Adding new features with TDD and JUnit 5

Cătălin Tudose

PHD

https://www.linkedin.com/in/catalin-tudose-847667a1

Overview

Use JUnit 5 more effectively

Introduce new features to the application

- Business logic understanding
- Activity diagrams
- Class diagram

Use JUnit5 new features

Transpose scenarios in JUnit 5

Fixing tests implementing business logic

Demo

Use JUnit 5 more effectively

Introduce @Nested inner classes

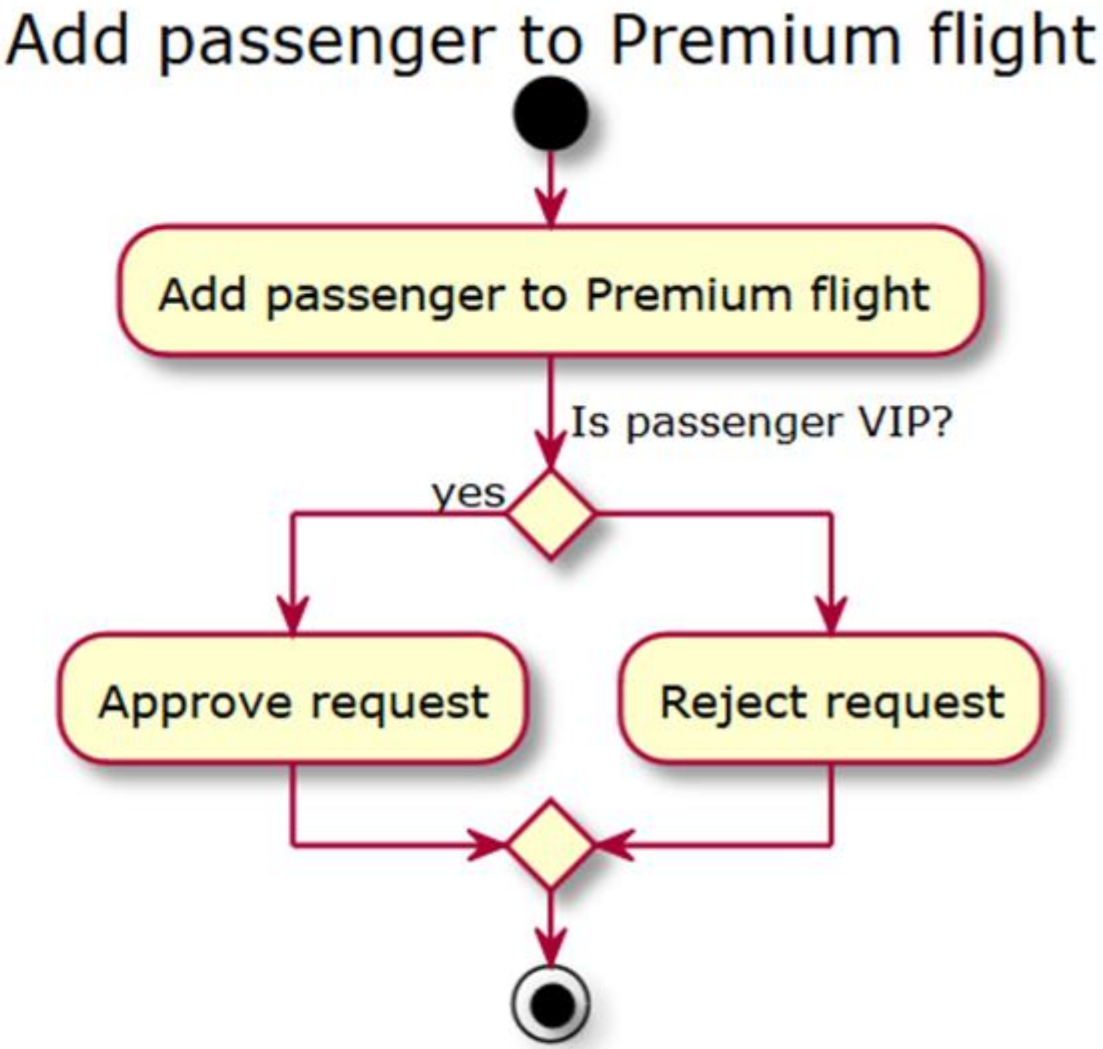
Separate the logic of testing at the level of inner classes

