

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

PersonAddress		
f	ZIP	String
f	city	String
f	street	String
f	streetNumber	String
m	getZIP()	String
m	getCity()	String
m	getStreet()	String
m	getStreetNumber()	String
m	setZIP(String)	void
m	setCity(String)	void
m	setStreet(String)	void
m	setStreetNumber(String)	void

Person		
f	firstName	String
f	surname	String
f	dateOfBirth	GregorianCalendar
f	personAddress	PersonAddress
m	getName()	String
m	getDateOfBirth()	GregorianCalendar
m	getPersonAddress()	PersonAddress
m	setNewName(String, String)	void
m	setPersonAddress(PersonAddress)	void

PaymentMethod		
f	cardNumber	String
f	cardType	CardType
f	validThrough	GregorianCalendar
f	CCV	String
m	getCardNumber()	String
m	getCardType()	CardType
m	getValidThrough()	GregorianCalendar
m	getCCV()	String
m	setCardNumber(String)	void
m	setCardType(CardType)	void
m	setValidThrough(GregorianCalendar)	void
m	setCCV(String)	void

PersonManager		
f	customers	ArrayList<Customer>
f	employees	ArrayList<Employee>
f	customerCounter	int
f	employeeCounter	int
m	getAndIncrementCustomerCounter()	int
m	getAndIncrementEmployeeCounter()	int
m	addCustomerIntoCustomers(Customer)	void
m	addEmployeeIntoEmployees(Employee)	void
m	getCustomerIndexInCustomerList(Customer)	int
m	getEmployeeIndexInEmployeeList(Employee)	int
m	removeCustomerFromCustomers(Customer)	void
m	removeEmployeeFromEmployees(Employee)	void

Employee		
f	employeeID	int
f	salary	float
f	typeOfActivity	TypeOfActivity
m	employeeHelpsCustomer()	boolean
m	getRandomBoolean()	boolean
m	getEmployeeID()	int
m	getTypeOfActivity()	TypeOfActivity
m	setTypeOfActivity(TypeOfActivity)	void
m	getSalary()	float
m	setSalary(float)	void
m	employeeRepairsElectricCar(ElectricCar)	void
m	employeeRepairsCombustionCar(CombustionCar)	void
m	employeeRefuelsCar(CombustionCar)	void

Customer		
f	customerID	int
f	customerLevel	CustomerLevel
f	paymentMethod	PaymentMethod
f	fuelRentals	ArrayList<FuelRental>
f	electricRentals	ArrayList<ElectricRental>
m	getCustomerID()	int
m	getCustomerLevel()	CustomerLevel
m	getElectricRentals()	ArrayList
m	getElectricRentalWithIndex(int)	ElectricRental
m	getFuelRentals()	ArrayList
m	getFuelRentalWithIndex(int)	FuelRental
m	getPaymentMethod()	PaymentMethod
m	setCustomerLevel(CustomerLevel)	void
m	setPaymentMethod(PaymentMethod)	void
m	customerNeedHelp(Employee)	boolean
m	getElectricCarIndexInElectricRentalList(ElectricCar)	int
m	getCombustionCarIndexInFuelRentalList(CombustionCar)	int
m	rentAnElectricCar(ElectricCar, CarManagementSystem, LocalDate, int, int, int, int, int, RentalManager)	void
m	modifyAnElectricRental(ElectricRental, ElectricCar, CarManagementSystem, LocalDate, int, int, int, int, int, RentalManager)	void
m	cancelElectricRental(ElectricRental)	void
m	rentAFuelCar(RentalManager, CombustionCar, CarManagementSystem, LocalDate, int, int, int, int, int, int)	void
m	modifyAREgularRental(FuelRental, RentalManager, CombustionCar, CarManagementSystem, LocalDate, int, int, int, int, int, int)	void
m	cancelFuelRental(FuelRental)	void
m	customerDamagesAnElectricCar(ElectricCar)	void
m	customerDamagesAFuelCar(CombustionCar)	void

