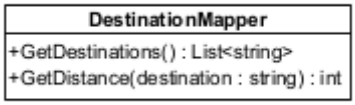
Airport Opgave

Table of Contents

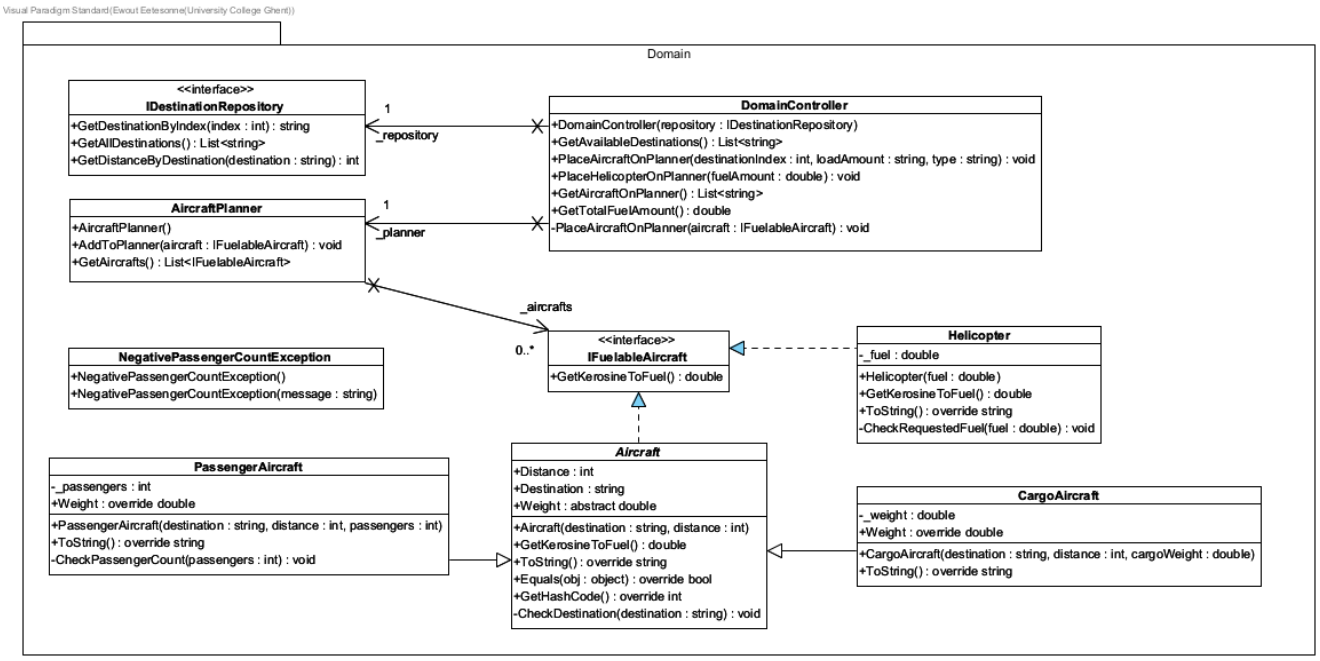
1. Gegeven

Namespace Persistentie, klasse `DestinationMapper`



Vul in onderstaande namespaces de ontbrekende code aan. Respecteer telkens het UML-schema.

2. Namespace Domain



2.1. Klasse NegativePassengerCountException

Voorzie een klasse `NegativePassengerCountException`.

NegativePassengerCountException

Deze `NegativePassengerCountException` wordt gegooid indien het aantal passagiers van een passagiervliegtuig minder dan nul is.

2.2. Interface IFuelableAircraft

Voorzie een interface `IFuelableAircraft`

2.3. Interface IDestinationRepository

Voorzie een interface `IDestinationRepository`

2.4. Abstracte klasse Aircraft

Voorzie de abstracte klasse Aircraft

- Voorzie een constructor met 2 parameters.
  - De destination (string) kan achteraf niet meer wijzigen en moet ingevuld zijn (dus niet `null` en niet leeg). In de hulpmethode `CheckDestination` wordt een `ArgumentException` met passende foutboodschap gegooid indien de omschrijving niet voldoet.
  - Distance wordt uit de persistence laag gehaald via de `GetDistanceByDestination` methode.
- De weight wordt berekend in de implementaties.
- Vermenigvuldig in de `GetKerosineToFuel` methode `Distance` met `Weight` om te weten hoeveel

brandstof een vliegtuig nodig heeft.

