Utils

<<Interface>> ListBehavior<Type>

+ add(element: Type): boolean
+ add(elements: Type[]): boolean
+ getSize(): int
+ isEmpty(): boolean
+ getElement(index: int): Type
+ removeElement(int index): boolean
+ clear(): void
+ contains(element: Type): boolean
+ contains(elements: Type[]): boolean
+ equals(obj: Object): boolean
+ hashCode(): int

ListHelper

+ listToString(list: ListBehavior<?>): String

CarCreator

+ CAR_BRANDS: String[]
- LETTERS_FOR_MODEL: String
+ MIN_COST_FOR_PASS_CAR: int
+ MAX_COST_FOR_PASS_CAR: int
+ MIN_COST_FOR_MINIBUS: int
+ MAX_COST_FOR_MINIBUS: int

+ createCar(cost: double): Car
+ createCar(minCost: double, maxCost: double): Car
+ createPassengerCar(): PassengerCar
+ createPassengerCar(cost: double): PassengerCar
+ createPassengerCar(minCost: double, maxCost: double): PassengerCar
+ createMinibus(): Minibus
+ createMinibus(cost: double): Minibus
+ createMinibus(minCost: double, maxCost: double): Minibus

LimitedList<Type>

DEFAULT_CAPACITY: int
size: int
value: Object[]

- rangeCheck(index: int): void
+ getCapacity(): int
+ removeElement(element: Type): boolean
+ indexOf(element: Type): int

UnlimitedList<Type>

- grow(): void

buisnesLogicLayer

entities

taxiStation

TaxiStation

- RIDE_TIME_IN_MILLIS: int
- budget: double
- name: String
- garage: Car [0..*]
- onCall: Car [0..*]
- orders: Thread [0..*]

+ getName(): String
+ getBudget(): double
+ setName(name: String): void
+ setBudget(budget: double): void
+ addCar(car: Car): void
+ removeCar(car: Car): void
+ sendCarOnCall(car: Car): void
+ trimCalls(): void
+ contains(car: Car): boolean
+ getStatinCapacity(): int
+ getGarage(): Car[0..*]
+ getOnCall(): Car[0..*]
+ getAllCars(): Car[0..*]
+ waitCars(): void
+ equals(o: Object): boolean
+ hashCode(): int
+ toString(): String
+ finalize(): void

cars

Car

+ MIN_SITS: int
# sits: int
# cost: double
# carBrand: String
# model: String

+ getSits(): int
+ getCost(): double
+ getCarBrand(): String
+ getModel(): String
+ setSits(sits: int): void
+ setCost(cost: double): void
+ setCarBrand(carBrand: String): void
+ setModel(model: String): void
+ equals(o: Object): boolean
+ hashCode(): int
+ toString(): String

PassengerCar

+ MAX_SITS: int

Minibus

+ MIN_SITS: int
+ MAX_SITS: int

logic

staff

Administrator

+ buyCar(station: TaxiStation, car: Car): boolean
+ sellCar(station: TaxiStation, car: Car): boolean
+ calculateStationCost(station: TaxiStation): double
+ findMostExpensiveCar(station: TaxiStation): Car
+ findCheapestCar(station: TaxiStation): Car

