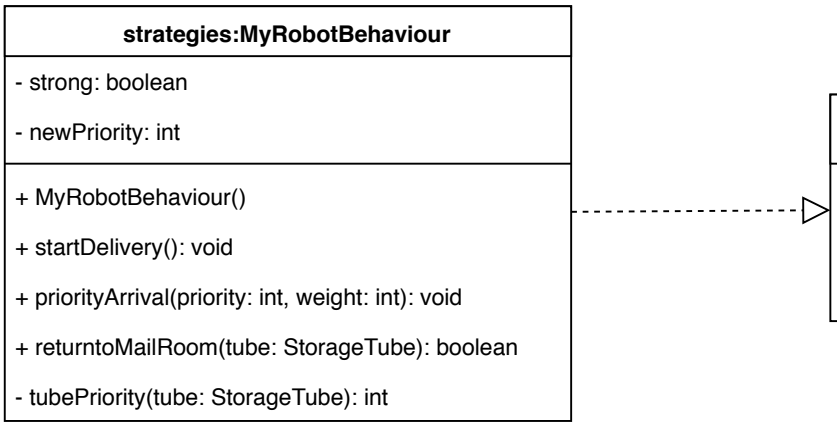
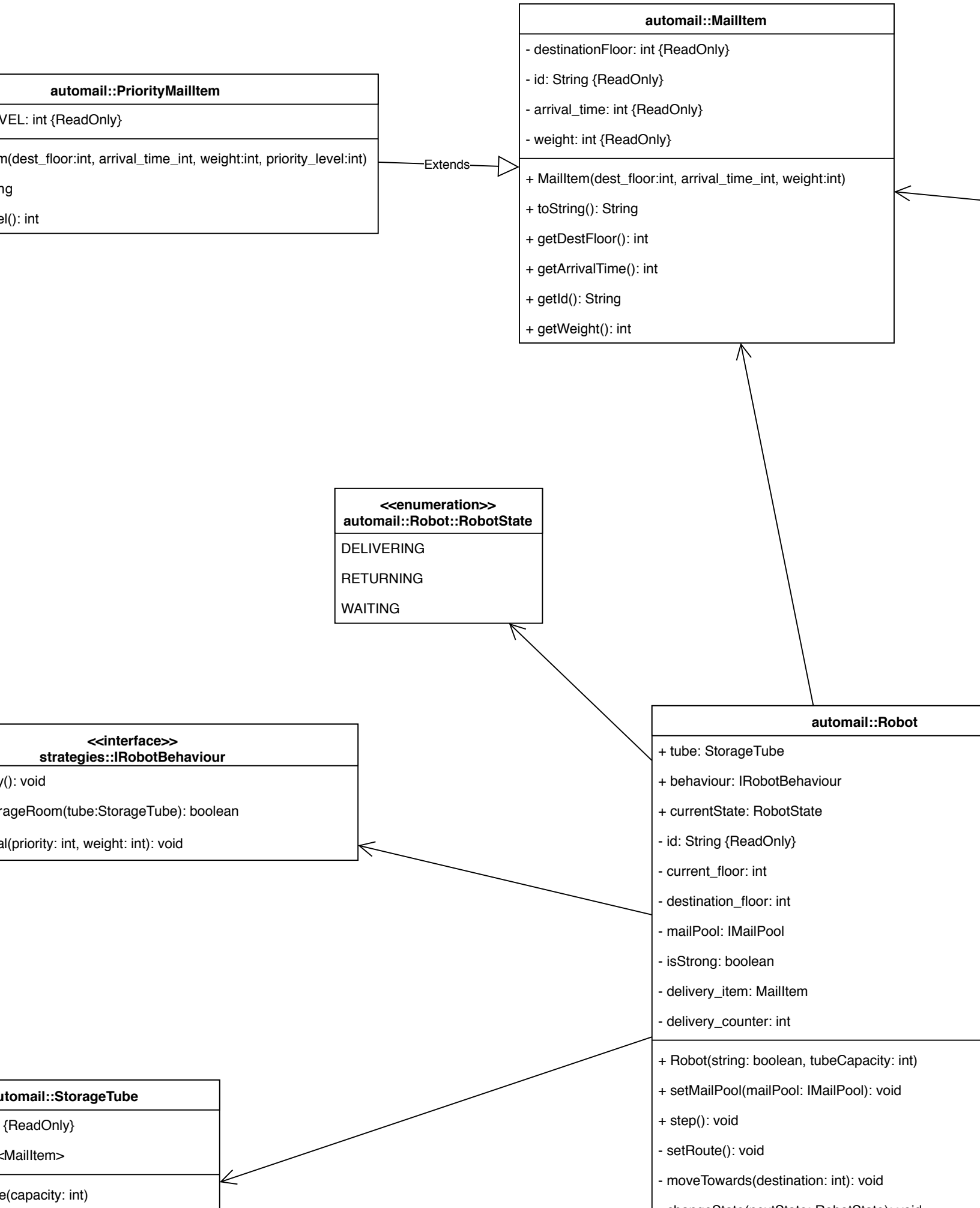
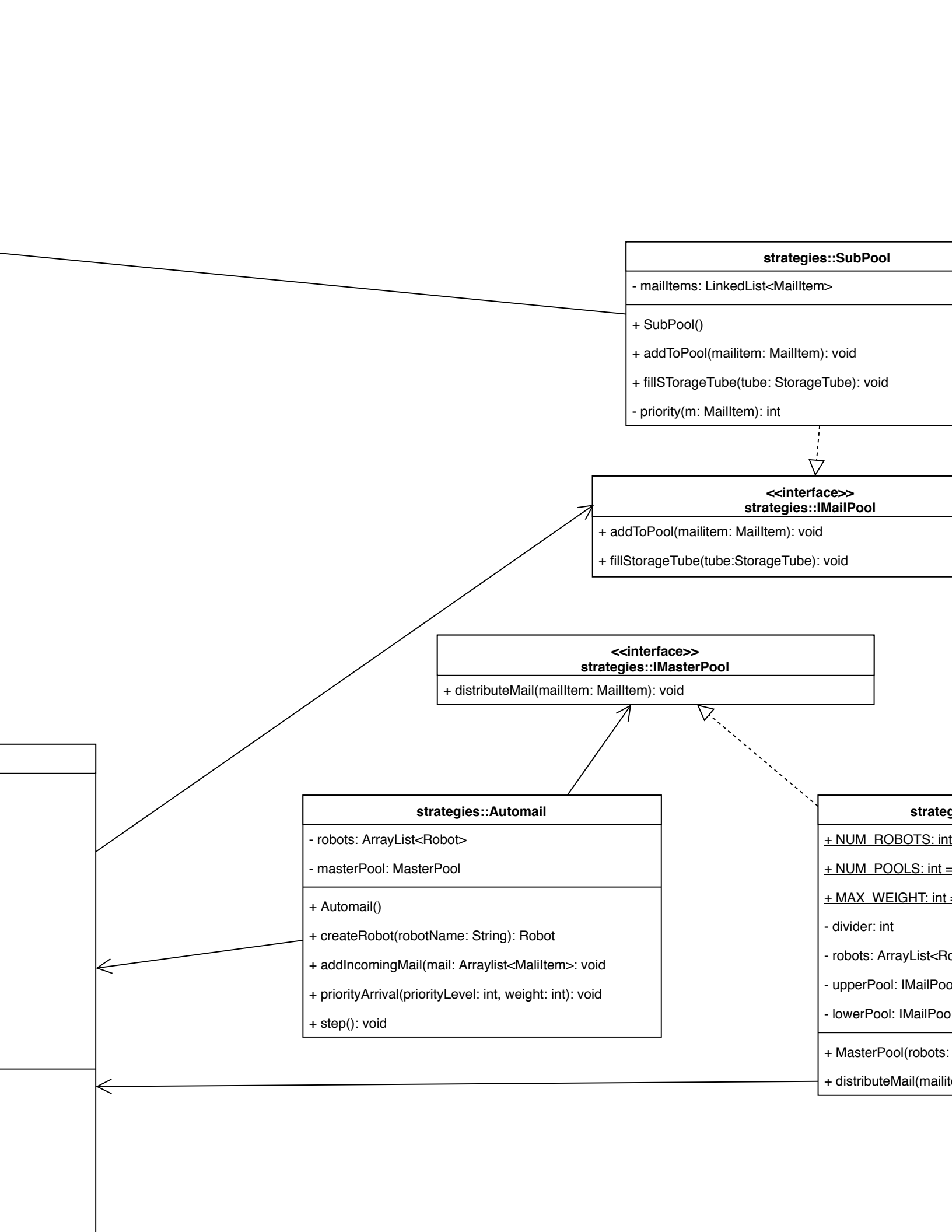
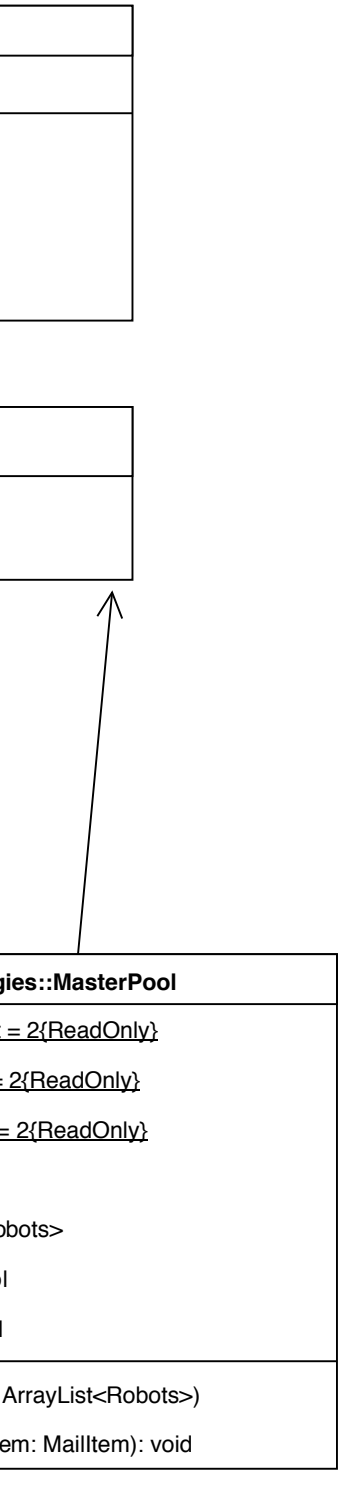
Project Group 13:
-Liam Whittle
-Chris Chapman
-Raymond Sun




