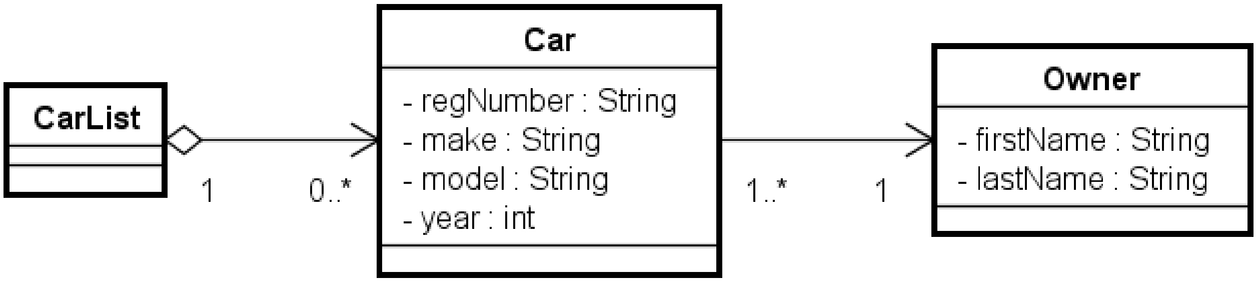
Programming Exercises - PRO1 - Session 25

The purpose of today's exercises is to go through the steps of creating the following Java model, and to make it possible to store it in a binary file.



If you need some data to add to the model when it's finished, then either create some yourself, or use this list of cars and their owners:

```
AL19742,Ferrari,F50,1997,Allan,Henriksen
HE23477,Audi,A7,2014,Allan,Henriksen
XY54679,Toyota,Yaris,2007,Charlie,Pace
JA72921,Seat,Mii,2012,Claire,Littleton
RT90456,Mercedes,W212,2010,Jack,Shephard
TY86934,Dodge,Viper,2010,James,Ford
OS38067,Volkswagen,Golf,2012,John,Locke
MD21739,Volvo,S80,2009,Juliet,Burke
JK42234,Ford,Focus,2008,Kate,Austen
```

Exercise 25.01

First implement a class `Owner` representing the owner of a car with a first name and last name. The class should have:

- Two instance variables: `firstName` and `lastName` both of type `String.`
- A constructor setting both instance variables.
- Set and get methods for both instance variables.
- A `toString` method returning a String with the properties of an owner (in a single line).
- An `equals` method returning true if two `Owner`-objects have the same first and last name.

Exercise 25.02

Implement a class `Car` representing a car with a registration number, a make, a model, the year of manufacture and the owner of the car. The class should have:

- Instance variables: `regNumber`, `make`, `model`, all of type `String`, `year` of type `int`, and `owner` as an `Owner`-object.
- A constructor setting all five instance variables.
- Get methods for all instance variables.
- A `toString` method returning a `String` with the properties of a car (in a single line).
- An `equals` method returning true if two `Car`-objects have the same registration number, make, model, year, and owner.

