One Night Car

Carsharing Verwaltungssystem

Team

Benito Grauel

Pascal Giese

Ahmad Abo Louha

Alejandro Restrepo Klinge

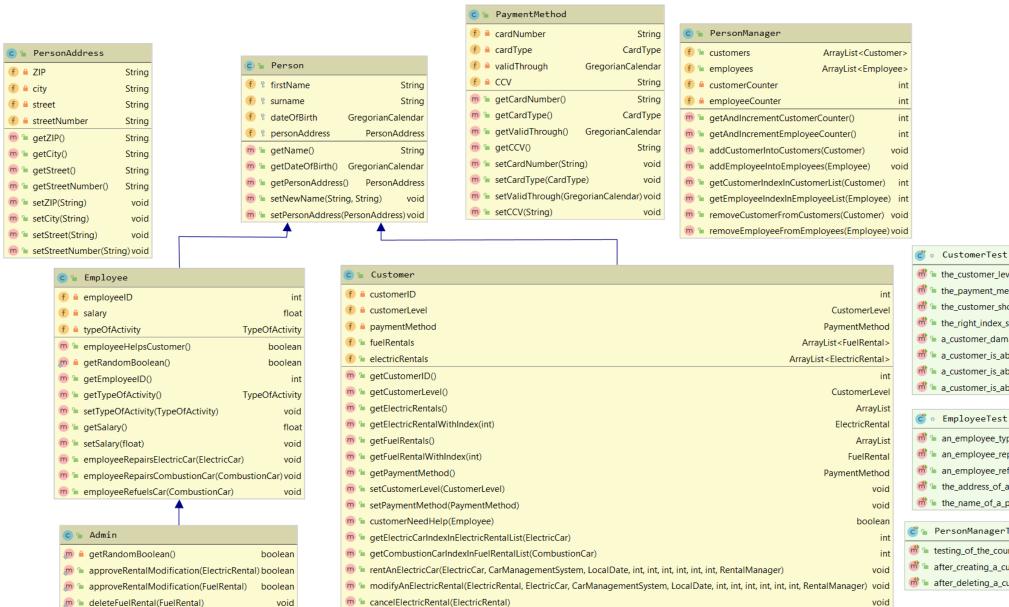
GitHub Repository: https://github.com/fh-erfurt/One-Night-Car

GitHub Team: https://github.com/orgs/fh-erfurt/teams/team-car-sharing

Packages

- ♦ Person: Verwaltung von Kunden (Registrieren, Login, Type, ...) und Mitarbeiter
- ♦ Car: Verwaltung von Autos (Auto Informationen, Benzin, ...)
- ParkingArea: Verwaltung von dazugehörigen Autos (Verfügbarkeit)
- Rental: Reservierung von Autos verwalten (Abrechnung, Datum)

Package Person



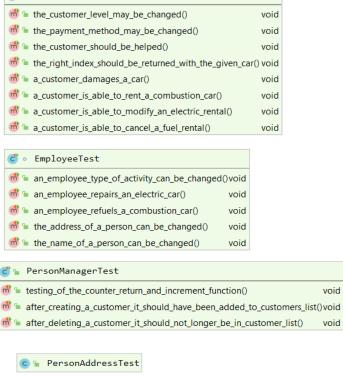
m = cancelFuelRental(FuelRental)

m = customerDamagesAnElectricCar(ElectricCar)

m = customerDamagesAFuelCar(CombustionCar)

m 乍 rentAFuelCar(RentalManager, CombustionCar, CarManagementSystem, LocalDate, int, int, int, int, int, int)

m 🤚 modifyARegularRental(FuelRental, RentalManager, CombustionCar, CarManagementSystem, LocalDate, int, int, int, int, int, int) void



void

void

void

void

m 🖢 deleteElectricRental(ElectricRental)

m = deleteEmployee(Employee, PersonManager)