Use the @DisplayName and @BeforeEach annotations

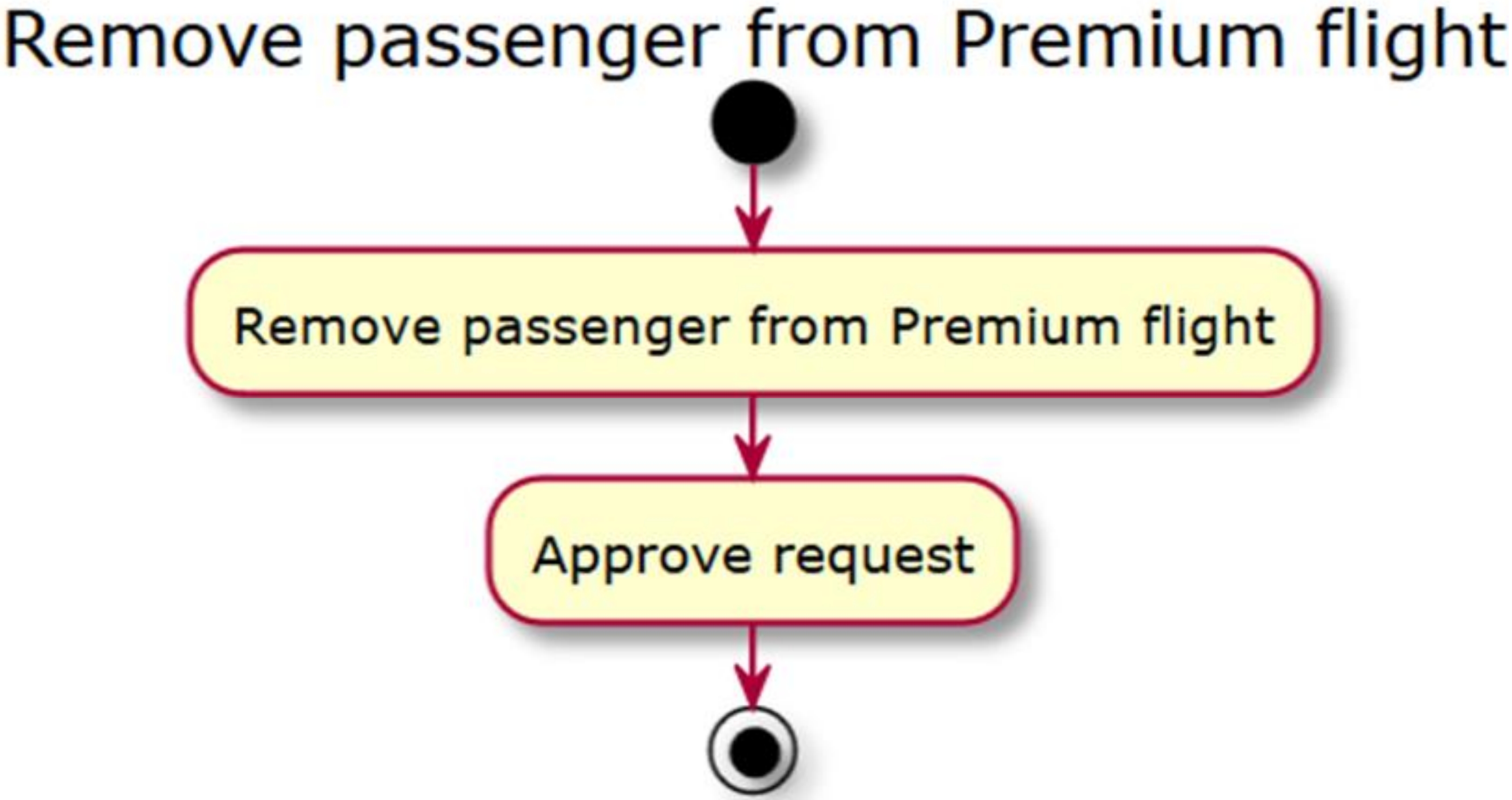
Run the tests

Adding a Premium Flight

Adding Passenger New Business Logic


Add passenger to Premium flight
Add passenger to Premium flight
Is passenger VIP?
yes
Approve request
Reject request

Removing Passenger New Business Logic



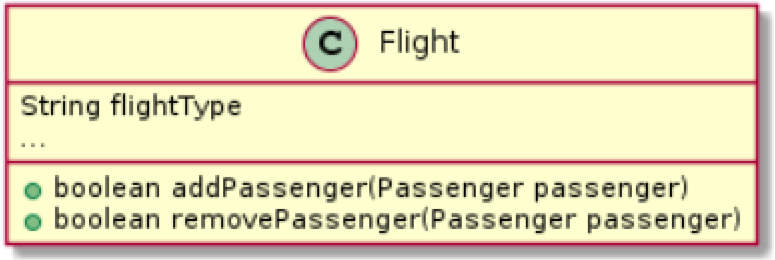
Remove passenger from Premium flight

