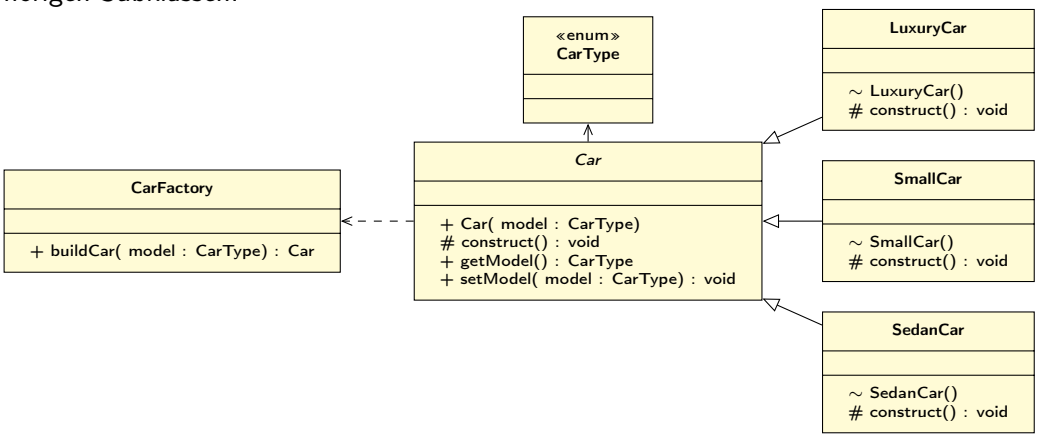
In diesem Aufgabenblatt werden Sie das Factory und das Singleton Design Pattern anwenden.

1 Factory

In dieser Aufgabe wird das Factory Pattern angewendet. Eine *Car* Factory Klasse soll Objekte der Subklasse von *Car* erstellen. *Car* ist eine abstrakte Klasse, 3 konkrete Klassen, *LuxuryCar, SmallCar, SedanCar*, implementieren diese abstrakte Klasse. Die 3 *Car*-Subklassen unterscheiden sich in im Attribut *model*, das den Type *CarType* hat und die folgenden Werte annehmen kann: *SMALL, SEDAN, LUXURY*.
Implementieren sie entsprechend einen Enummeration für den Type CarType:

```
public enum CarType {
SMALL, SEDAN, LUXURY
}
```

Folgendes Diagram zeigt die *CarFactory* Klasse, sowie die abstrakte *Car* Klasse und die dazugehörigen Subklassen.



Printen Sie beim Aufruf der *construct()* Methode in den Subklassen, welche Art von Car gerade erstellt wird, z.B. "SedanCar wird fabriziert."
Verwenden Sie die folgende Klasse um ihre Implementation zu testen:

```
public class TestFactoryPattern {
   public static void main(String[] args) {
   System.out.println(CarFactory.buildCar(CarType.SMALL));
   System.out.println(CarFactory.buildCar(CarType.SEDAN));
   System.out.println(CarFactory.buildCar(CarType.LUXURY));
   }
}
```

2 Singleton

In dieser Aufgabe wird das *CarFactory* Objekt zu einem Singleton Objekt gemacht. Ein Singleton Objekt kann nur ein einziges Mal instanziiert werden. Im Fall eines Factory Objekts, dass dann beliebige viele Objekte erstellen kann, ist dies wünschenswert, da mehrere Factory Objekte nur unnötig Resourcen verwenden würden ohne einen zusätzlichen Nutzen.
Übernehmen Sie die Quelldateien aus der ersten Aufgabe und passen Sie nur die Klasse *CarFactory* an, sodass diese nur ein einziges Mal instanziiert werden kann und somit ein Singleton ist.