LogWriterInterface +writeLogMessage(in message : String) Handler -connectionHandler : ConnectionHandler Runnable O May 9, 2013 -requestLogRecord: RequestLogRecord connectionHandlerSerialNumber : long LogWriterInterface -connectionHandlerMessageCount : long +Handler(in clientSocket : Socket) +run() +writeLogMessage(in message : String) \oplus ConnectionHandler -serialNumberCounter : long = 0 -clientConnection : Socket RequestLogRecord -serialNumber : long -timeInMilliseconds: long -logWriter: LogWriterInterface +ConnectionHandler(in clientConnection : Socket, in logWriter : LogWriterInterface) +RequestLogRecord(in clientSocket : Socket) +setMessage(in message : String) +getSerialNumber(): long +setDateTimeStampToNow() Server -portNumber : int Check all classes for Runnable -serverIsRunning : boolean serverLog : Log synchronization needs serverName : String serverName is final +Server(in portNumber : int, in log : Log) +serverIsRunning(): boolean +startServer() +stopServer() +getPort(): int Log LogRecord -isClosed : boolean -logFile : PrintWriter -fields : String[] -recordNumber : long +LogRecord(in numberOfFields : int) +Log(in logDirectory : File) +getFieldSeparator(): String +Log(in logDirectory : File, in sleepAmount : int) +setField(in fieldNumber : int, in data : String) +close() +getAsString() : String getNow(): String +writeToLog(in logFile : Log) +isClosed(): boolean writeLogRecord(in logRecord : LogRecord) : String ServerLogRecord +MessageType : enum **EchoingLog** +ServerLogRecord() +setServerName(in serverName : String) +setClientIPNumber(in clientIPNumber : String) +EchoingLog(in logDirectory : File) +setClientPort(in clientPort : int) +EcholingLog(in logDirectory : File, in sleepAmount : int) +setMessage(in message : String) +writeLogRecord(in logRecord : LogRecord) : String +setMessageType(in messageType : enum) EchoingLog class causes the log record to be written The enum in ServerLogRecord has a both to the log file and the name and an integer code. It also has a monitor. constructor and a private method: getMessageCode that returns the integer LogFlusher code associated with the name.

-recordCount : long -sleepAmount : int

Runnable O

+LogFlusher(in sleepAmount : int)

«interface»

All methods in the Agent class must advertise the applicable exceptions except the close methods and the abstract methods. The close methods and abstract methods do not advertise. Agent -serialVersionUID : long = 1 +Agent() May 9, 2013 Serializable +readObjectFrom(in socket : Socket) : Object +readObjectFrom(in in : InputStream) : Object **New Classes:** +writeMyselfTo(in socket : Socket) PlaySelectedSong, +writeMyselfTo(in out : OutputStream) See Next Page for more new classes +closeSocket(in socket : Socket) +closeEverything(in socket : Socket) +closeInputStream(in in : InputStream) +closeOutputStream(in out : OutputStream) +getInputStreamFrom(in socket): InputStream +getOutputStreamFrom(in socket : Socket) : OutputStream +clientSideRun(in socket : Socket) +serverSideRun(in socket : Socket, in logWriter : LogWriterInterface) MediaCatalog is implemented as a Singleton MediaCatalog -mediaCatalog : MediaCatalog = null -catalog: Hashtable TestAgent -MediaCatalog() -serialVersionUserId : long = 1 +getInstance(): MediaCatalog \oplus +newInstance(): MediaCatalog -message : String +loadSongsFromFile(in dataFile : File) +TestAgent(in message : String) +getSong(in songld : String) : Song +clientSideRun(in socket : Socket) +getSongs() : Song[] +serverSideRun(in socket : Socket, in logWriter : LogWriterInterface) getRandomSong(): Song JukeBox is the driver for the system. IllegalAgurmentException Based on the values in the xml Properites file, it starts the server and log. **JukeBox** InvalidFieldDataException +main(in args : String[]) +InvalidFieldDataException() +InvalidFieldDataException(in message : String) Sona -serialVersionUID : long = 1 -serialNumber : int = 0 Serializable O--artistName : String encodingMethod : String filePath: File songld: int songTitle : String +Song(in songTitle: String, in artistName: String, in encodingMethod: String, in filePath: File) +getArtist(): String getSongTitle(): String getSongld(): String getEncodingMethod(): String -getInputStream(): InputStream -initializeSongId() The file containing the song information has one record per song. Each record contains the following fields: song title, artist name, encoding **PlaySong** (WAV. MP3, etc), and file path -serialVersionUID : long = 1 **PlavSelectedSong** song : Song -serialVersionUID : long = 1 error : String +PlaySelectedSong(in song : Song) +PlaySong() +serverSideRun(in socket : Socket, in logWriter : LogWriterInterface) PlaySong(in song : Song) +errorWasGenerated(): boolean Overriding method in PlaySelectedSong calls super.setSong passing +getSong(): Song it the matching song from the MediaCatalog and then calls the super +setSong(in song : Song) class serverSideRun method. +clientSideRun(in socket : Socket) +serverSideRun(in socket : Socket, in logWriter : LogWriterInterface) **PlayRandomSong** -serialVersionUID : long = 1 See clientSideRun and serverSideRun flowcharts +PlayRandomSong()

Overriding method in PlayRandomSong calls super.setSong passing it a random song from the MediaCatalog and then calls the super class serverSideRun method.

+serverSideRun(in socket : Socket, in logWriter : LogWriterInterface)

