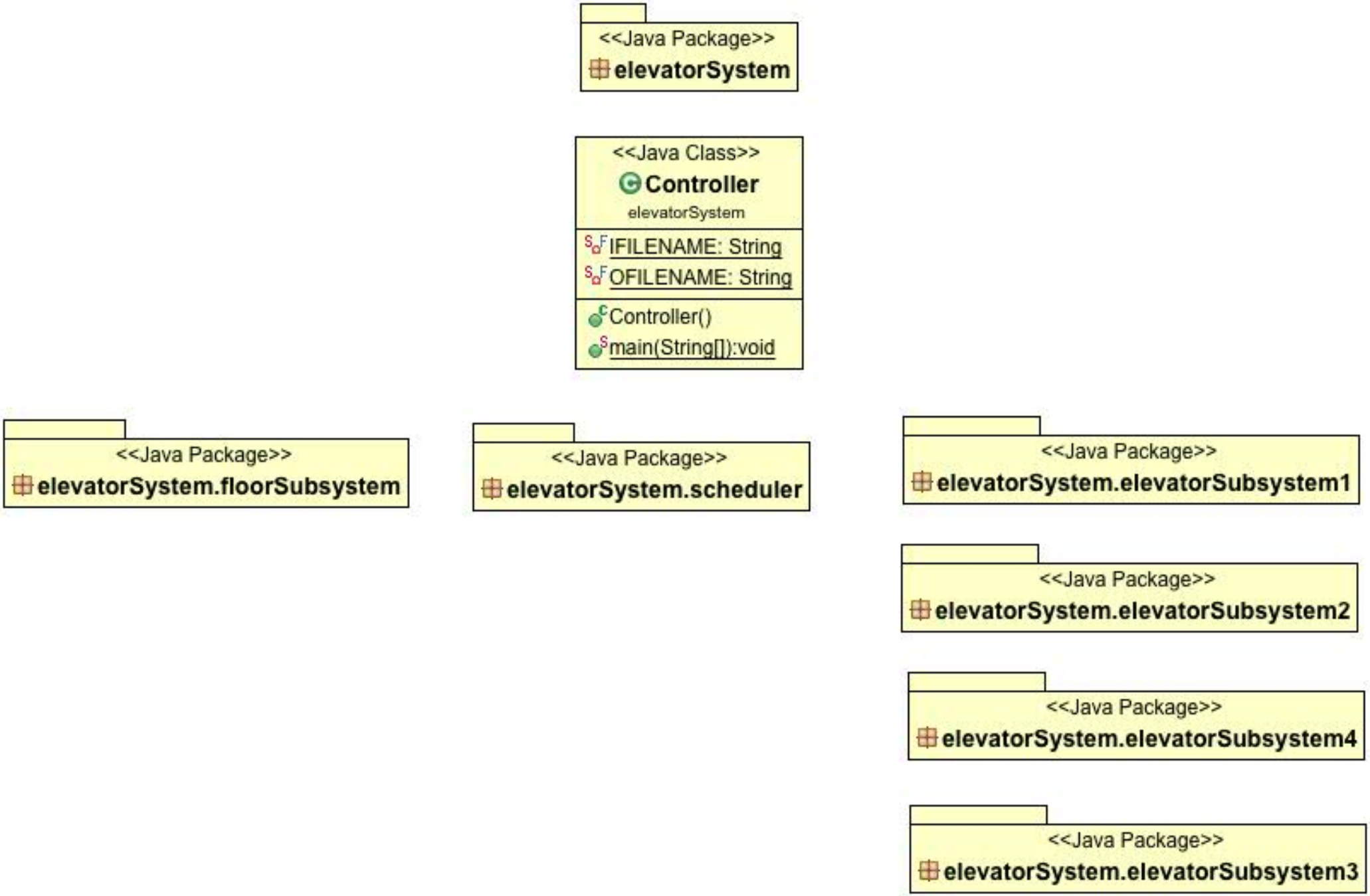
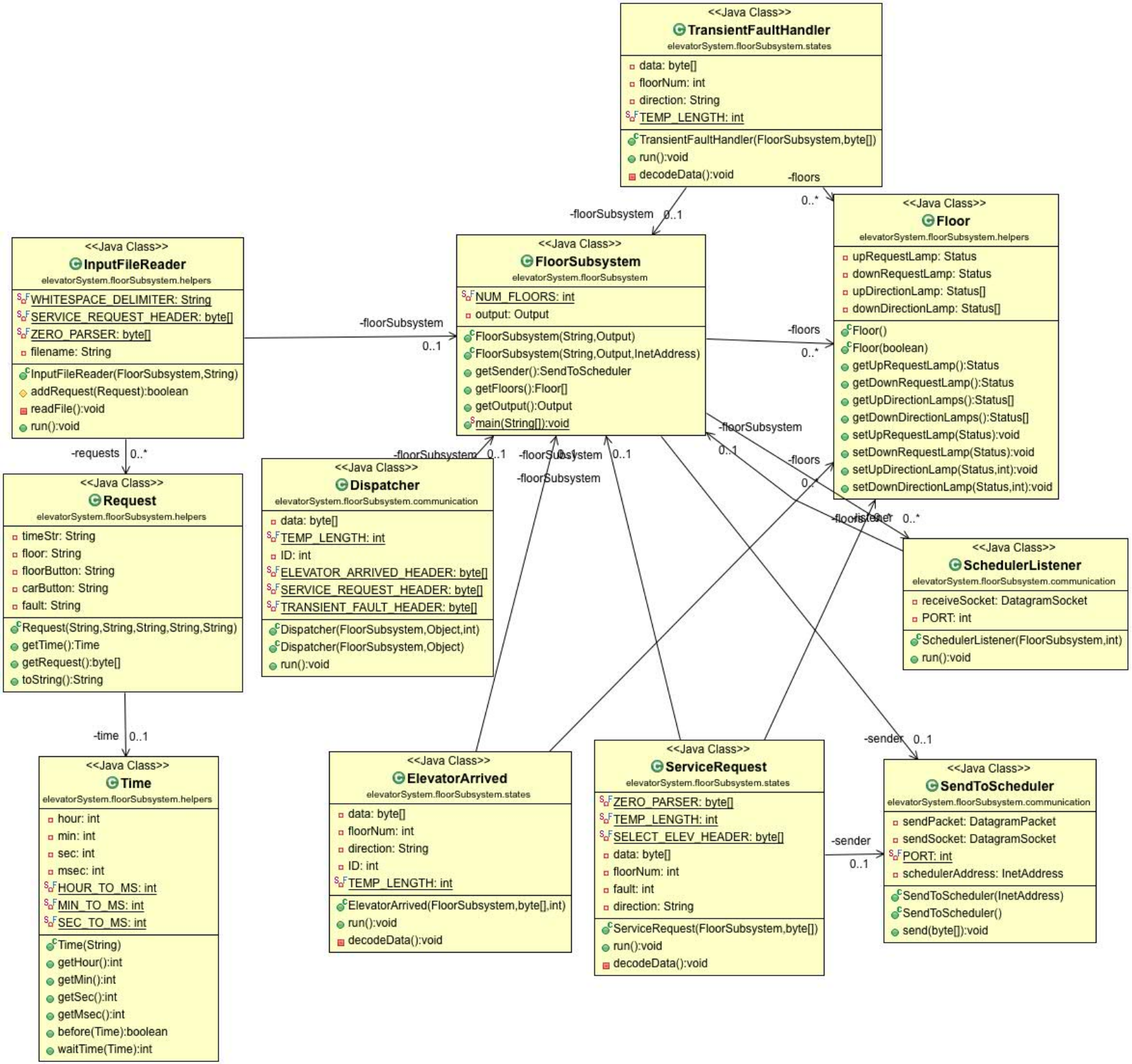
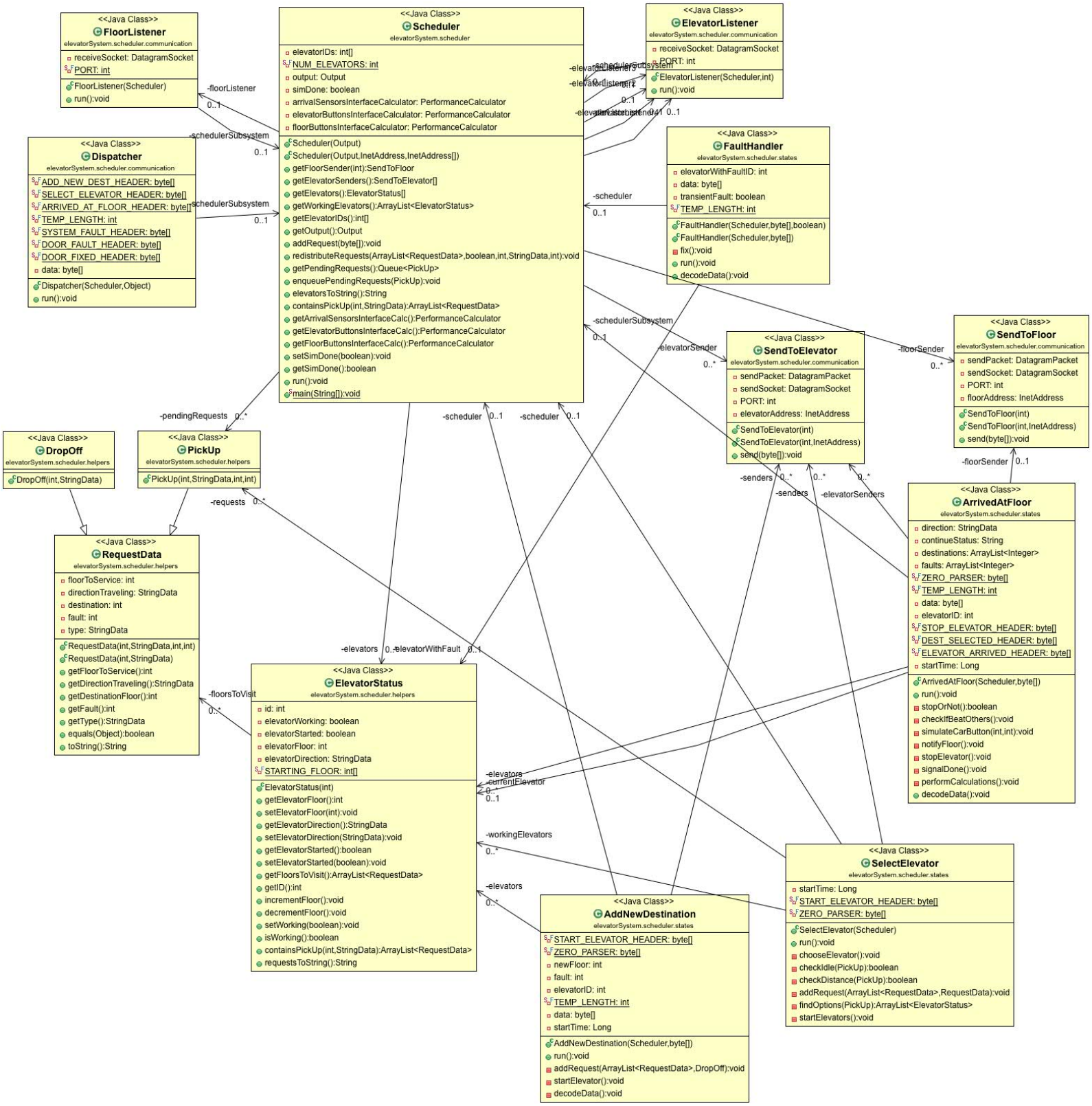
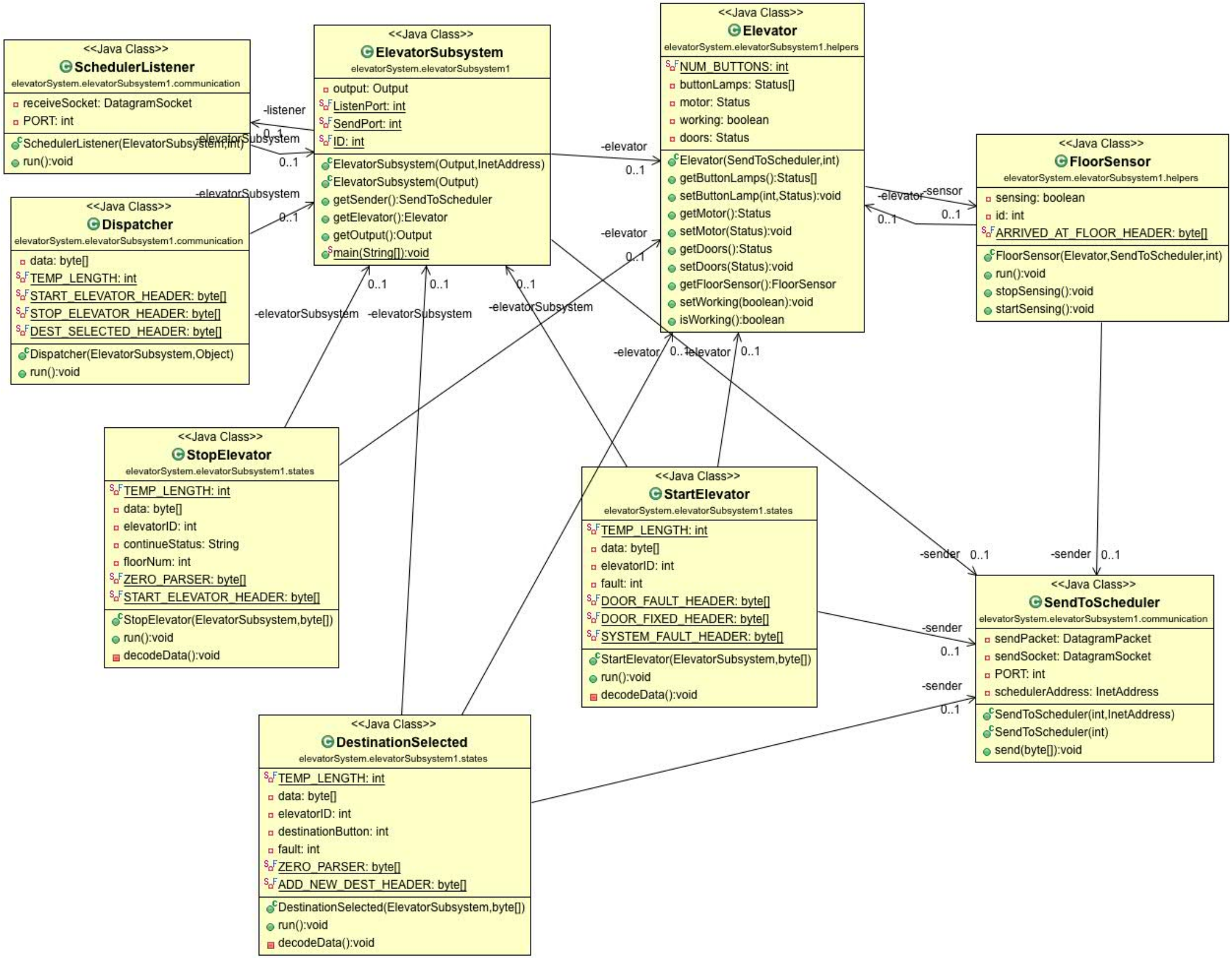

<<Java Package>>
elevatorSystem
<<Java Class>>
Controller
elevatorSystem
IFILENAME: String
OFILENAME: String
Controller()
main(String[]):void
<<Java Package>>
elevatorSystem.floorSubsystem
<<Java Package>>
elevatorSystem.scheduler
<<Java Package>>
elevatorSystem.elevatorSubsystem1
<<Java Package>>
elevatorSystem.elevatorSubsystem2
<<Java Package>>
elevatorSystem.elevatorSubsystem4
<<Java Package>>
elevatorSystem.elevatorSubsystem3


<<Java Class>>
TransientFaultHandler
elevatorSystem.floorSubsystem.states
data: byte[]
floorNum: int
direction: String
TEMP_LENGTH: int
TransientFaultHandler(FloorSubsystem,byte[])
run():void
decodeData():void
-floors
0..*
-floorSubsystem
0..1
<<Java Class>>
Floor
elevatorSystem.floorSubsystem.helpers
upRequestLamp: Status
downRequestLamp: Status
upDirectionLamp: Status[]
downDirectionLamp: Status[]
Floor()
Floor(boolean)
getUpRequestLamp():Status
getDownRequestLamp():Status
getUpDirectionLamps():Status[]
getDownDirectionLamps():Status[]
setUpRequestLamp(Status):void
setDownRequestLamp(Status):void
setUpDirectionLamp(Status,int):void
setDownDirectionLamp(Status,int):void
<<Java Class>>
InputFileReader
elevatorSystem.floorSubsystem.helpers
WHITESPACE_DELIMITER: String
SERVICE_REQUEST_HEADER: byte[]
ZERO_PARSER: byte[]
filename: String
InputFileReader(FloorSubsystem,String)
addRequest(Request):boolean
readFile():void
run():void
-floorSubsystem
0..1
<<Java Class>>
FloorSubsystem
elevatorSystem.floorSubsystem