- Implementeer de `Equals` & `GetHashCode` methodes en zorg er voor dat twee vliegtuigen beschouwd worden als hetzelfde indien ze dezelfde bestemming hebben.
- Voorzie een gegeneraliseerde `ToString`-methode: leg de uitvoer van de klassen `PassengerAircraft` en `CargoAircraft` naast elkaar om het gemeenschappelijke deel hier te implementeren.
  - Voorbeeld (PassengerAircraft):

    > PassengerAircraft with destination CAN requires 32424000 L of fuel containing 50 passengers

  - Voorbeeld (CargoAircraft):

    > CargoAircraft with destination HND requires 4736000 L of fuel with a cargo of 500 kilograms

    De in fluo gekleurde waarden verwijzen naar variabele waarden.

2.5. Klasse PassengerAircraft

- Het aantal passagiers mag niet lager dan 0 zijn. Er wordt een `NegativePassengerCountException` met passende foutboodschap gegooid indien het aantal passagiers niet voldoet.
- Het gewicht van dit vliegtuig wordt berekend door het aantal passagiers te vermenigvuldigen met 70.
- Specializeer indien nodig de `ToString`-methode.
  - Voorbeeld (PassengerAircraft):

    > PassengerAircraft with destination CAN requires 32424000 L of fuel containing 50 passengers

    De in fluo gekleurde waarden verwijzen naar variabele waarden.

2.6. Klasse CargoAircraft

- Het gewicht dat meegegeven wordt in de constructor mag je hier rechtstreeks toekennen aan de property. Er moeten geen verdere berekeningen op gebeuren.

  > CargoAircraft with destination HND requires 4736000 L of fuel with a cargo of 500 kilograms

  De in fluo gekleurde waarden verwijzen naar variabele waarden.

2.7. Klasse Helicopter

- Voorzie een constructor met een parameter
  - Fuel (**niet wijzigbaar**) moet groter dan 100 zijn en moet deelbaar zijn door 10. In de hulpmethode `CheckRequestedFuel` wordt een `ArgumentException` met passende foutboodschap gegooid indien de waarde niet voldoet.
- Voorzie de nodige methodes om de klasse concreet te maken. Op een helikopter mag zonder berekening bepalen hoeveel kerosine er getankt moet worden.
- Voorzie een `ToString`-methode.
  - Voorbeeld:

    > Helicopter that needs 5000 L of fuel

    De in fluo gekleurde waarden verwijzen naar variabele waarden.

2.8. Klasse AircraftPlanner

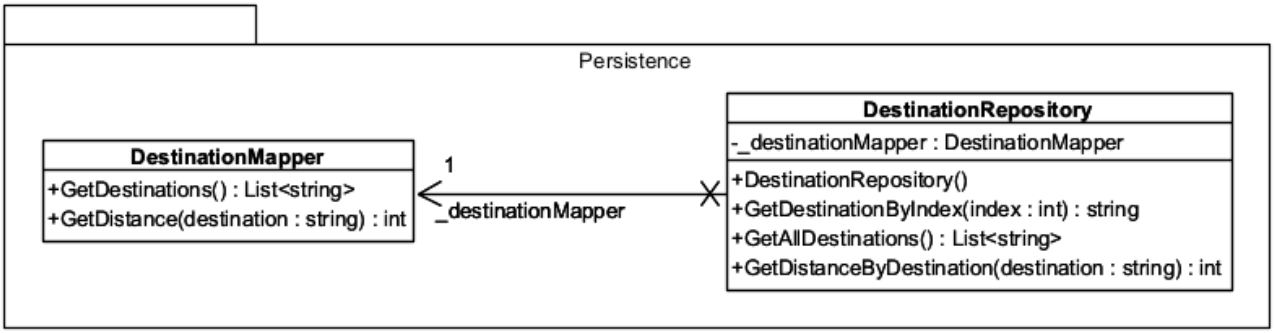
- De methode `AddToPlanner` zorgt ervoor dat om het even welk `IFuelableAircraft` kan toegevoegd worden aan de dynamische lijst `_aircrafts`.
- De methode `GetAircrafts` geeft de lijst van vliegtuigen terug.