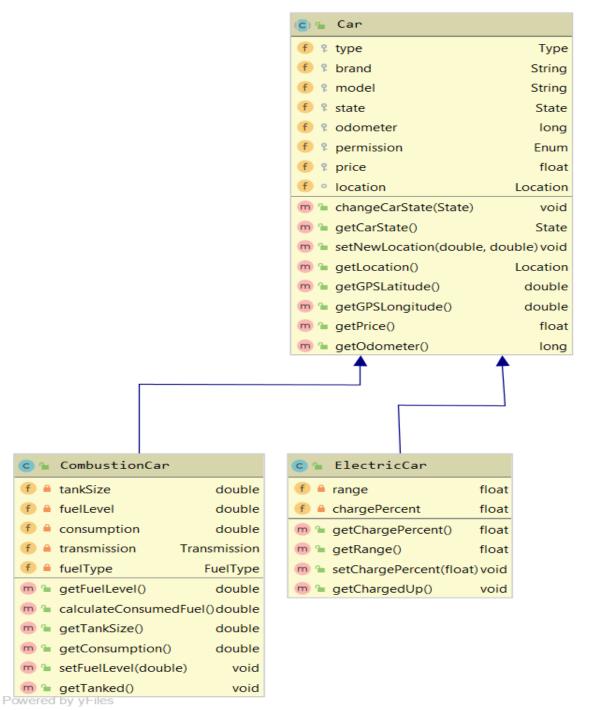
m = deleteCustomer(Customer, PersonManager)

resolveProblem()

void

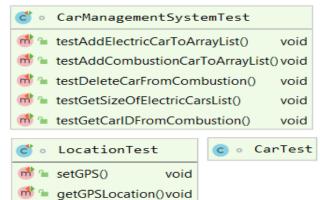
boolean

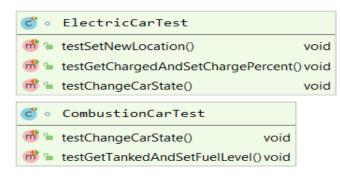
Package Car



© ← CarManagementSystem		
f = combustionCarsList ArrayList <combustion< td=""><td>Car></td></combustion<>	Car>	
f electricCarsList ArrayList <electric< td=""><td>Car></td></electric<>	Car>	
m = addCarIntoElectrics(ElectricCar)	void	
m = addCarIntoCombustion(CombustionCar)	void	
m = deleteCarFromElectric(ElectricCar)	void	
m 🐿 deleteCarFromCombustion(CombustionCar)void		
m = getSizeOfElectricCarsList()	int	
m = getSizeOfCombustionCarsList()	int	
$\begin{tabular}{ll} \hline m & $=$ getCarIDFromCombustion(CombustionCar) \\ \hline \end{tabular}$	int	
m = getCarlDFromElectric(ElectricCar)	int	

C Location	
f ≜ GPSLatitude	double
f ≜ GPSLongitude	double
m = setGPSLatitude(doub	le) void
m = setGPSLongitude(dou	ıble) void
m = getGPSLatitude()	double
m = getGPSLongitude()	double





Package ParkingArea

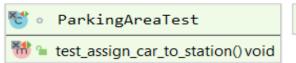
C ← ParkingArea		
f ≜ parkID int		
f ≜ maxCapacity int		
f o parkingAreaAddress ParkingAreaAddress		
f ← carsInStation ArrayList <combustioncar></combustioncar>		
f ← availableCars ArrayList < CombustionCar >		
f ← notAvailableCars ArrayList < CombustionCar >		
m = assignCarToStation(CombustionCar) void		
m = removeCarFromStation(CombustionCar) void		
m 🔓 carlsNoLongerBeingUsed(CombustionCar) void		
m = numberOfCarsAssignedToStation() int		
m = getParkingAreaAddress() ParkingAreaAddress		
m ← getMaxCapacity() int		
m = getCarsInStation() ArrayList <combustioncar></combustioncar>		

C i	ElectricParkingAre	ea
f A	${\it maxElectric Car Capacity}$	int
f a	electricCarsInStation	ArrayList <electriccar></electriccar>
f 🚡	availableElectricCars	ArrayList < ElectricCar >
f =	not Available Electric Cars	ArrayList < ElectricCar >
m 1	getMaxElectricCarCapac	ity() int
m 1	as sign Electric Car To Station	on(ElectricCar) void
m 🚡	removeElectricCarFromS	station(ElectricCar)void