Operator

COST_FOR_RIDE_PRE_PERSON: double

+ takeOrder(station: TaxiStation, passengers: int): double
- findCar(station: TaxiStation, passengers: int): Car
+ getCarsOnCall(station: TaxiStation): String
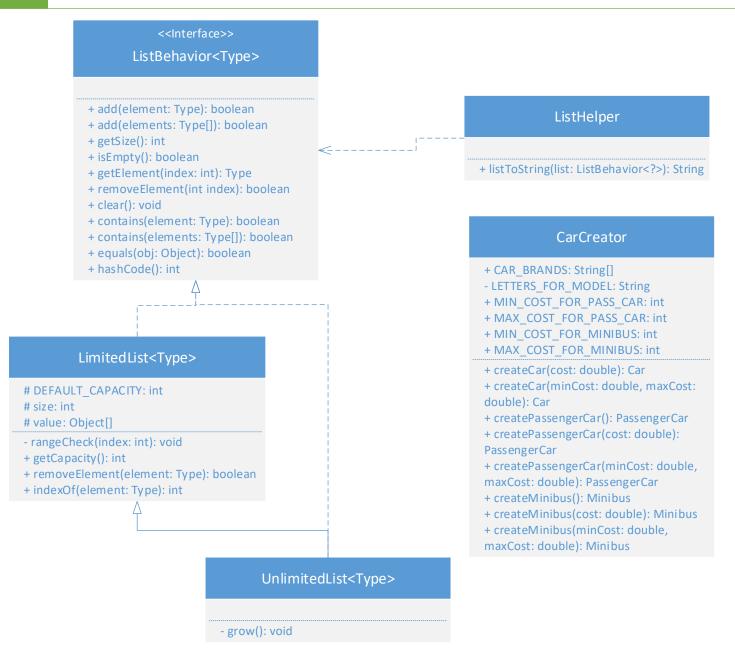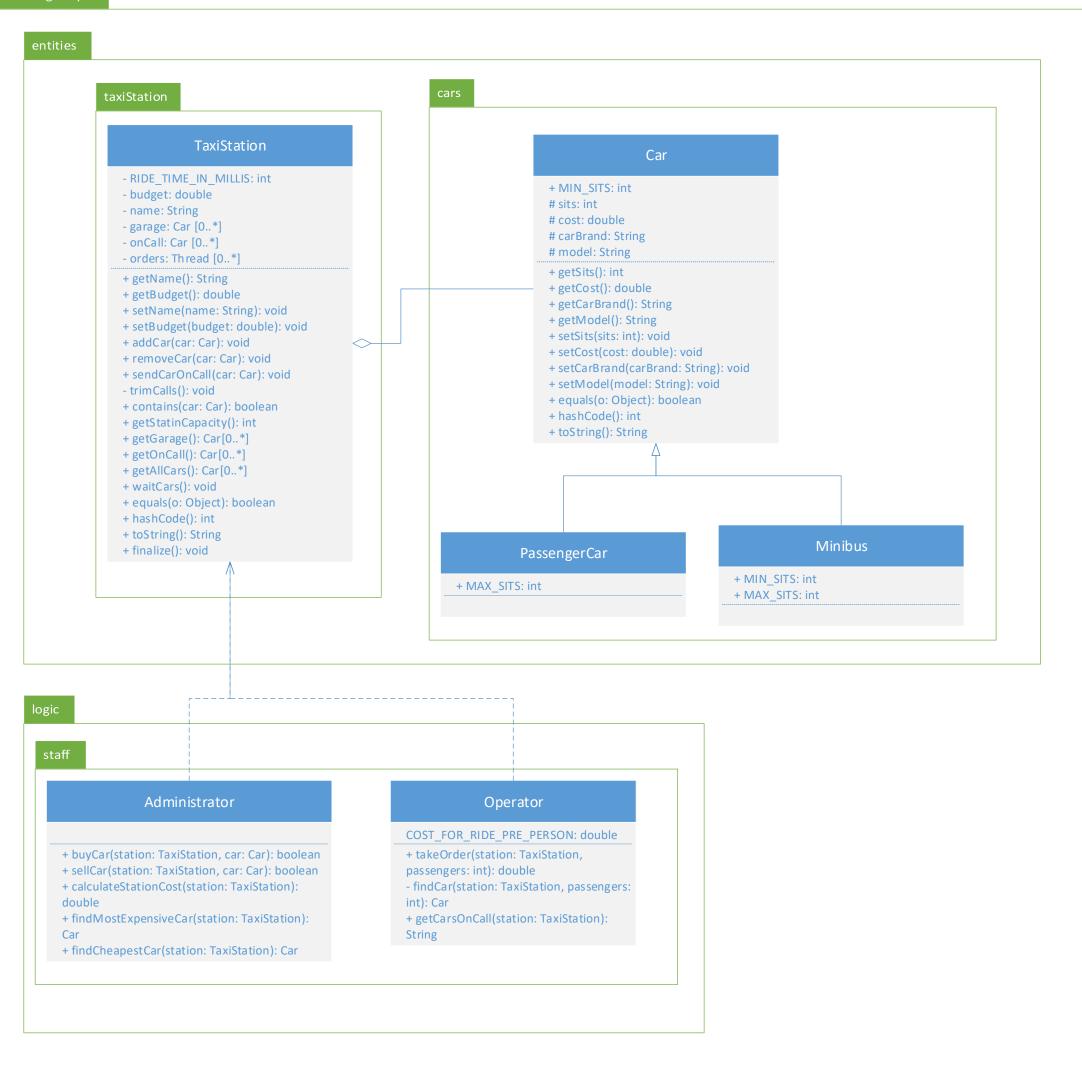
presentationLayer

ConsolePrinter

+ print(info: byte): void
+ print(info: short): void
+ print(info: char): void
+ print(info: int): void
+ print(info: long): void
+ print(info: float): void
+ print(info: double): void
+ print(info: boolean): void
+ print(info: Object): void

controller

Task07

+ main(args: String[]): void
+ presentation(): void