NUM_FLOORS: int
output: Output
FloorSubsystem(String,Output)
FloorSubsystem(String,Output,InetAddress)
getSender():SendToScheduler
getFloors():Floor[]
getOutput():Output
main(String[]):void
-floors
0..*
-requests
0..*
<<Java Class>>
Request
elevatorSystem.floorSubsystem.helpers
timeStr: String
floor: String
floorButton: String
carButton: String
fault: String
Request(String,String,String,String,String)
getTime():Time
getRequest():byte[]
toString():String
-floorSubsystem
0..1
-floorSubsystem
0..1
-floorSubsystem
-floorSubsystem
0..1
-floors
0..*
-floorSubsystem
0..1
-listener
0..*
<<Java Class>>
Dispatcher
elevatorSystem.floorSubsystem.communication
data: byte[]
TEMP_LENGTH: int
ID: int
ELEVATOR_ARRIVED_HEADER: byte[]
SERVICE_REQUEST_HEADER: byte[]
TRANSIENT_FAULT_HEADER: byte[]
Dispatcher(FloorSubsystem,Object,int)
Dispatcher(FloorSubsystem,Object)
run():void
<<Java Class>>
SchedulerListener
elevatorSystem.floorSubsystem.communication
receiveSocket: DatagramSocket
PORT: int
SchedulerListener(FloorSubsystem,int)
run():void
-time
0..1
-sender
0..1
<<Java Class>>
Time
elevatorSystem.floorSubsystem.helpers
hour: int
min: int
sec: int
msec: int
HOUR_TO_MS: int
MIN_TO_MS: int
SEC_TO_MS: int
Time(String)
getHour():int
getMin():int
getSec():int
getMsec():int
before(Time):boolean
waitTime(Time):int
<<Java Class>>
ElevatorArrived
elevatorSystem.floorSubsystem.states
data: byte[]
floorNum: int
direction: String
ID: int
TEMP_LENGTH: int
ElevatorArrived(FloorSubsystem,byte[],int)
run():void
decodeData():void
<<Java Class>>
ServiceRequest
elevatorSystem.floorSubsystem.states
