddress &groupAddress)

bool joinMulticastGroup(const QHostAddress &groupAddress, const

QNetworkInterface &iface)

interface에서 multicast groupAddress에 join. (default: 운영 체제의 기본 multicast group)

bool leaveMulticastGroup(const QHostAddress &groupAddress)

bool leaveMulticastGroup(const QHostAddress &groupAddress, const QNetworkInterface &iface)

interface의 지정된 multicast group에서 나간다.

QNetworkInterface multicastInterface()

발신 interface를 반환.

qint64 pendingDatagramSize()

pending중인 첫번째 datagram의 크기를 반환.

qint64 readDatagram(char *data, qint64 maxSize, QHostAddress *address = nullptr, quint16 *port = nullptr)

maxSize이내의 datagram을 수신하여 저장.

QNetworkDatagram receiveDatagram(qint64 maxSize = -1)

maxSize이내의 datagram을 수신하여 저장하고 호스트 주소와 포트를 반환.

void setMulticastInterface(const QNetworkInterface &iface)

interface를 발신 interface로 설정.

qint64 writeDatagram(const char *data, qint64 size, const QHostAddress &address, quint16 port)

qint64 writeDatagram(const QNetworkDatagram &datagram)

qint64 writeDatagram(const QByteArray &datagram, const QHostAddress &host, quint16 port)

datagram 전송.

QAbstractSocket

Public types

enum BindFlag { ShareAddress, DontShareAddress, ReuseAddressHint, DefaultForPlatform }

bind() 동작 제어.

flags BindMode

Bind mode 설정.

enum NetworkLayerProtocol { IPv4Protocol, IPv6Protocol, AnyIPProtocol, UnknownNetworkLayerProtocol }

사용할 네트워크 지정.

enum PauseMode { PauseNever, PauseOnSslErrors }

전송 중지 조건 정의.

flags PauseModes

pause 조건 설정.

enum SocketError { ConnectionRefusedError, RemoteHostClosedError, HostNotFoundError, SocketAccessError, SocketResourceError, ..., UnknownSocketError }

오류 유형 정의.

enum SocketOption { LowDelayOption, KeepAliveOption, MulticastTtlOption, MulticastLoopbackOption, TypeOfServiceOption, ..., PathMtuSocketOption }

운영 체제 옵션 정의.

enum SocketState { UnconnectedState, HostLookupState, ConnectingState, ConnectedState, BoundState, ..., ListeningState }

소켓의 연결 상태.

enum SocketType { TcpSocket, UdpSocket, SctpSocket, UnknownSocketType }

전송 프로토콜 정의.

Public functions

QAbstractSocket(QAbstractSocket::SocketType socketType, QObject *parent)

생성자

~QAbstractSocket()

소멸자

void abort()

연결을 중단하고 버퍼 삭제후 소켓 재설정.

bool bind(const QHostAddress &address, quint16 port = 0, QAbstractSocket::BindMode mode = DefaultForPlatform)

bool bind(quint16 port = 0, QAbstractSocket::BindMode mode = DefaultForPlatform)

bool bind(QHostAddress::SpecialAddress addr, quint16 port = 0, QAbstractSocket::BindMode mode = DefaultForPlatform)

주소와 포트에 소켓을 바인딩.

void connectToHost(const QString &hostName, quint16 port, QIODeviceBase::OpenMode openMode = ReadWrite, QAbstractSocket::NetworkLayerProtocol protocol = AnyIPProtocol)

void connectToHost(const QHostAddress &address, quint16 port, QIODeviceBase::OpenMode openMode = ReadWrite)

호스트에 연결.

void disconnectFromHost()

소켓 연결 종료.

QAbstractSocket::SocketError error() const

마지막 소켓 오류 반환.

bool flush()

버퍼의 데이터 전송.

bool isValid() const

소켓이 유효성 확인.

QHostAddress localAddress() const

로컬 소켓의 IP 주소 반환.

quint16 localPort() const

로컬 소켓의 포트 번호 반환.

QAbstractSocket::PauseModes pauseMode() const

현재 설정된 pause 모드 반환.

QHostAddress peerAddress() const

연결된 상대방의 IP 주소 반환.

QString peerName() const

연결된 상대방의 호스트 이름 반환.

quint16 peerPort() const

연결된 상대방의 포트 번호 반환.

QString protocolTag() const

소켓에 설정된 프로토콜 태그 반환.

QNetworkProxy proxy() const

현재 설정된 네트워크 프록시 반환.

qint64 readBufferSize() const

읽기 버퍼의 크기 반환.

void resume()

pause 상태에서 데이터 전송 재개.

void setPauseMode(QAbstractSocket::PauseModes pauseMode)

데이터 전송 pause 조건 설정.

void setProtocolTag(const QString &tag)

protocol tag 설정.

void setProxy(const QNetworkProxy &networkProxy)

소켓에 사용할 프록시 설정.

void setReadBufferSize(qint64 size)

read buffer 크기 설정.

bool setSocketDescriptor(qintptr socketDescriptor,
QAbstractSocket::SocketState socketState = ConnectedState,
QIODeviceBase::OpenMode openMode = ReadWrite)

기존 소켓의 descriptor로 소켓 초기화.

void setSocketOption(QAbstractSocket::SocketOption option, const

QVariant &value)

소켓 옵션 설정.

qintptr socketDescriptor() const

소켓의 descriptor 반환.