automail::MailItem
- destinationFloor: int {ReadOnly}
- id: String {ReadOnly}
- arrival_time: int {ReadOnly}
- weight: int {ReadOnly}
+ MailItem(dest_floor:int, arrival_time_int, weight:int)
+ toString(): String
+ getDestFloor(): int
+ getArrivalTime(): int
+ getId(): String
+ getWeight(): int
automail::PriorityMailItem
VEL: int {ReadOnly}
m(dest_floor:int, arrival_time_int, weight:int, priority_level:int)
ng
el(): int
Extends
<<enumeration>>
automail::Robot::RobotState
DELIVERING
RETURNING
WAITING
automail::Robot
+ tube: StorageTube
+ behaviour: IRobotBehaviour
+ currentState: RobotState
- id: String {ReadOnly}
- current_floor: int
- destination_floor: int
- mailPool: IMailPool
- isStrong: boolean
- delivery_item: MailItem
- delivery_counter: int
+ Robot(string: boolean, tubeCapacity: int)
+ setMailPool(mailPool: IMailPool): void
+ step(): void
- setRoute(): void
- moveTowards(destination: int): void
<<interface>>
strategies::IRobotBehaviour
y(): void
rageRoom(tube:StorageTube): boolean
al(priority: int, weight: int): void
utomail::StorageTube
{ReadOnly}
<MailItem>
e(capacity: int)


