TDD by Integrating JUnit 5 with Mockito

Cătălin Tudose
PHD IN COMPUTER SCIENCE, JAVA AND WEB TECHNOLOGIES EXPERT
https://www.linkedin.com/in/catalin-tudose-847667a1

Overview

The need for mock objects

Introducing Mockito through the JUnit 5 extension model

Uses cases for mock objects

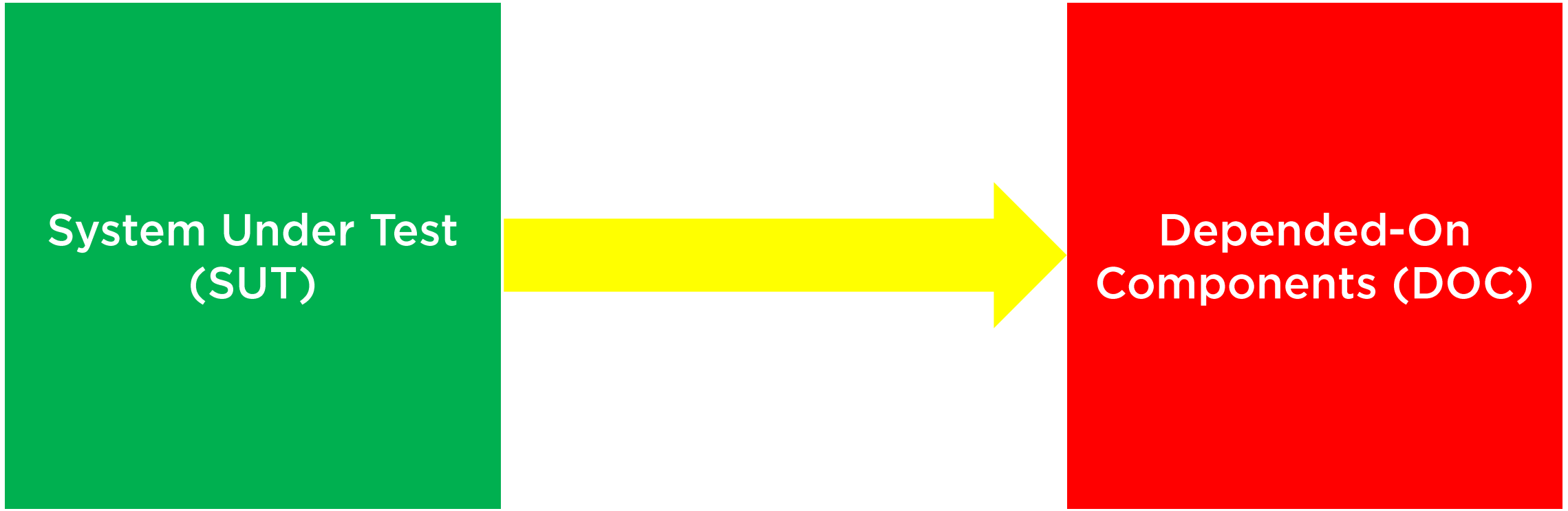
Mock external devices

Mock external database

Transpose scenarios in JUnit 5

Fixing tests implementing business logic

SUT and DOC



Mock Objects

Simulated objects that mimic the behavior of real objects in a controlled way

Created to test the behavior of some other object

Simulate the behavior of complex, real objects

JUnit 5 Extensions

Extend the behavior of test classes or methods

Class implementing one or more interfaces - the JUnit 5 extension points.

Injecting dependencies into the instance.