ZERO_PARSER: byte[]
TEMP_LENGTH: int
SELECT_ELEV_HEADER: byte[]
data: byte[]
floorNum: int
fault: int
direction: String
ServiceRequest(FloorSubsystem,byte[])
run():void
decodeData():void
-sender
0..1
<<Java Class>>
SendToScheduler
elevatorSystem.floorSubsystem.communication
sendPacket: DatagramPacket
sendSocket: DatagramSocket
PORT: int
schedulerAddress: InetAddress
SendToScheduler(InetAddress)
SendToScheduler()
send(byte[]):void


<<Java Class>>
FloorListener
elevatorSystem.scheduler.communication
receiveSocket: DatagramSocket
PORT: int
FloorListener(Scheduler)
run():void
<<Java Class>>
Scheduler
elevatorSystem.scheduler
elevatorIDs: int[]
NUM_ELEVATORS: int
output: Output
simDone: boolean
arrivalSensorsInterfaceCalculator: PerformanceCalculator
elevatorButtonsInterfaceCalculator: PerformanceCalculator
floorButtonsInterfaceCalculator: PerformanceCalculator
Scheduler(Output)
Scheduler(Output,InetAddress,InetAddress[])
getFloorSender(int):SendToFloor
getElevatorSenders():SendToElevator[]
getElevators():ElevatorStatus[]
getWorkingElevators():ArrayList<ElevatorStatus>
getElevatorIDs():int[]
getOutput():Output
addRequest(byte[]):void
redistributeRequests(ArrayList<RequestData>,boolean,int,StringData,int):void
getPendingRequests():Queue<PickUp>
enqueuePendingRequests(PickUp):void
elevatorsToString():String
containsPickUp(int,StringData):ArrayList<RequestData>
getArrivalSensorsInterfaceCalc():PerformanceCalculator
getElevatorButtonsInterfaceCalc():PerformanceCalculator
getFloorButtonsInterfaceCalc():PerformanceCalculator
setSimDone(boolean):void
getSimDone():boolean
run():void
main(String[]):void