Admin		
m	getRandomBoolean()	boolean
m	approveRentalModification(ElectricRental)	boolean
m	approveRentalModification(FuelRental)	boolean
m	deleteFuelRental(FuelRental)	void
m	deleteElectricRental(ElectricRental)	void
m	resolveProblem()	boolean
m	deleteEmployee(Employee, PersonManager)	void
m	deleteCustomer(Customer, PersonManager)	void

CustomerTest		
m	the_customer_level_may_be_changed()	void
m	the_payment_method_may_be_changed()	void
m	the_customer_should_be_helped()	void
m	the_right_index_should_be_returned_with_the_given_car()	void
m	a_customer_damages_a_car()	void
m	a_customer_is_able_to_rent_a_combustion_car()	void
m	a_customer_is_able_to_modify_an_electric_rental()	void
m	a_customer_is_able_to_cancel_a_fuel_rental()	void

EmployeeTest		
m	an_employee_type_of_activity_can_be_changed()	void
m	an_employee_repairs_an_electric_car()	void
m	an_employee_refuels_a_combustion_car()	void
m	the_address_of_a_person_can_be_changed()	void
m	the_name_of_a_person_can_be_changed()	void

PersonManagerTest		
m	testing_of_the_counter_return_and_increment_function()	void
m	after_creating_a_customer_it_should_have_been_added_to_customers_list()	void
m	after_deleting_a_customer_it_should_not_longer_be_in_customer_list()	void

PersonAddressTest

PaymentMethodTest

PersonTest

Package Car

Car		
f	type	Type
f	brand	String
f	model	String
f	state	State
f	odometer	long
f	permission	Enum
f	price	float
f	location	Location
m	changeCarState(State)	void
m	getCarState()	State
m	setNewLocation(double, double)	void
m	getLocation()	Location
m	getGPSLatitude()	double
m	getGPSLongitude()	double
m	getPrice()	float
m	getOdometer()	long

CarManagementSystem		
f	combustionCarsList	ArrayList<CombustionCar>
f	electricCarsList	ArrayList<ElectricCar>
m	addCarIntoElectrics(ElectricCar)	void
m	addCarIntoCombustion(CombustionCar)	void
m	deleteCarFromElectric(ElectricCar)	void
m	deleteCarFromCombustion(CombustionCar)	void
m	getSizOfElectricCarsList()	int
m	getSizOfCombustionCarsList()	int
m	getCarIDFromCombustion(CombustionCar)	int
m	getCarIDFromElectric(ElectricCar)	int

Location		
f	GPSLatitude	double
f	GPSLongitude	double
m	setGPSLatitude(double)	void
m	setGPSLongitude(double)	void
m	getGPSLatitude()	double
m	getGPSLongitude()	double

CombustionCar		
f	tankSize	double
f	fuelLevel	double
f	consumption	double
f	transmission	Transmission
f	fuelType	FuelType
m	getFuelLevel()	double
m	calculateConsumedFuel()	double
m	getTankSize()	double
m	getConsumption()	double
m	setFuelLevel(double)	void
m	getTanked()	void

ElectricCar		
f	range	float
f	chargePercent	float
m	getChargePercent()	float
m	getRange()	float
m	setChargePercent(float)	void
m	getChargedUp()	void

CarManagementSystemTest		
m	testAddElectricCarToArrayList()	void
m	testAddCombustionCarToArrayList()	void
m	testDeleteCarFromCombustion()	void
m	testGetSizeOfElectricCarsList()	void
m	testGetCarIDFromCombustion()	void

LocationTest		
m	setGPS()	void
m	getGPSLocation()	void

CarTest		
---------	--	--

ElectricCarTest		
m	testSetNewLocation()	void
m	testGetChargedAndSetChargePercent()	void
m	testChangeCarState()	void

CombustionCarTest		
m	testChangeCarState()	void
m	testGetTankedAndSetFuelLevel()	void