2.9. Klasse DomeinController

- Voorzie een constructor met de nodige *dependency injection*.
- Let ook op de multipliciteit met de klasse AircraftPlanner!
- De methode `GetAvailableDestinations` geeft een lijst terug met daarin alle mogelijke bestemmingen in tekstvorm.
- De methode `PlaceAircraftOnPlanner` zoekt aan de hand van `destinationIndex` de bestemming op (via de repository) en voegt dan het vliegtuig (op basis van het type: "Passenger" of "Cargo") toe aan de lijst met vliegtuigen in de planner na de distance op te halen in de repository. De parameter `loadAmount` bevat voor een passagiersvliegtuig het aantal `passengers`, voor een cargo de `cargoweight`.
- De methode `PlaceHelicopterOnPlanner` maakt een helikopter met de opgegeven hoeveelheid brandstof en voegt deze toe aan de planner.
- De methode `GetAircraftOnPlanner` maakt een lijst van strings met de informatie over de vliegtuigen op de planner. Zie hiervoor de twee mogelijke voorbeeld uitvoeren verder in dit document.
- De methode `GetTotalFuelAmount` gebruikt de lijst met vliegtuigen de hoeveelheid brandstof die elk vliegtuig nodig heeft op te tellen en geeft deze waarde terug.

3. Namespace Persistentie

3.1. Klasse DestinationRepository



- Voorzie een constructor (let op de multipliciteit van de associatie met de mapper klasse)
- De methode `GetAllDestinations` geeft de lijst van mogelijke bestemmingen terug op index. Deze index in de lijst van mogelijke bestemmingen kan later gebruikt worden om een specifieke bestemming op te vragen.
- De methode `GetDestinationByIndex` geeft de `Destination` op de opgegeven index uit de lijst van beschikbare bestemmingen terug.
- De methode `GetDistanceByDestination` geeft een afstand terug aan de hand van de meegegeven `Destination` string.

indien de index buiten de lijst valt wordt een `IndexOutOfRangeException` gegooid, met een duidelijke foutboodschap.

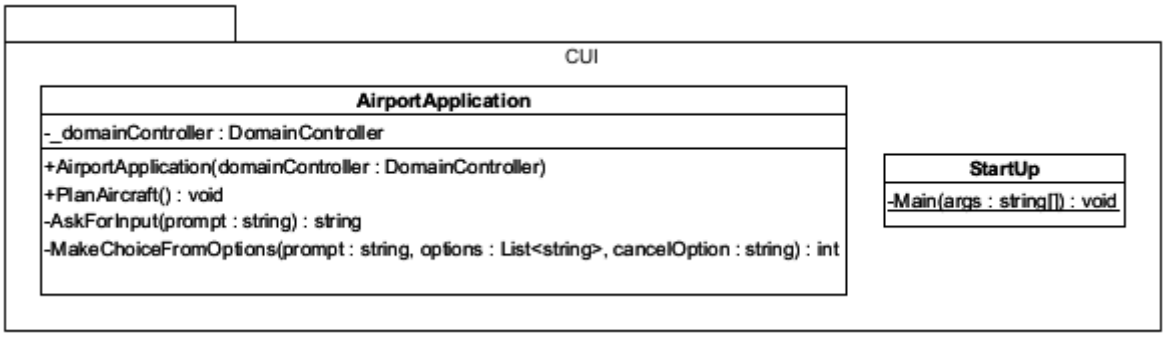
4. Namespace Cui

4.1. Klasse Startup

Voorzie de StartUp-klasse die de AirportApplication op een correcte manier opstart in een Main methode:

- Maak een instantie aan van `DestinationRepository`
- Maak een instantie aan van `DomeinController` en injecteer de repository
- Maak een instantie aan van `AirportApplication` en injecteer de domeincontroller

4.2. Klasse AirportApplication



Om onderstaande functionaliteit te implementeren ben je vrij om zelf hulpmethodes aan te maken.