strategies::SubPool
- mailItems: LinkedList<MailItem>
+ SubPool()
+ addToPool(mailitem: MailItem): void
+ fillSTorageTube(tube: StorageTube): void
- priority(m: MailItem): int
<<interface>>
strategies::IMailPool
+ addToPool(mailitem: MailItem): void
+ fillStorageTube(tube:StorageTube): void
<<interface>>
strategies::IMasterPool
+ distributeMail(mailItem: MailItem): void
strategies::Automail
- robots: ArrayList<Robot>
- masterPool: MasterPool
+ Automail()
+ createRobot(robotName: String): Robot
+ addIncomingMail(mail: Arraylist<MaliItem>: void
+ priorityArrival(priorityLevel: int, weight: int): void
+ step(): void
+ NUM_ROBOTS: int
+ NUM_POOLS: int =
+ MAX_WEIGHT: int =
- divider: int
- robots: ArrayList<Ro
- upperPool: IMailPoo
- lowerPool: IMailPoo
+ MasterPool(robots:
+ distributeMail(mailit



automail::Simulation

- MAIL_DELIVERED; ArrayList<MailItem>
- total_score: double

+ main(args: String[]): void
+ deliver(deliveryitem: MalilItem): void
- calculateDeliveryScore(deliveryitemL MailItem); double
+ printResults(): void

automail::MailGenerator

+ MAIL_AMOUNT: int {ReadOnly}
- random Random {ReadOnly}
- mailCreated: int
- allMail: HashMap<Integer, ArrayList<MailItem>>