Remove passenger from Premium flight

Approve request

Open/close principle

Using conditional



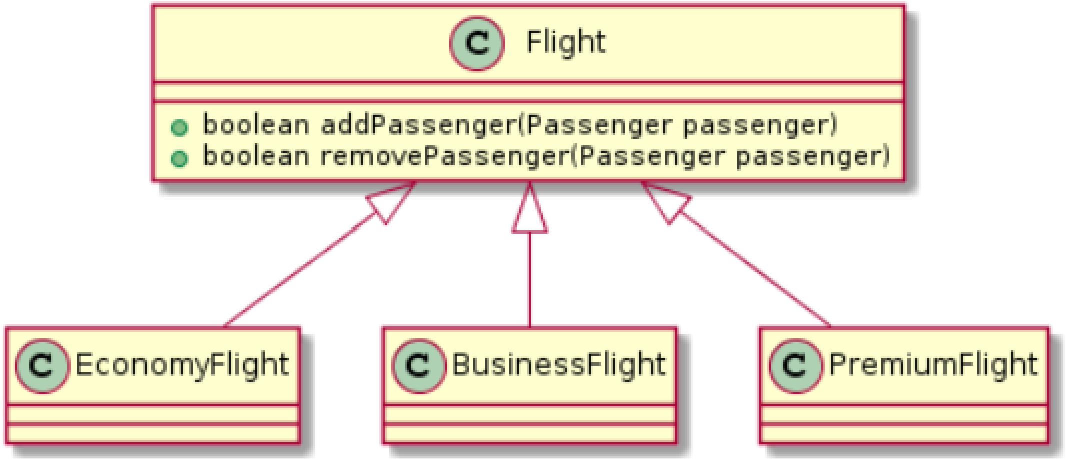
Flight

String flightType

...

- boolean addPassenger(Passenger passenger)
- boolean removePassenger(Passenger passenger)

Using polymorphism



Flight

- boolean addPassenger(Passenger passenger)
- boolean removePassenger(Passenger passenger)