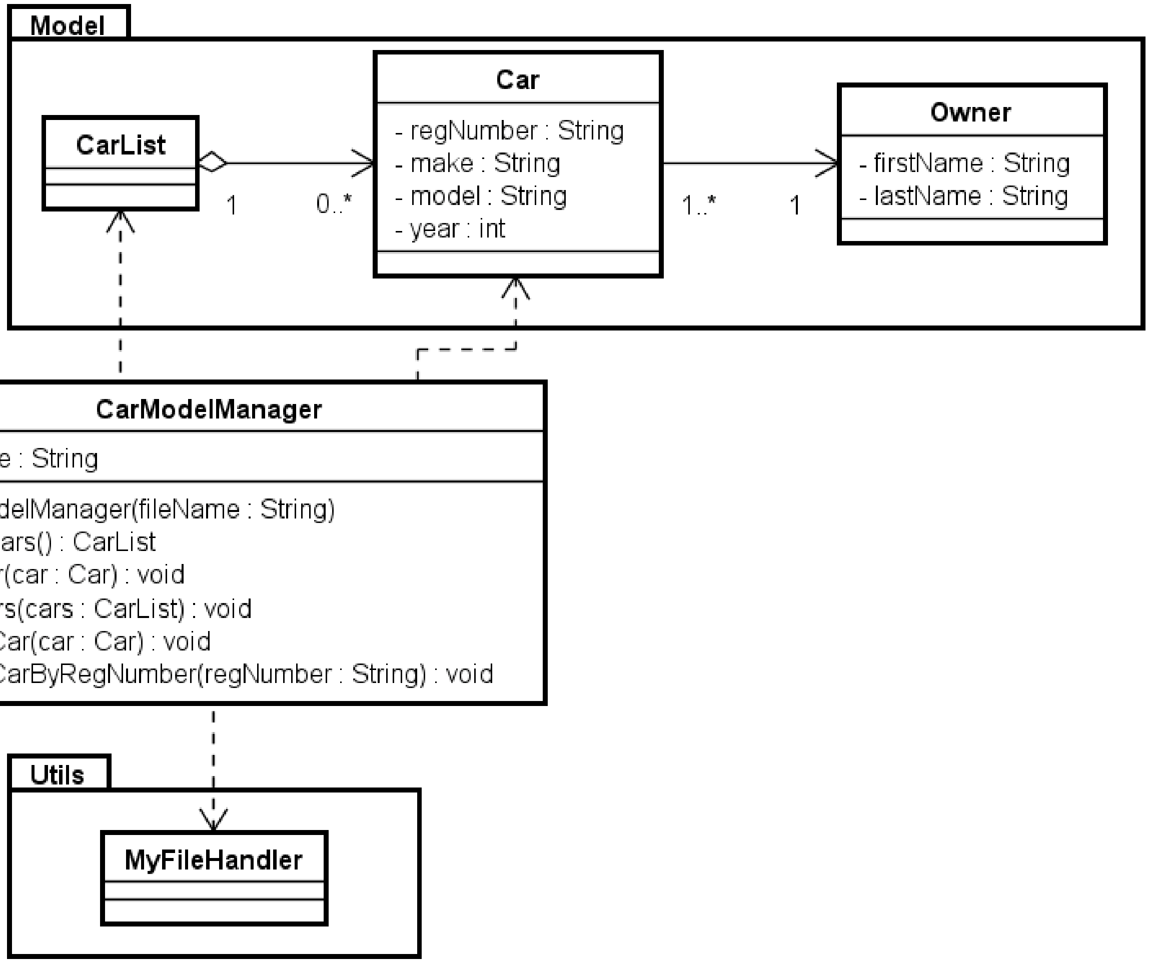
Exercise 25.03

Implement a class `CarList` representing a collection of cars (`Car`-objects). The class should have:

- One instance variable: `cars` as a collection of `Car`-objects, e.g. `ArrayList<Car>.`
- A zero-argument constructor initializing the car list.
- A method `size` that returns the size of the car list.
- A method `indexOfRegNumber` that takes an argument of type `String` (representing the registration number for a car) and returns the index from the list of the `Car`-object having this specific registration number. If the list does not contain a car with this specific registration number, then the method should return `-1.`
- A method `getCar` that takes an argument of type `int` representing an index and returns the `Car`-object at the specific index.
- An overloaded version of method `getCar` that takes an argument of type `String` representing the registration number for a car and returns the `Car`-object with the specific registration number. If the list does not contain a car with this specific registration number, then the method should return `null`.
- A method `addCar` that takes a `Car`-object as argument and adds this to the car list if the list does not already contain a car with the same registration number.
- A `toString` method returning a `String` with the properties of each car in the list (each car on a separate line).

Exercise 25.04

Implement a class `CarModelManager` representing the access point to the model (`Owner`, `Car`, `CarList`) and containing methods using the file functionality (`MyFileHandler`)



The class `CarModelManager` should have:

- An instance variable `fileName` holding the name of the file where car information is stored
- A method `getAllCars` that reads all cars from the file and returns them as a `CarList`-object
- A method `addCar` that takes a single `Car` object and adds it to the list of stored cars
- A method `addCars` that takes a `CarList` object and adds all cars inside it to the list of stored cars
- A method `deleteCar`, that takes a `Car` object, and removes it from the list of stored cars
- A method `deleteCarByRegNumber` that takes a `String` with a registration number, and removes the car with that give registration number from the list of stored cars, if it exists

Exercise 25.05

Finally create a test class where you test the functionality in the classes you have created.