<<Java Class>>
ElevatorListener
elevatorSystem.scheduler.communication
receiveSocket: DatagramSocket
PORT: int
ElevatorListener(Scheduler,int)
run():void
<<Java Class>>
Dispatcher
elevatorSystem.scheduler.communication
ADD_NEW_DEST_HEADER: byte[]
SELECT_ELEVATOR_HEADER: byte[]
ARRIVED_AT_FLOOR_HEADER: byte[]
TEMP_LENGTH: int
SYSTEM_FAULT_HEADER: byte[]
DOOR_FAULT_HEADER: byte[]
DOOR_FIXED_HEADER: byte[]
data: byte[]
Dispatcher(Scheduler,Object)
run():void
<<Java Class>>
FaultHandler
elevatorSystem.scheduler.states
elevatorWithFaultID: int
data: byte[]
transientFault: boolean
TEMP_LENGTH: int
FaultHandler(Scheduler,byte[],boolean)
FaultHandler(Scheduler,byte[])
fix():void
run():void
decodeData():void
<<Java Class>>
SendToFloor
elevatorSystem.scheduler.communication
sendPacket: DatagramPacket
sendSocket: DatagramSocket
PORT: int
floorAddress: InetAddress
SendToFloor(int)
SendToFloor(int,InetAddress)
send(byte[]):void
<<Java Class>>
SendToElevator
elevatorSystem.scheduler.communication
sendPacket: DatagramPacket
sendSocket: DatagramSocket
PORT: int
elevatorAddress: InetAddress
SendToElevator(int)
SendToElevator(int,InetAddress)
send(byte[]):void
<<Java Class>>
DropOff
elevatorSystem.scheduler.helpers
DropOff(int,StringData)
<<Java Class>>
PickUp
elevatorSystem.scheduler.helpers
PickUp(int,StringData,int,int)
<<Java Class>>
ArrivedAtFloor
elevatorSystem.scheduler.states
direction: StringData
continueStatus: String
destinations: ArrayList<Integer>
faults: ArrayList<Integer>
ZERO_PARSER: byte[]
TEMP_LENGTH: int
data: byte[]
elevatorID: int
STOP_ELEVATOR_HEADER: byte[]
DEST_SELECTED_HEADER: byte[]