Verloop van de applicatie: Bij het opstarten krijgt de gebruiker volgend keuzemenu te zien.

```
Menu
1. Add aircraft
2. Add helicopter
0. Stop
Pick an option
```

Als hij geen juiste invoer geeft, dan komt er een foutmelding en mag hij opnieuw proberen.

```
Menu
1. Add aircraft
2. Add helicopter
0. Stop
Pick an option
5
ArgumentException Choice must be between 0 & 2
```

OF

```
Menu
1. Add aircraft
2. Add helicopter
0. Stop
Pick an option
i don't know
FormatException Input string was not in a correct format.
```

4.3. Optie 1

Als hij optie 1 kiest, krijgt de gebruiker de lijst van beschikbare bestemmingen te zien. De gebruiker moet een keuze maken, eerst het type vliegtuig invoeren, en dan het aantal passagiers of het

gewicht van de lading ingeven. Daarna wordt opnieuw het hoofdmenu getoond, ongeacht of deze keuzes juist waren of fout. In het eerste geval wordt het opgegeven vliegtuig toegevoegd aan de planner, maar in het tweede geval wordt enkel een foutmelding getoond en wordt er niks toegevoegd.

```
Menu
1. Add aircraft
2. Add helicopter
0. Stop
Pick an option
1

Pick a destination
1. CAN
2. ATL
3. HND
4. DEL
5. DXB
6. IST
7. CDG
8. LHR
9. MEX
10. SGN
0. Don't select a destination
Pick an option
1

Pick an aircraft type
1. Passenger
2. Cargo
0. Don't select a type
Pick an option
1
Please provide number of persons
50
```

OF fout in aantal stuks

```
Pick an aircraft type
1. Passenger
2. Cargo
0. Don't select a type
Pick an option
1
Please provide number of persons
abc
FormatException Input string was not in a correct format.

Menu
1. Add aircraft
2. Add helicopter
0. Stop
Pick an option
```

4.4. Optie 2

Als de gebruiker optie 2 kiest in het hoofdmenu, dan moet hij de gewenste hoeveelheid brandstof voor de helikopter invoeren. Ook hier geldt: als deze waarde niet aan de voorwaarden voldoet, dan wordt een foutmelding getoond. Als de gebruiker wel een geldige waarde invoert, dan wordt de helikopter toegevoegd aan de planner. In beide gevallen worden we daarna terug naar het hoofdmenu geleid.

```
Menu
1. Add aircraft
2. Add helicopter
0. Stop
Pick an option
2
Please provide amount of fuel
5000

Menu
1. Add aircraft
2. Add helicopter
0. Stop
Pick an option
```

OF

```
Menu
1. Add aircraft
2. Add helicopter
0. Stop
Pick an option
2
Please provide amount of fuel
abc
FormatException Input string was not in a correct format.

Menu
1. Add aircraft
2. Add helicopter
0. Stop
Pick an option
```

4.5. Optie 0

Bij invoer van keuze 0 in het hoofdmenu, tenslotte, eindigt het programma met het tonen van de planning.

Hierbij zijn verschillende mogelijkheden:

- Er werd geen enkel geldig item toegevoegd aan de planner. In dat geval wordt hier gewoon een melding rond gemaakt.

  ```
  Menu
  1. Add aircraft
  2. Add helicopter
  0. Stop
  Pick an option
  0
  No flights were added to planner!
  ```

- Er zijn allerlei vluchten toegevoegd aan de planner, variërend van passagiers- en cargovliegtuigen tot helikopters. Er wordt een overzicht getoond van deze vliegtuigen, samen met de totale hoeveelheid brandstof die vereist is.

```
Menu
1. Add aircraft
2. Add helicopter
0. Stop
Pick an option
0
PassengerAircraft with destination CAN requires 32424000L of fuel containing 50
passengers
Helicopter that needs 5000 liters of fuel
CargoAircraft with destination SGN requires 495600000L of fuel with a cargo of
50000 kilograms
Total fuel required 528029000L
```