public abstract class AbstractConnection implements NIOConnection {

public AbstractConnection(NetworkChannel channel) {

this.channel = channel;

boolean isAIO = (channel instanceof AsynchronousChannel);

if (isAIO) {

socketWR = new AIOSocketWR(this);

} else {

socketWR = new NIOSocketWR(this);

}

this.isClosed = new AtomicBoolean(false);

this.startupTime = TimeUtil.currentTimeMillis();

this.lastReadTime = startupTime;

this.lastWriteTime = startupTime;

}

public AbstractConnection() {

boolean isAIO = (channel instanceof AsynchronousChannel);

if (isAIO) {

socketWR = new AIOSocketWR(this);

} else {

socketWR = new NIOSocketWR(this);

}

this.isClosed = new AtomicBoolean(false);

this.startupTime = TimeUtil.currentTimeMillis();

this.lastReadTime = startupTime;

this.lastWriteTime = startupTime;

}

public String getCharset()

public boolean setCharset(String charset)

public boolean isSupportCompress()

public void setSupportCompress(boolean isSupportCompress)

public int getCharsetIndex()

public long getIdleTimeout()

public SocketWR getSocketWR()

public void setIdleTimeout(long idleTimeout)

public int getLocalPort()

public String getHost()

public void setHost(String host)

public int getPort()

public void setPort(int port)

public void setLocalPort(int localPort)

public long getId()

public void setId(long id)

public boolean isIdleTimeout()

public NetworkChannel getChannel()

public int getPacketHeaderSize()

public void setPacketHeaderSize(int packetHeaderSize)

public int getMaxPacketSize()

public void setMaxPacketSize(int maxPacketSize)

public long getStartupTime()

public long getLastReadTime()

public void setProcessor(NIOProcessor processor)

public long getLastWriteTime()

public long getNetInBytes()

public long getNetOutBytes()

public int getWriteAttempts()

public NIOProcessor getProcessor()

public ByteBuffer getReadBuffer()

public ByteBuffer allocate()

public final void recycle(ByteBuffer buffer)

public void setHandler(NIOHandler handler)

public void handle(byte[] data)

public void register() throws IOException

public void asynRead() throws IOException

public void doNextWriteCheck() throws IOException

public void onReadData(int got) throws IOException

private boolean isConReadBuffer(ByteBuffer buffer)

private ByteBuffer ensureFreeSpaceOfReadBuffer(ByteBuffer buffer, int offset, final int pkgLength)

private ByteBuffer compactReadBuffer(ByteBuffer buffer, int offset)

public void write(byte[] data)

private final void writeNotSend(ByteBuffer buffer)

public final void write(ByteBuffer buffer)

public ByteBuffer checkWriteBuffer(ByteBuffer buffer, int capacity, boolean writeSocketIfFull)

public ByteBuffer writeToBuffer(byte[] src, ByteBuffer buffer)

public void close(String reason)

public boolean isClosed()

public void idleCheck()

protected void cleanup()

protected int getPacketLength(ByteBuffer buffer, int offset)

public ConcurrentLinkedQueue<ByteBuffer> getWriteQueue()

private void closeSocket()