Client
<<Java Class>>
Client
(default package)

- △ sendPacket: DatagramPacket
- △ receivePacket: DatagramPacket
- △ sendReceiveSocket: DatagramSocket
- ◘ⁱ Input: Scanner
- △ filename: String
- ◖ⁱ fileDirectory: String
- △ mode: String
- ◘ in: BufferedInputStream
- ◖ⁱ MAX_DATA: int
- △ addr: InetAddress
- △ port: int

- ● Client()
- ●ˢ main(String[]):void
- ● ui():void
- ● connection():void
- ● readReq():void
- ● writeReq():void
- ●ˢ createRequest(byte,String,String):byte[]
- ● createAck(byte):byte[]
- ● createData(byte,byte[]):byte[]
- ● createError(byte,String):byte[]
- ● parseAck(byte[]):void
- ● parseData(byte[]):byte[]
- ● parseError(byte[]):void
- ● send(byte[],InetAddress,int):void
- ● receive():DatagramPacket
- ● processDatagram(DatagramPacket):byte[]
- ● getOpcode(DatagramPacket):Opcode
- ● isValidPacket(DatagramPacket):boolean

<<Java Class>>
ErrorSim
(default package)

- ◘ sendPacket: DatagramPacket
- ◘ receivePacket: DatagramPacket
- ◘ receiveSocket: DatagramSocket
- ◘ sendReceiveSocket: DatagramSocket
- △ addr: InetAddress
- ◘ˢ port: int
- ◖ⁱ MAX_DATA: int

- ● ErrorSim()
- ● UI():void
- ● ErrsimQuit():void
- ● receivePacket():DatagramPacket
- ● processDatagramPacket(DatagramPacket):byte[]
- ● sendPack(byte[]):void
- ● createError5():void
- ● genericError(String,String):void
- ● sendError(DatagramPacket):void
- ●ˢ main(String[]):void

<<Java Class>>
serverConnection
(default package)

- ◘ receivePacket: DatagramPacket
- ◘ sendPacket: DatagramPacket
- ◘ sendReceiveSocket: DatagramSocket
- ◘ sendSocket: DatagramSocket

- ● serverConnection(DatagramPacket,DatagramPa
- ● run():void

<<Java Class>>
Server
(default package)

- △ receivePacket: DatagramPac
- △ˢ receiveSocket: DatagramSoc
- ◖ⁱ MAX_DATA: int

- ● Server()
- ●ˢ main(String[]):void
- ●ˢ getSocket():DatagramSocket
- ● listener():void
- ● makeConnection(DatagramP
- ● makeConnection(DatagramP
- ● receive():DatagramPacket
- ● processDatagram(DatagramI
- ● getOpcode(DatagramPacket)
- ● isValidPacket(DatagramPack
- ● parseError(byte[]):void
- ● createError(byte,String):byte[

<<Java Class>>
UserInput
(default package)

- ◘ input: Scanner

- ● UserInput()
- ● run():void

<<Java Class>>
ClientConnection
(default package)

- ◖ⁱ MAX_DATA: int
- △ requestPacket: DatagramPacket
- △ sendPacket: DatagramPacket
- △ receivePacket: DatagramPacket
- △ sendReceiveSocket: DatagramSocket
- ◖ⁱ fileDirectory: String
- △ filename: String
- △ addr: InetAddress
- △ port: int
- △ error: byte[]
- ◘ in: BufferedInputStream

- ● ClientConnection(DatagramPacket)
- ● ClientConnection(DatagramPacket,byte[])
- ● run():void
- ● readReq():void
- ● writeReq():void
- ● closeConnection():void
- ● getOpcode(DatagramPacket):Opcode
- ● getFilename(DatagramPacket):String
- ● send(byte[]):void
- ● receive():DatagramPacket
- ● processDatagram(DatagramPacket):byte[]
- ● parseAck(byte[]):void
- ● parseData(byte[]):byte[]
- ● parseError(byte[]):void
- ● createAck(byte):byte[]
- ● createData(byte,byte[]):byte[]

<<Java Enumeration>>
Opcode
(default package)

- ◖ⁱ RRQ: Opcode
- ◖ⁱ WRQ: Opcode
- ◖ⁱ DATA: Opcode
- ◖ⁱ ACK: Opcode
- ◖ⁱ ERROR: Opcode
- ◘ⁱ op: byte

- ▲ Opcode(byte)
- ■ op():byte

<<Java Enumeration>>
Opcode
(default package)

- ◖ⁱ RRQ: Opcode
- ◖ⁱ WRQ: Opcode
- ◖ⁱ ACK: Opcode
- ◖ⁱ DATA: Opcode
- ◖ⁱ ERROR: Opcode

- ● Opcode()

<<Java Enumeratio
Opcode
(default package)

- ◖ⁱ RRQ: Opcode
- ◖ⁱ WRQ: Opcode
- ◖ⁱ ACK: Opcode
- ◖ⁱ DATA: Opcode
- ◖ⁱ ERROR: Opcod

- ● Opcode()

~op
0..1

cket,DatagramSocket)

>

cket
cket

acket):void
acket,byte[]):void

Packet):byte[]
):Opcode
et):boolean

]

on>>

de

createError(byte,String):byte[]

isValidPacket(DatagramPacket):boolean