EconomyFlight

BusinessFlight

PremiumFlight

Demo

Write the PremiumFlight class with dummy methods

Write the tests that are checking the new functionality

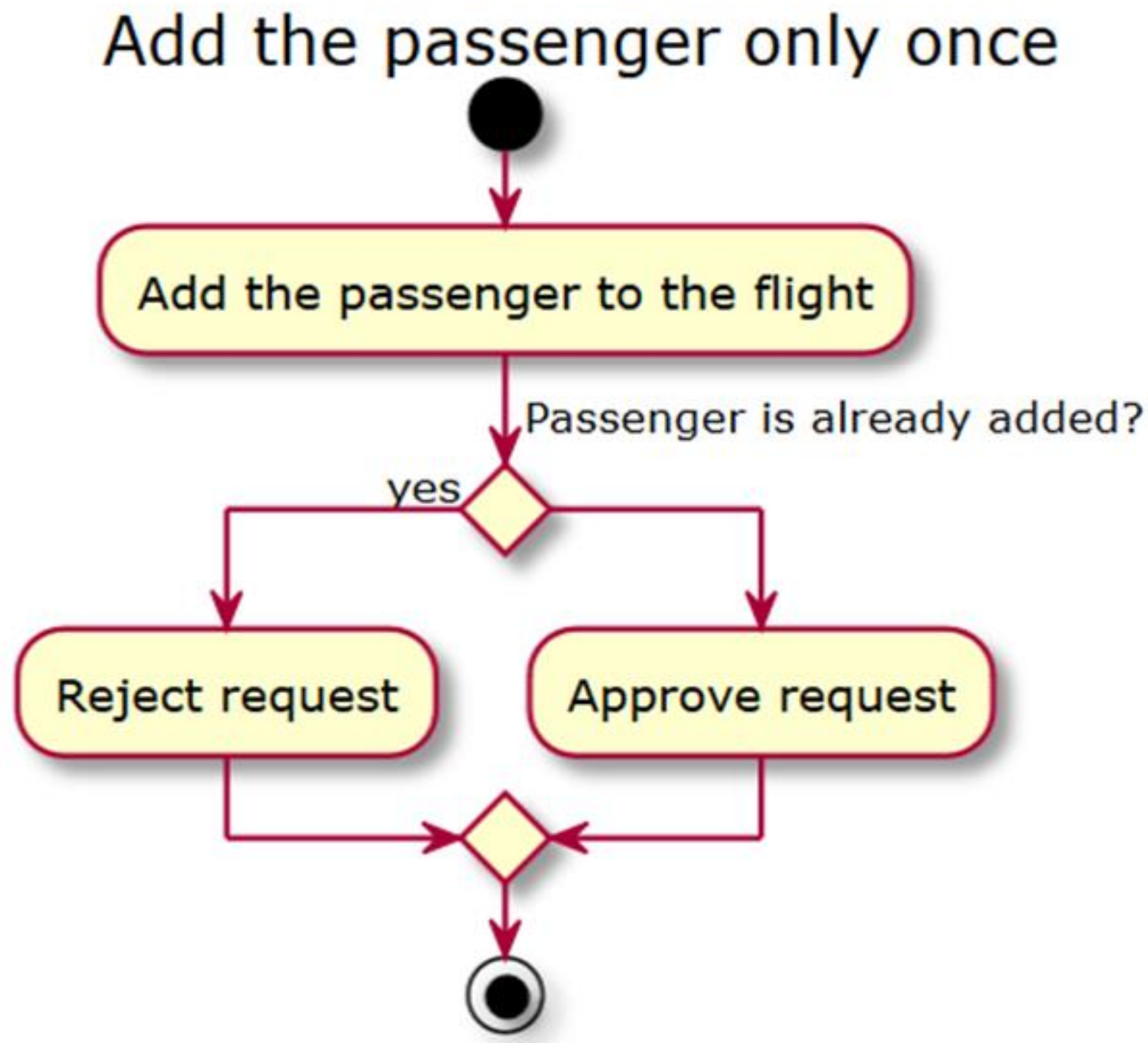
Run the tests - to be red

Effectively add the code that implements the new functionality

Run the tests - to be green

Adding A Passenger Only Once

Adding Passenger Only Once


Add the passenger only once
Add the passenger to the flight
Passenger is already added?
yes
Reject request
Approve request

Demo

Write the tests checking the restriction

Run the tests - to be red

Effectively add the code that implements the restriction

Run the tests - to be green