QVariant socketOption(QAbstractSocket::SocketOption option)

소켓 옵션 값 반환.

QAbstractSocket::SocketType socketType() const

소켓 타입 반환.

QAbstractSocket::SocketState state() const

소켓의 현재 상태 반환.

bool waitForConnected(int msecs = 30000)

지정된 시간 동안 연결 완료 대기.

bool waitForDisconnected(int msecs = 30000)

지정된 시간 동안 연결 종료 대기.

qint64 bytesAvailable() const

읽을 수 있는 bytes 반환.

qint64 bytesToWrite() const

전송 대기 중인 bytes 반환.

void close()

소켓을 닫고 연결 종료.

bool isSequential() const

데이터가 순차적으로 처리되는지 판단.

bool waitForBytesWritten(int msecs = 30000)

데이터가 전송될 때까지 대기.

bool waitForReadyRead(int msecs = 30000)

데이터를 수신할 때까지 대기.

qint64 writeData(const char *data, qint64 size)

소켓에 데이터 기록.

Reimplemented Public Functions

virtual qint64 bytesAvailable() const override

Reimplements: QIODevice::bytesAvailable() const.

읽을 수 있는 bytes 반환.

virtual qint64 bytesToWrite() const override

Reimplements: QIODevice::bytesToWrite() const.

Write 대기 중인 bytes 반환.

virtual void close() override

Reimplements: QIODevice::close().

aboutToClose() 시그널을 발생후 종료하고 OpenMode NotOpen으로 설정.

virtual bool isSequential() const override

Reimplements: QIODevice::isSequential() const.

순차적인지 판단.

virtual bool waitForBytesWritten(int msecs = 30000) override

Reimplements: QIODevice::waitForBytesWritten(int msecs).

지정된 시간 동안 데이터가 기록될 때까지 대기.

virtual bool waitForReadyRead(int msecs = 30000) override

Reimplements: QIODevice::waitForReadyRead(int msecs).

지정된 시간 동안 데이터가 도착할 때까지 블로킹.

Signals

void connected()

연결 완료 시.

void disconnected()

연결을 종료 시.

void errorOccurred(QAbstractSocket::SocketError socketError)

소켓 오류 발생 시.

void hostFound()

호스트 이름 조회 시.

void proxyAuthenticationRequired(const QNetworkProxy &proxy, QAuthenticator *authenticator)

프록시 인증이 요구 시.

void stateChanged(QAbstractSocket::SocketState socketState)

소켓 상태 변경 시.

Protected Functions

void setLocalAddress(const QHostAddress &address)

로컬 주소 설정.

void setLocalPort(quint16 port)

로컬 포트 설정.

void setPeerAddress(const QHostAddress &address)

상대방 주소 설정.

void setPeerName(const QString &name)

상대방 호스트 이름 설정.

void setPeerPort(quint16 port)

상대방 포트 설정.

void setSocketError(QAbstractSocket::SocketError socketError)

소켓 오류 상태 설정.

void setSocketState(QAbstractSocket::SocketState state)

소켓 상태 설정.

Reimplemented Protected Functions

qint64 readData(char *data, qint64 maxSize)

Reimplements: QIODevice::readData(char *data, qint64 maxSize).

버퍼의 데이터 읽음

qint64 readLineData(char *data, qint64 maxlen)

Reimplements: QIODevice::readLineData(char *data, qint64 maxSize).

한 줄 저장하고 '₩0'추가

qint64 skipData(qint64 maxSize)

Reimplements: QIODevice::skipData(qint64 maxSize).

maxSize까지 이동.

qint64 writeData(const char *data, qint64 size)

Reimplements: QIODevice::writeData(const char *data, qint64 maxSize).

maxSize까지 write.

적용

연결 설정

```
udpSocket = new QUdpSocket(this);
udpSocket->bind(QHostAddress::Any, 포트 번호);
connect(udpSocket, &QUdpSocket::readyRead, this,
&MainWindow::get_udp);
```

수신

```
while (udpSocket->hasPendingDatagrams())
{
    QByteArray datagram;
    datagram.resize(int(udpSocket->pendingDatagramSize()));
    QHostAddress sender;
    quint16 senderPort;

    udpSocket->readDatagram(datagram.data(), datagram.size(),
&sender, &senderPort);
}
```

송신

udpSocket->writeDatagram(**데이터**, QHostAddress("**IP**"), **포트 번호**);

출처

https://en.wikipedia.org/wiki/User_Datagram_Protocol

https://en.wikipedia.org/wiki/Transport_layer

https://doc.qt.io/qt-6/qudpsocket.html#QUdpSocket

https://doc.qt.io/qt-6/qabstractsocket.html

https://doc.qt.io/qt-6/qiodevice.html