ELEVATOR_ARRIVED_HEADER: byte[]
startTime: Long
ArrivedAtFloor(Scheduler,byte[])
run():void
stopOrNot():boolean
checkIfBeatOthers():void
simulateCarButton(int,int):void
notifyFloor():void
stopElevator():void
signalDone():void
performCalculations():void
decodeData():void
<<Java Class>>
RequestData
elevatorSystem.scheduler.helpers
floorToService: int
directionTraveling: StringData
destination: int
fault: int
type: StringData
RequestData(int,StringData,int,int)
RequestData(int,StringData)
getFloorToService():int
getDirectionTraveling():StringData
getDestinationFloor():int
getFault():int
getType():StringData
equals(Object):boolean
toString():String
<<Java Class>>
ElevatorStatus
elevatorSystem.scheduler.helpers
id: int
elevatorWorking: boolean
elevatorStarted: boolean
elevatorFloor: int
elevatorDirection: StringData
STARTING_FLOOR: int[]
ElevatorStatus(int)
getElevatorFloor():int
setElevatorFloor(int):void
getElevatorDirection():StringData
setElevatorDirection(StringData):void
getElevatorStarted():boolean
setElevatorStarted(boolean):void
getFloorsToVisit():ArrayList<RequestData>
getID():int
incrementFloor():void
decrementFloor():void
setWorking(boolean):void
isWorking():boolean
containsPickUp(int,StringData):ArrayList<RequestData>
requestsToString():String
<<Java Class>>
SelectElevator
elevatorSystem.scheduler.states
startTime: Long
START_ELEVATOR_HEADER: byte[]
ZERO_PARSER: byte[]
SelectElevator(Scheduler)
run():void
chooseElevator():void
checkIdle(PickUp):boolean
checkDistance(PickUp):boolean
addRequest(ArrayList<RequestData>,RequestData):void
findOptions(PickUp):ArrayList<ElevatorStatus>