C TarkingAreaManager		
	Area>	
	Area>	
f = parkingAreaCounter	int	
m = getAndIncrementCounter()	int	
m = addElectricParkingAreaIntoElectricParkingAreas(ElectricParkingArea)	void	
m = addParkingAreaIntoParkingAreas(ParkingArea)	void	
m = removeElectricParkingAreaIntoElectricParkingAreas(ElectricParkingArea)void		
m = removeParkingAreaIntoParkingAreas(ParkingArea)	void	
m = getSizeOfElectricParkingAreas()	int	
m = getSizeOfParkingAreas()	int	
m = getParkIDFromElectricParkingAreas(ElectricParkingArea)	int	
m = getParkIDFromParkingAreas(ParkingArea)	int	

C ParkingAreaAdd	dress
f ≜ ZIP	String
f ≜ city	String
f ≜ street	String
f ≜ streetNumber	String
m = getZIP()	String
m = getCity()	String
m = getStreet()	String
m = getStreetNumber()	String

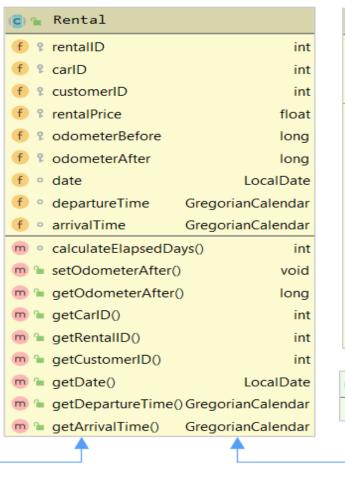
c •	ParkingAreaManagerTest		
₼ •	d a testing_of_the_counter_return_and_increment_function()void		
*	testing_add_parking_area_into_parking_areas()	void	
₼ •	testing_removing_parking_area_from_parking_areas()	void	
₼	d a testing_get_size_from_parking_area()		
₼ •	testing_get_parkID_from_electric_parking_areas()	void	



c • ParkingAreaAddressTest

c • ElectricParkingAreaTest

Package Rental

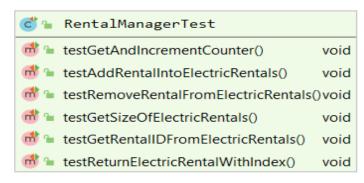


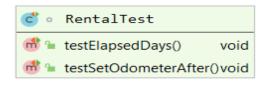
C 1=	RentalManager		
f a	ElectricRentals	ArrayList < ElectricRe	ental>
f A	FuelRentals	ArrayList <fuelre< td=""><td>ental></td></fuelre<>	ental>
f A	RentalCounter		int
m 強	getAndIncrementCounter()		int
m 🖢	add Rental Into Electric Rentals	(ElectricRental)	void
m 🖢	add Rental Into Fuel Rentals (Full Note of the Control of the Co	elRental)	void
m 1	remove Rental From Electrics Rental From Electric Rental	entals(ElectricRental	l) void
m 🖢	remove Rental From Fuel From Fuel Rental From Fuel Rental From Fuel Rental From Fuel From From From From From From From From	als(FuelRental)	void
m 1	getSizeOfElectricRentals()		int
m 🖢	getSizeOfFuelRentals()		int
m 🖢	get Rental ID From Electric	tals(ElectricRental)	int
m 🖢	get Rental ID From Fuel Rentals	(FuelRental)	int
m 🖢	return Electric Rental With Index	x(ElectricRental)	int
m 1=	returnFuelRentalWithIndex(F	uelRental)	int

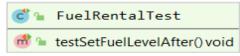


f a chargePercentBefore float
f a chargePercentAfter float
f a electricCar ElectricCar
m a setChargePercentAfter(ElectricCar) void
m a getChargePercentAfter() float
m a calculateRentalPriceForElectric(ElectricCar) float
m a getElectricCar() ElectricCar

f in fuelLevelBefore double
f in fuelLevelAfter double
f in combustionCar CombustionCar
m in setFuelLevelAfter(CombustionCar) void
m in getFuelLevelAfter() double
m in calculateRentalPriceForCombustion(CombustionCar) float
m in getCombustionCar() CombustionCar







Lessons Learned

- Vererbung ist ein sehr n\u00fctzliches Werkzeug, trotzdem sollte man nicht es \u00fcbernutzen (Problem mit Autos und ArrayListen)
- ♦ Planung ist immer sehr wichtig, aber wichtiger ist es, sich an dem Plan zu halten
- Aufgaben nicht unterschätzen
- ♦ Testen ist IMMER sehr wichtig, erst dann merkt man, dass man immer testen muss

Danke für die Aufmerksamkeit