Package ParkingArea

ParkingArea		
f	parkID	int
f	maxCapacity	int
f	parkingAreaAddress	ParkingAreaAddress
f	carsInStation	ArrayList<CombustionCar>
f	availableCars	ArrayList<CombustionCar>
f	notAvailableCars	ArrayList<CombustionCar>
m	assignCarToStation(CombustionCar)	void
m	getIndexInStationCarIDList(CombustionCar)	int
m	removeCarFromStation(CombustionCar)	void
m	carsBeingUsed(CombustionCar)	void
m	carsNoLongerBeingUsed(CombustionCar)	void
m	numberOfCarsAssignedToStation()	int
m	getParkingAreaAddress()	ParkingAreaAddress
m	getMaxCapacity()	int
m	getCarsInStation()	ArrayList<CombustionCar>



ElectricParkingArea		
f	maxElectricCarCapacity	int
f	electricCarsInStation	ArrayList<ElectricCar>
f	availableElectricCars	ArrayList<ElectricCar>
f	notAvailableElectricCars	ArrayList<ElectricCar>
m	getMaxElectricCarCapacity()	int
m	assignElectricCarToStation(ElectricCar)	void
m	removeElectricCarFromStation(ElectricCar)	void

ParkingAreaManager		
f	ParkingAreas	ArrayList<ParkingArea>
f	ElectricParkingAreas	ArrayList<ElectricParkingArea>
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

ParkingAreaAddress		
f	ZIP	String
f	city	String
f	street	String
f	streetNumber	String
m	getZIP()	String
m	getCity()	String
m	getStreet()	String
m	getStreetNumber()	String

ParkingAreaManagerTest		
m	testing_of_the_counter_return_and_increment_function()	void
m	testing_add_parking_area_into_parking_areas()	void
m	testing_removing_parking_area_from_parking_areas()	void
m	testing_get_size_from_parking_area()	void
m	testing_get_parkID_from_electric_parking_areas()	void

ParkingAreaTest		
m	test_assign_car_to_station()	void

ParkingAreaAddressTest		
------------------------	--	--

ElectricParkingAreaTest		
-------------------------	--	--

Package Rental

Rental		
f	rentalID	int
f	carID	int
f	customerID	int
f	rentalPrice	float
f	odometerBefore	long
f	odometerAfter	long
f	date	LocalDate
f	departureTime	GregorianCalendar
f	arrivalTime	GregorianCalendar
m	calculateElapsedDays()	int
m	setOdometerAfter()	void
m	getOdometerAfter()	long
m	getCarID()	int
m	getRentalID()	int
m	getCustomerID()	int
m	getDate()	LocalDate
m	getDepartureTime()	GregorianCalendar
m	getArrivalTime()	GregorianCalendar

RentalManager		
f	ElectricRentals	ArrayList<ElectricRental>
f	FuelRentals	ArrayList<FuelRental>
f	RentalCounter	int
m	getAndIncrementCounter()	int
m	addRentalIntoElectricRentals(ElectricRental)	void
m	addRentalIntoFuelRentals(FuelRental)	void
m	removeRentalFromElectricsRentals(ElectricRental)	void
m	removeRentalFromFuelRentals(FuelRental)	void
m	getSizeOfElectricRentals()	int
m	getSizeOfFuelRentals()	int
m	getRentalIDFromElectricRentals(ElectricRental)	int
m	getRentalIDFromFuelRentals(FuelRental)	int
m	returnElectricRentalWithIndex(ElectricRental)	int
m	returnFuelRentalWithIndex(FuelRental)	int

ElectricRentalTest		
m	testSetChargePercentAfter()	void

RentalManagerTest		
m	testGetAndIncrementCounter()	void
m	testAddRentalIntoElectricRentals()	void
m	testRemoveRentalFromElectricRentals()	void
m	testGetSizeOfElectricRentals()	void
m	testGetRentalIDFromElectricRentals()	void
m	testReturnElectricRentalWithIndex()	void

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

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

RentalTest		
m	testElapsedDays()	void
m	testSetOdometerAfter()	void

FuelRentalTest		
m	testSetFuelLevelAfter()	void

Lessons Learned

- ◆ Vererbung ist ein sehr nützliches Werkzeug, trotzdem sollte man nicht es übernutzen (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