+ MailGenerator()
+ generateAllMail(): void
+ getMail(): ArrayList<MailItem>
- generateMail(): MailItem
- generateDestinationFloor(): int
- generatePriorityLevel(): int
- generateWeight(): int
- generateArrivalTime(): int
- getRandom(array: Object[]); Object

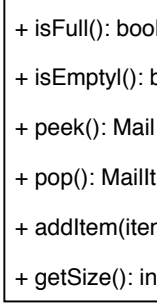
automail::Clock

- Time: int=0
+ LAST_DELIVERY_TIME: int {ReadOnly}

+ Time(): int
+ Tick(): void

automail::Building

+ FLOORS: int {ReadOnly}


+ isFull(): bool
+ isEmptyl(): b
+ peek(): Mail
+ pop(): MailIt
+ addItem(iter
+ getSize(): in

lean

oolean

Item

em

m: MailItem): void

t

- changeState(nextState: RobotState): void

Extends

Extends

automail::RobotBig

+ IS_STRONG: boolean = true {ReadOnly}

+ TAG: String = "big" {ReadOnly}

+ MAX_TUBE_CAPACITY: int = 6 {ReadOnly}

+ RobotBig()

automail::RobotStrong

+ IS_STRONG: boolean = true {ReadOnly}

+ TAG: String = "strong" {ReadOnly}

+ MAX_TUBE_CAPACITY: int = 4{ReadOnly}

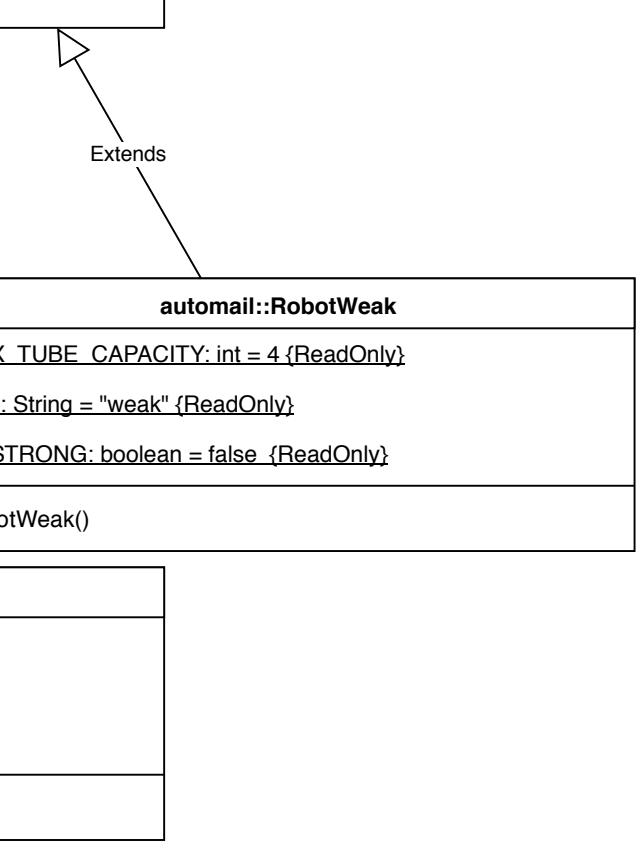
+ RobotStrong()

+ MAX

+ TAG

+ IS_S

+ Robo


Extends
automail::RobotWeak
X_TUBE_CAPACITY: int = 4 {ReadOnly}
: String = "weak" {ReadOnly}
STRONG: boolean = false {ReadOnly}
otWeak()

+ LOWEST_FLOOR: int=1 {ReadOnly}

+ MAILROOM_LOCATION: int=1 {ReadOnly}

automail::PropertiesLoader

- seed:Integer

- timePenalty: Double

- mailToCreate:Integer

- lastDeliveryTime:Integer

- floorsInBuilding: Integer

- robotTypes: ArrayList<String>

+ loadProperties(): void

+ getSeed(): integer

+ getTimePenalty():Double

+ getMailToCreate():Integer

+ getLastDeliveryTime(): Integer

+ getFloorsInBuilding(): Integer

+ getRobotTypes(): ArrayLIst<String>