Class Diagram

Floor

+ floorNum: int
+ numElevators: int
+ upButton: boolean
+ downButton: boolean
+ isUpperFloor boolean
+ isUpperFloor: boolean
+ lampDirection: Direction

+ Floor(int, int, boolean, boolean)
+ getElevNum: int
+ getFloorNum: int
+ getDirection(): Direction
+ setUpLampOn: void
+ setDownLampOn: void
+ setUpLampOff: void
+ setDownLampOff: void
+ getUpLampStatus(): boolean
+ getDownLampStatus(): boolean

<< enumeration >>
Direction

UP
DOWN
IDLE

<< Java Class>>
FloorSubsytem

+ sendPacket: DatagramPacket
+ receivePacket: DatagramPacket
+ sendReceivePacket: DatagramSocket
+ ACK: int
+ CMD: int
+ DATA: int
+ ERROR: int
+ FLOORPORT: int
+ SCHEDPORT: int
+ numFloors: int
+ upperFloor: int
+ lowerFloor: int
+ floorMin: int
+ floorTotal: int
+ elevatorTotal: int
+ floors: ArrayList<Floor>
+ requestCount: int

+ FloorSubsystem()
+ Direction: enumeration
+ createPacketData(int, String): byte
+ readPacketData(byte[]): String
+ receive(DatagramSocket, byte[]): DatagramPacket
+ createServiceRequest(String, int, int, Direction): byte
+ send(byte[], int, DatagramSocket): void
+ sendServiceRequest(String, int, int, Direction, DatagramSocket): void
+ cmdRequest(String, int): void
+ running(): void

<<Java Class>>
Scheduler

+ ACK: String
+ CMD: String
+ DATA: String
+ ERROR: String
+ FLOOR_BUTTON: String
+ ELEVATOR_ARRIVED: String
+ UP_PICKUP:String
+ UP_DROPOFF: String
+ DOWN_PICKUP: String
+ DOWN_DROPOFF: String
+ DOOR_OPEN: String
+ DOOR_CLOSE: String
+ STOP: String
+ currentFloor: int
+ HOSTPORT: int
+ EPORT1: int
+ EPORT2: int
+ EPORT3: int
+ EPORT4: int
+ FPORT: int
+ hostSocket: DatagramSocket
+ floorSocket: DatagramSocket
+ FLOORS: int
- upQ1: BlockingQueue<String>
- downQ: BlockingQueue<String>
- upQ2: BlockingQueue<String>
- downQ2: BlockingQueue<String>
- upQ3: BlockingQueue<String>
- downQ3: BlockingQueue<String>
- upQ4: BlockingQueue<String>
- downQ4: BlockingQueue<String>
- handler1: ElevatorHandler
- handler2: ElevatorHandler
- handler3: ElevatorHandler
- handler4: ElevatorHandler

+ Scheduler()
+ handleFloorCommand(String, int): void
+ calculateSuitability(int, int, int, String, String): int
+ parsePacket(byte[]): String
+ createPacket(String, String, int): DatagramPacket

<< Java Class>>
ElevatorControl

- ACK: String
- CMD: String
- DATA: String
- UP_PICKUP: String
- UP_DROPOFF: String
- DOWN_PICKUP: String
- DOWN_DROPOFF: String
- DOOR_OPEN: String
- DOOR_CLOSE: String
- STOP: String
- aport: int
- MAX_FLOOR: int
- num_lamp: int
- send: int
- s_elevator: int
+ Elv_1: ElevatorControl
+ Elv_2: ElevatorControl
+ Elv_3: ElevatorControl
+ Elv_4: ElevatorControl
- num_elevator: int
- sendSocket: DatagramSocket
- receiveSocket: DatagramSocket
- sendPacket: DatagramPacket
- sendAPacket: DatagramPacket
- receivePacket: DatagramPacket
- elevator: Elevator

+ ElevatorControl()
+ createPacket(String, String, int): DatagramPacket
+ packetToString(byte[]): String
+ toInt(String): int
+ control(): void
+ main(String): void

UserInputHandler

+ requestSocket: DatagramSocket
+ serviceReqList: ArrayList<String>
+ serviceReqs:ArrayList<byte[]>
+ fs: FloorSubsystem

+ UserInputHandler(FloorSubsystem,)
+ parseInputFile(String, ArrayList<String>: void
+ parseServiceReqs(ArrayList<String>): void
+ run(): void

<< Interface >>
Runnable

+ run(): void

ElevatorHandler

- eSocket: DatagramSocket
- eport: int
- fport: int
- upQ: BlockingQueue<String>
- downQ: BlockingQueue<String>
currentDirection: String
currentFloor: int
currentDestination: int
id: int

+ ElevatorHandler(int, int, BlockingQueue<String>, BlockingQueue<String>, int, String, int)
+ run(): void
+ performPickup(String, String): void
+ performDropOff(String, String): void
+ sendPositionToFloor(String): void

<< Interface >>
Thread

+ run(): void

<<Java Class>>
Elevator

+ currentFloor: String
+ direction: ElevatorDirection
- MAX_FLOOR: int
+ Lamp: int[]

+ Elevator(String, ElevatorDirection)
+ getCurrentFloor(): String
+ getIntFloor(): int
+ getDirection(): ElevatorDirection
+ getStatus(): ElevatorStatus
+ open(int): void
+ close(int): void

<< enumeration >>
ElevatorDirection

E_UP
E_DOWN
E_HOLD

<< Interface >>
Runnable

+ run(): void

<< enumeration >>
ElevatorStatus

E_IN_USE
E_EMPTY