startElevators():void
<<Java Class>>
AddNewDestination
elevatorSystem.scheduler.states
START_ELEVATOR_HEADER: byte[]
ZERO_PARSER: byte[]
newFloor: int
fault: int
elevatorID: int
TEMP_LENGTH: int
data: byte[]
startTime: Long
AddNewDestination(Scheduler,byte[])
run():void
addRequest(ArrayList<RequestData>,DropOff):void
startElevator():void
decodeData():void
-floorListener
0..1
-schedulerSubsystem
0..1
-schedulerSubsystem
0..1
-scheduler
0..1
-schedulerSubsystem
0..1
-elevatorSender
0..*
-floorSender
0..*
-floorSender
0..1
-pendingRequests
0..*
-scheduler
0..1
-scheduler
0..1
-requests
0..*
-elevators
0..*
-elevatorWithFault
0..1
-floorsToVisit
0..*
-elevators
-currentElevator
0..*
0..1
-workingElevators
0..*
-elevators
0..*
-senders
0..*
-senders
0..*
-elevatorSenders
0..*


<<Java Class>>
SchedulerListener
elevatorSystem.elevatorSubsystem1.communication
receiveSocket: DatagramSocket
PORT: int
SchedulerListener(ElevatorSubsystem,int)
run():void
<<Java Class>>
ElevatorSubsystem
elevatorSystem.elevatorSubsystem1
output: Output
ListenPort: int
SendPort: int
ID: int
ElevatorSubsystem(Output,InetAddress)
ElevatorSubsystem(Output)
getSender():SendToScheduler
getElevator():Elevator
getOutput():Output
main(String[]):void
<<Java Class>>
Elevator
elevatorSystem.elevatorSubsystem1.helpers
NUM_BUTTONS: int
buttonLamps: Status[]
motor: Status
working: boolean
doors: Status
Elevator(SendToScheduler,int)
getButtonLamps():Status[]
setButtonLamp(int,Status):void
getMotor():Status
setMotor(Status):void
getDoors():Status
setDoors(Status):void
getFloorSensor():FloorSensor
setWorking(boolean):void