Use Cases for Mock Objects

Object supplies non-deterministic results

States difficult to create or reproduce

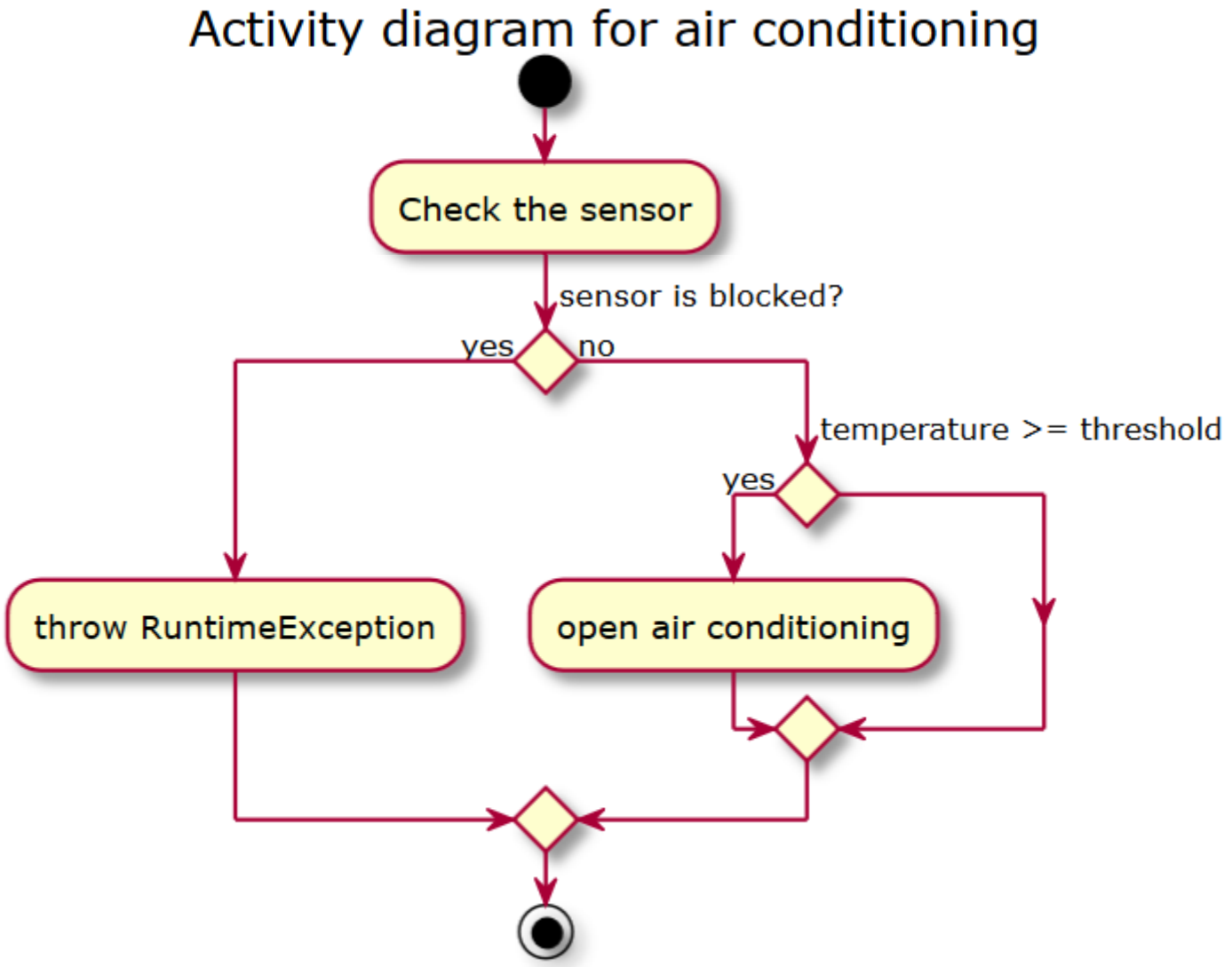
Slow

Does not exist or may change behavior

Include information and methods only for testing purposes

Implement the Air Conditioning Functionality

Implement the Air Conditioning Functionality



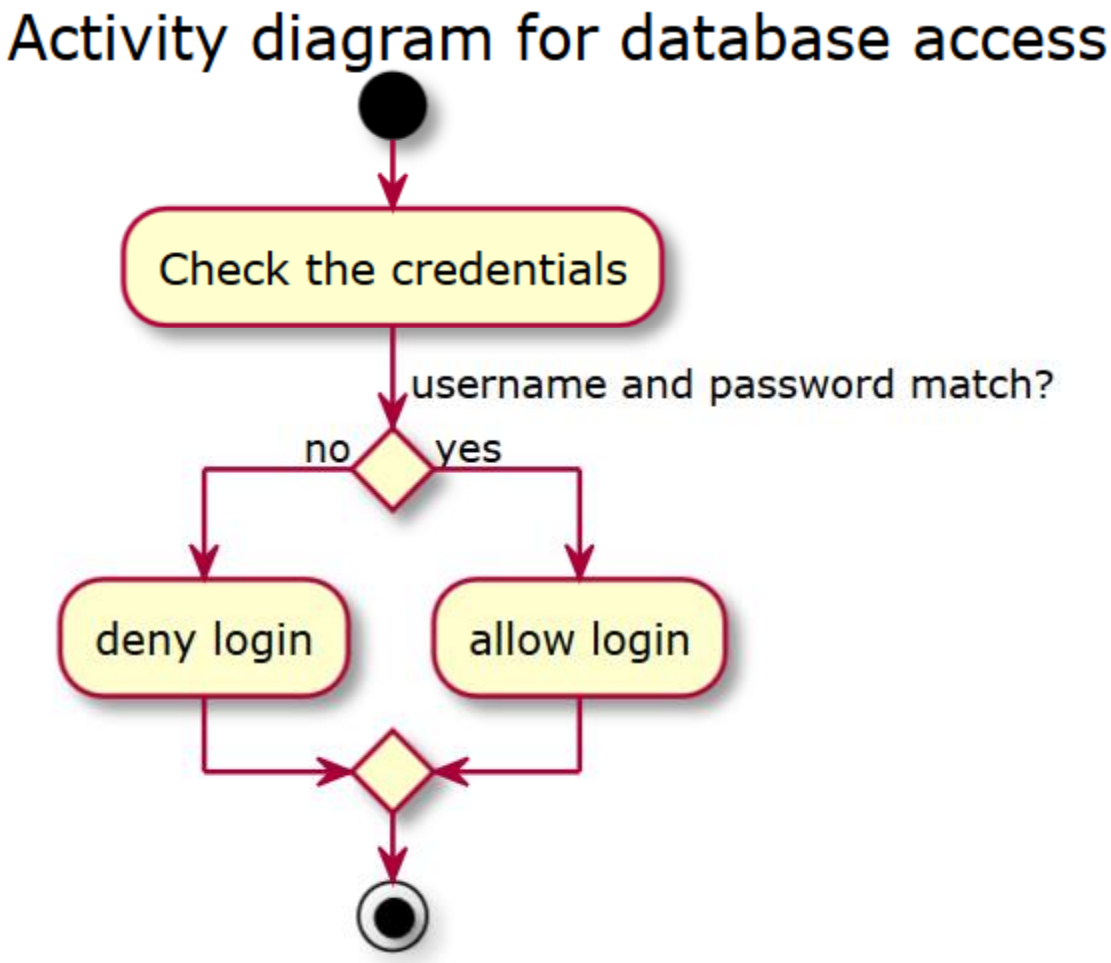
Relationship Between Components

Demo

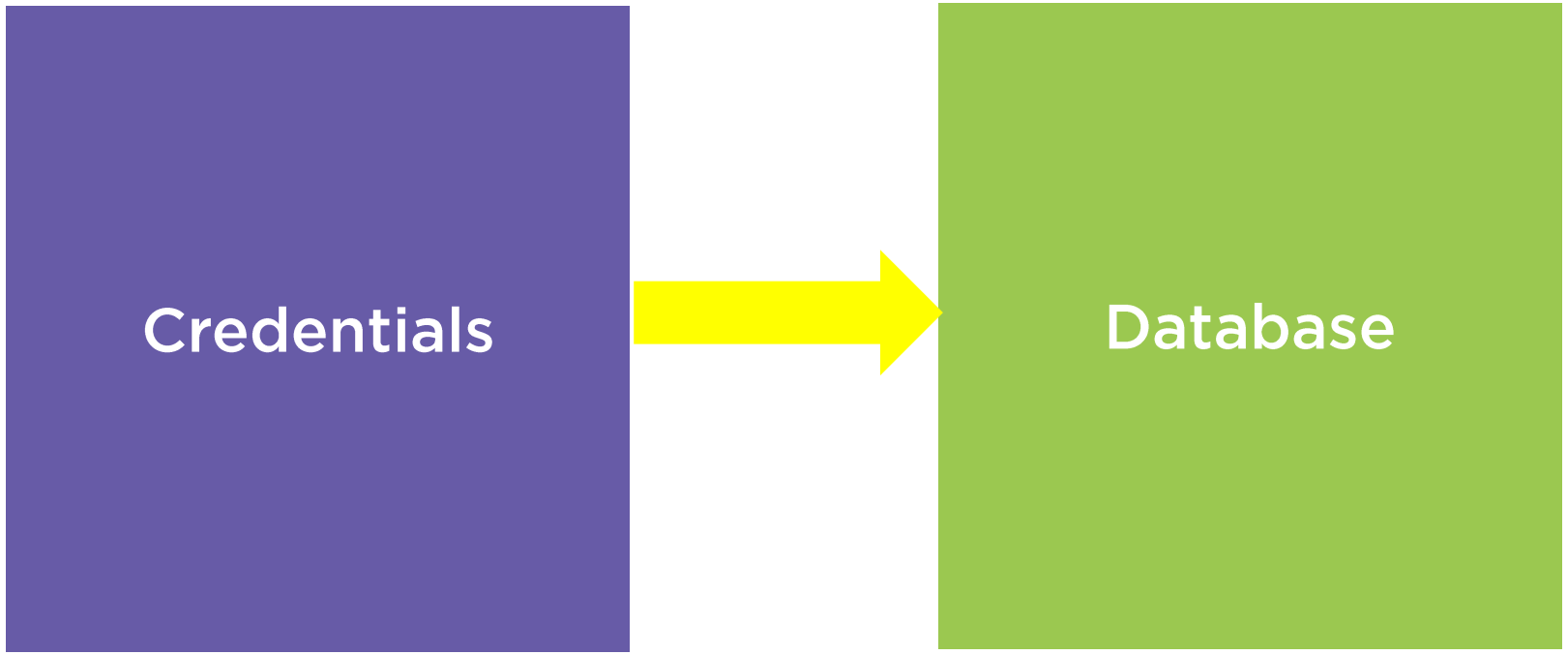
- Use mock objects for tests
- Use JUnit 5 extensions
