<<interface>> VehicleObserver

vehicleHasBeenRemoved(vehicle: Vehicle): void

vehicleHasMoved(newRoad: Road): void

SimulationController

- vehiclesInQueue: LinkedList<Vehicle>
- vehiclesOnGrid: ArrayList<Vehicle>
- dto: SimulationDTO
- simulationView: SimulationView
- closed: boolean
- simulationModel: Simulation

+ SimulationController(dto: SimulationDTO)
- initViewActions(): void
+ close(): void
+ updateCell(road: Road): void
+ getGridTable(): GridTable
+ loadVehicles(): LinkedList<Vehicles>
+ getVehiclesOnGrid(): ArrayList<Vehicle>
+ getVehiclesInQueue(): LinkedList<Vehicle>
+ stopAndWait(): void
+ vehicleHasBeenRemoved(vehicle: Vehicle): void
+ vehicleHasMoved(newRoad: Road): void

SimulationDTO

- carQuantity: int
- insertionInterval: int
- meshFileName: String
- maxCarInMeshQuantitySameTime: int
- methodString: String

+ SimulationDTO(carQuantity: int, insertionInterval: int, meshFileName: String, maxCarInMeshQuantitySameTime: int, methodString: String)
+ getMethod(): String
+ getCarQuantity(): int
+ getInsertionInterval(): int
+ getMeshFileName(): String
+ getMaxCarInMeshQuantitySameTime(): int

RoadController

- road: Road

+ RoadController(road: Road, grid: GridTable)
+ isUpperEntry(): boolean
+ isUpperExit(): boolean
+ isLowerEntry(gridTable: GridTable): boolean
+ isLowerExit(gridTable: GridTable): boolean
+ isLeftEntry(): boolean
+ isLeftExit(): boolean
+ isRightEntry(gridTable: GridTable): boolean
+ isRightExit(gridTable: GridTable): boolean
+ defineEntryOrExit(gridTable: gridTable): void

GridController

- gridTable: GridTable
- roadController: RoadController
- method: String

+ GridController(simulationDTO: SimulationDTO)
+ createMatrix(): void
+ getGridTable(): GridTable

Road

- ICONS_PATH: String
- semaphore: Semaphore
- iconDirectory: String
- entry: boolean
- exit: boolean
- type: int
- vehicle: Vehicle
- row: int
- column: int
- method: String
- lock: Lock

+ Road(row: int, column: int, type: int, method: String)
+ tryAcquire(): boolean
+ release(): void
+ defineCurrentIcon(): void
+ setIconDirectoryByType(): void
+ setVehicleIconDirectory(): void
+ getRow(): int
+ getColumn(): int
+ getIconDirectory(): String
+ setIconDirectory(iconDirectory: String): void
+ getType(): int
+ isEntry(): boolean
+ setEntry(isEntryCell: boolean): void
+ isExit(): boolean
+ setExit(isExitCell: boolean): void
+ getVehicle(): Vehicle
+ isIntersection(): boolean
+ isEmpty(): boolean
+ isRoad(): boolean

Vehicle

- simulation: Simulation
- route: ArrayList<Road>
- roadConnection: Road[][]
- random: Random
- speed: int
- currentRoad: Road
- closed: boolean
- type: int
- observers: ArrayList<VehicleObserver>

+ Vehicle(simulation: Simulation)
+ notifyVehicleRemoved(): void
+ notifyVehicleMoved(newRoad: Road): void
+ close(): void
- resolveIntersection(): void
- loadNecessaryIntersection(): ArrayList<Road>
- attemptToReserveIntersections(intersectionsToReserve: ArrayList<Road>): ArrayList<Road>
- releaseRoadList(roads: ArrayList<Road>): void
- move(nextRoad: Road, reserve: boolean): void
+ setRoute(entry: Road): void
+ chooseRoadByDirection(direction: int, currentRow: int, currentColumn: int): Road
- chooseIntersectionByDirection(direction: int, currentRow: int, curretnColumn: int, intersectionFound: int): Road
+ getType(): int
+ getCurrentRoad(): Road
+ setCurrentRoad(currentRoad: Road): void
+ delay(): void

Simulation

- vehiclesInQueue: LinkedList<Vehicle>
- vehiclesOnGrid: ArrayList<Vehicle>
- rowCount(): int
- columnCount(): int
- roadConnection(): Road[][]
- intersectionInterval: int
- closed: boolean
- maxVehiclesSameTime: int

+ Simulation(rowCount: int, columnCount: int, roadConnection: Road[][], intersectionInterval: int, maxVehiclesSameTime: int)
+ getRoadConnection(): Road[][]
+ addVehicleToGrid(vehicle: Vehicle): void
- executeQueue(): void
- sleepNextVehicle(): void
+ setVehiclesInQueue(vehiclesInQueue: LinkedList<Vehicle>): void
+ setVehiclesOnGrid(vehiclesOnGrid: ArrayList<Vehicle>): void

---

1 ← possui 1

possui → 1

1 possui → 1

1 possui → 1

possui → 1

* ← possui 1