isWorking():boolean
<<Java Class>>
FloorSensor
elevatorSystem.elevatorSubsystem1.helpers
sensing: boolean
id: int
ARRIVED_AT_FLOOR_HEADER: byte[]
FloorSensor(Elevator,SendToScheduler,int)
run():void
stopSensing():void
startSensing():void
<<Java Class>>
Dispatcher
elevatorSystem.elevatorSubsystem1.communication
data: byte[]
TEMP_LENGTH: int
START_ELEVATOR_HEADER: byte[]
STOP_ELEVATOR_HEADER: byte[]
DEST_SELECTED_HEADER: byte[]
Dispatcher(ElevatorSubsystem,Object)
run():void
<<Java Class>>
StopElevator
elevatorSystem.elevatorSubsystem1.states
TEMP_LENGTH: int
data: byte[]
elevatorID: int
continueStatus: String
floorNum: int
ZERO_PARSER: byte[]
START_ELEVATOR_HEADER: byte[]
StopElevator(ElevatorSubsystem,byte[])
run():void
decodeData():void
<<Java Class>>
StartElevator
elevatorSystem.elevatorSubsystem1.states
TEMP_LENGTH: int
data: byte[]
elevatorID: int
fault: int
DOOR_FAULT_HEADER: byte[]
DOOR_FIXED_HEADER: byte[]
SYSTEM_FAULT_HEADER: byte[]
StartElevator(ElevatorSubsystem,byte[])
run():void
decodeData():void
<<Java Class>>
SendToScheduler
elevatorSystem.elevatorSubsystem1.communication
sendPacket: DatagramPacket
sendSocket: DatagramSocket
PORT: int
schedulerAddress: InetAddress
SendToScheduler(int,InetAddress)
SendToScheduler(int)
send(byte[]):void
<<Java Class>>
DestinationSelected
elevatorSystem.elevatorSubsystem1.states
TEMP_LENGTH: int
data: byte[]
elevatorID: int
destinationButton: int
fault: int
ZERO_PARSER: byte[]
ADD_NEW_DEST_HEADER: byte[]
DestinationSelected(ElevatorSubsystem,byte[])
run():void
decodeData():void
-listener
-elevatorSubsystem
-elevator
-sensor
-sender
0..1