- Write tests to check the air conditioning simulating the functionality of devices
- Implement the expected functionality
- Run the tests

Implement the Database Access Functionality

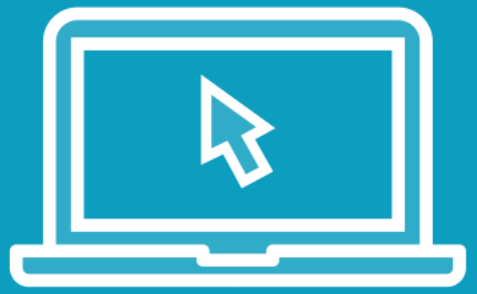
Implement the Database Access Functionality



Relationship Between Components



Demo



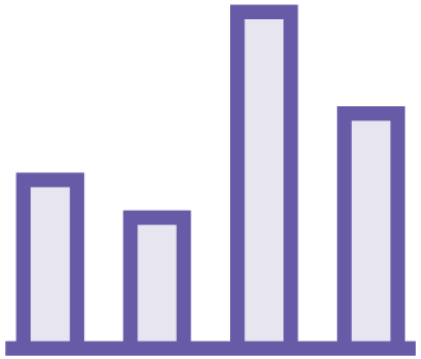
- Use mock objects and JUnit 5 extensions
- Write tests to check the database access with different credentials
- Implement the expected functionality
- Run the tests

Implement the Statistics Functionality

Database Query and Statistics

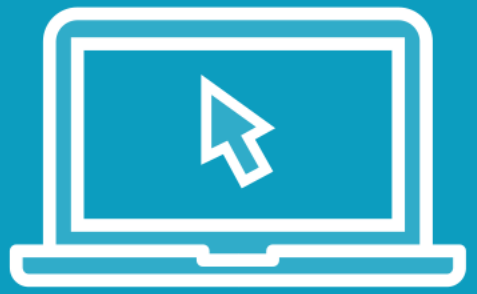
Query Database

Calculate



Statistics

Demo



Use mock objects to simulate the data retrieved from the database

Write tests to check the statistics

Implement the expected functionality

Run the tests

Summary

- Moved an existing program to TDD approach
- Build programs TDD style with JUnit 5
- Analysis of the new features
- Transpose scenarios in JUnit 5
- Implement the business logic driven by the tests
- Use mock objects to simulate the behavior of devices and database