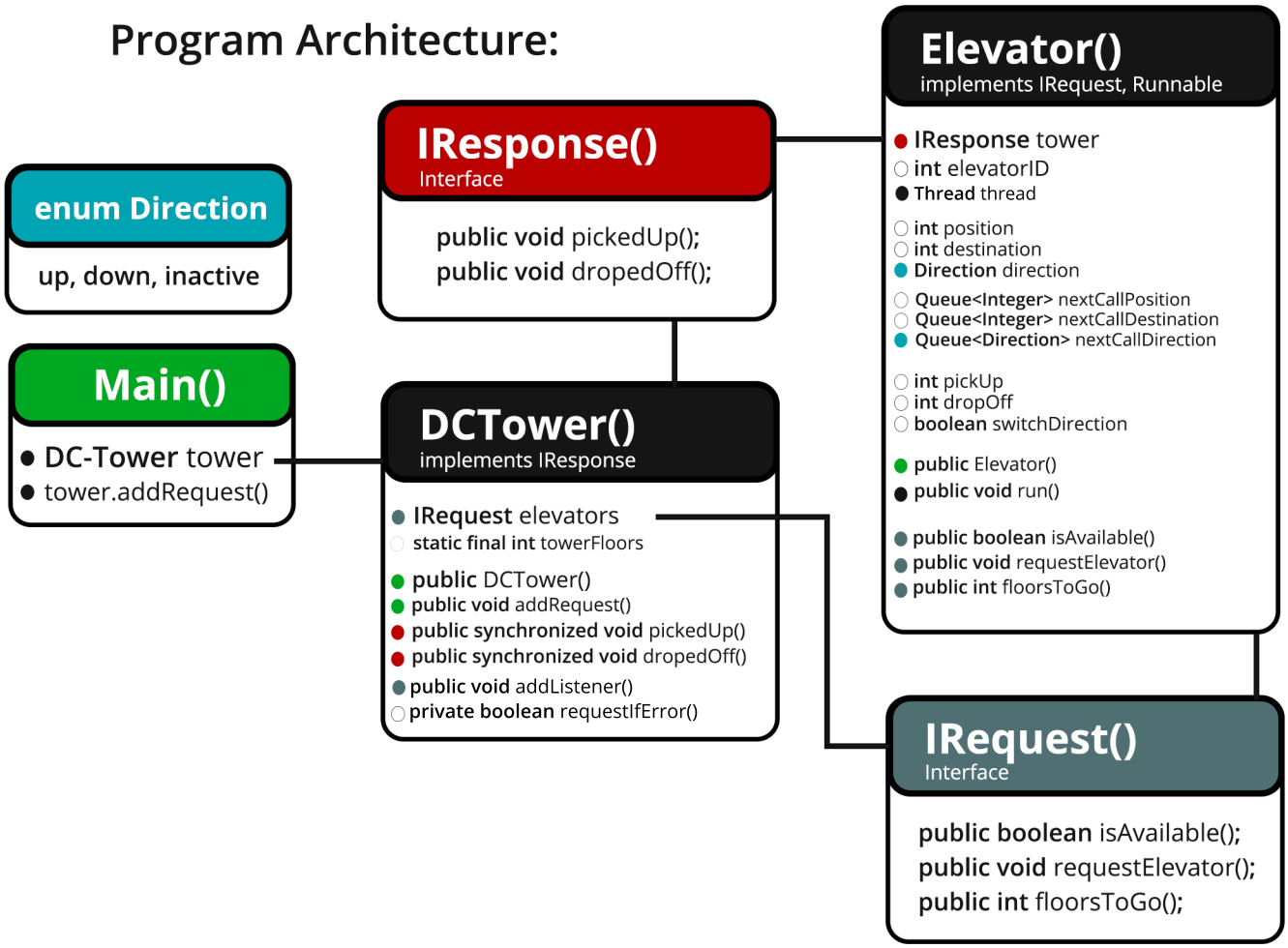
Experience.
Create.
Inspire.

DC Tower Elevator Challenge

Elisei Iovuta

Program Architecture:



IBM

IBM Client Innovation Center
Austria

DC Tower Elevator Challenge

Elisei Iovuta

For the “DC Tower Elevator Challenge”, I have chosen to use two classes:
(see architecture on page 1)

A simulation class **DCTower()** to run the requests and a Runnable class **Elevator()** that simulates the elevator and its movement between the floors of the DC Tower.

I have further implemented two Interfaces **IRequest()** and **IResponse()** to achieve communication between the Elevators and the Tower.
Each Elevator has a IResponse instance of the Tower to be called on when a pick up or a drop off has been achieved.

The Tower has a list of IRequest (Elevators) that are called upon whenever a request was sent to check for an available elevator and send it to solve the current request. If all the elevators are busy, the interface method **floorsToGo()** returns how many floors an elevator has to go through to get to the requested call position. The elevator with the fewest floors to go gets the request.

If an already active elevator gets a request, the request data is saved in queues and is used to make the request again once the elevator has reached an inactive state.

As long as the request moves only in one direction to an elevator in a lower or an equal position than the request position
for ex. addRequest(0,15,up) Elevator current position:0

the elevator moves incrementally upwards and sets off the **pickedUp()** method when the request position is met.
However, if an elevator has to first move in one direction to pick up the request and then change its direction, the requestElevator() method has a piece of code that checks if this switch must take place, changes the **switchDirection** variable to **true** and sets the pick-up position as a temporary destination.

When the elevator destination is met, the program checks if a switch has to take place, if so, the drop off destination is set as the new tower destination and the elevator direction is changed the other way, after this the **switchDirection** is set to **false** as a switch has taken place.

The Threads are started immediately as the elevators are created, a way to further better the code would be to put the threads to sleep until a request comes in.

DC Tower Elevator Challenge

Elisei Iovuta

Threads:

The elevator movement is simulated by a thread that moves the elevator position incrementally by one in whichever predetermined direction set. When the pick-up position matches the current elevator position, the Interface method pickedUp() is called sending the elevator ID and the current position to the Tower.

When the elevator destination and its position match, the program checks if a switch must take place, if not, the elevator calls the interface method dropedOff() that informs the Tower of its success and sets itself as inactive. At this stage if a request was made for the elevator in the meantime, the queues are checked if they are empty (only one queue needs to be checked).

If the queue is empty, the elevator remains inactive until a request comes in.

If the queue is not empty the requestElevator() method is called with the data of the queue heads.

Direction:

I have chosen to use the Enum Direction to simulate the three stages an elevator can present itself, namely; ***up***, ***down*** and ***inactive***

What was solved?

The elevator simulation was successful in bringing someone from the ground to a higher floor as well as from a higher floor back to the ground floor. The console lets you know if a thread was started, from what floor the request was made, the request destination and the direction of the trip.

The program successfully adds requests to the elevator queue when all elevators are active.
Once a request comes in, all elevators are notified of this.
The elevator can only take one request at the time.

Due to threads having different finishing rates, the IResponse methods are synchronized to avoid critical junctions.

This is my Solution to the **DC Tower Elevator Challenge** **Elisei Iovuta**