TDD approach

What if the code already exists...

Anastasia Vasilyeva
anavasilyeva@gmail.com
15-09-2021

Problem

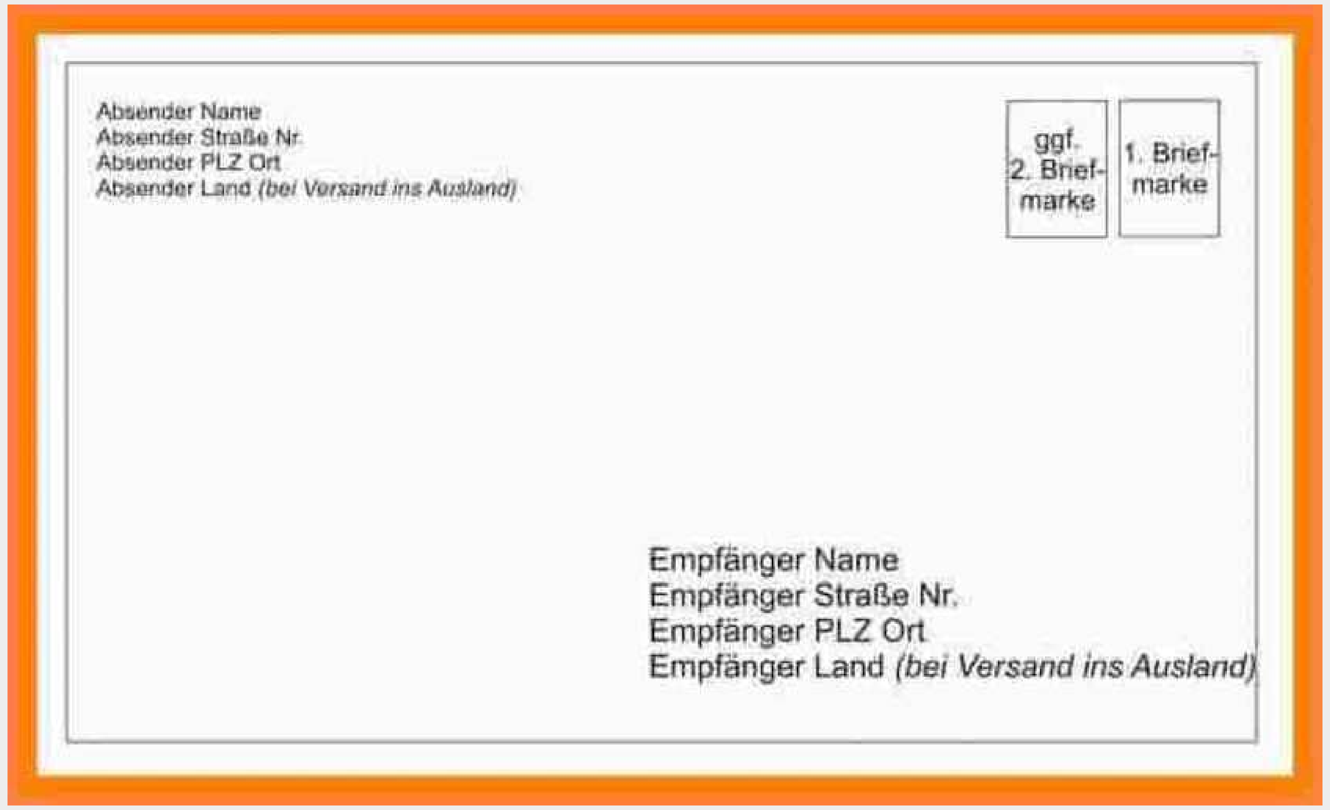
- TDD is great if you want to start to devepole a new feature.

- In real business cases, we are confronted with the fact that the code already exists. The functionality has already been implemented. And the current task is to make some changes or add new functionality.

- How do we follow the principles of TDD in this case?

Business Case

- We need to change an existing global logic - mapping the data of letter recipients and changing the output on the document.
- More than 30 components use this logic.
- There are no direct tests for global logic. There is only a set of indirect tests in dependent components - integration tests.
- The test coverage is unknown.



Absender Name
Absender Straße Nr.
Absender PLZ Ort
Absender Land *(bei Versand ins Ausland)*

ggf. 2. Brief-marke

1. Brief-marke

Empfänger Name
Empfänger Straße Nr.
Empfänger PLZ Ort
Empfänger Land *(bei Versand ins Ausland)*

Expectations

Business stakeholder expectations:

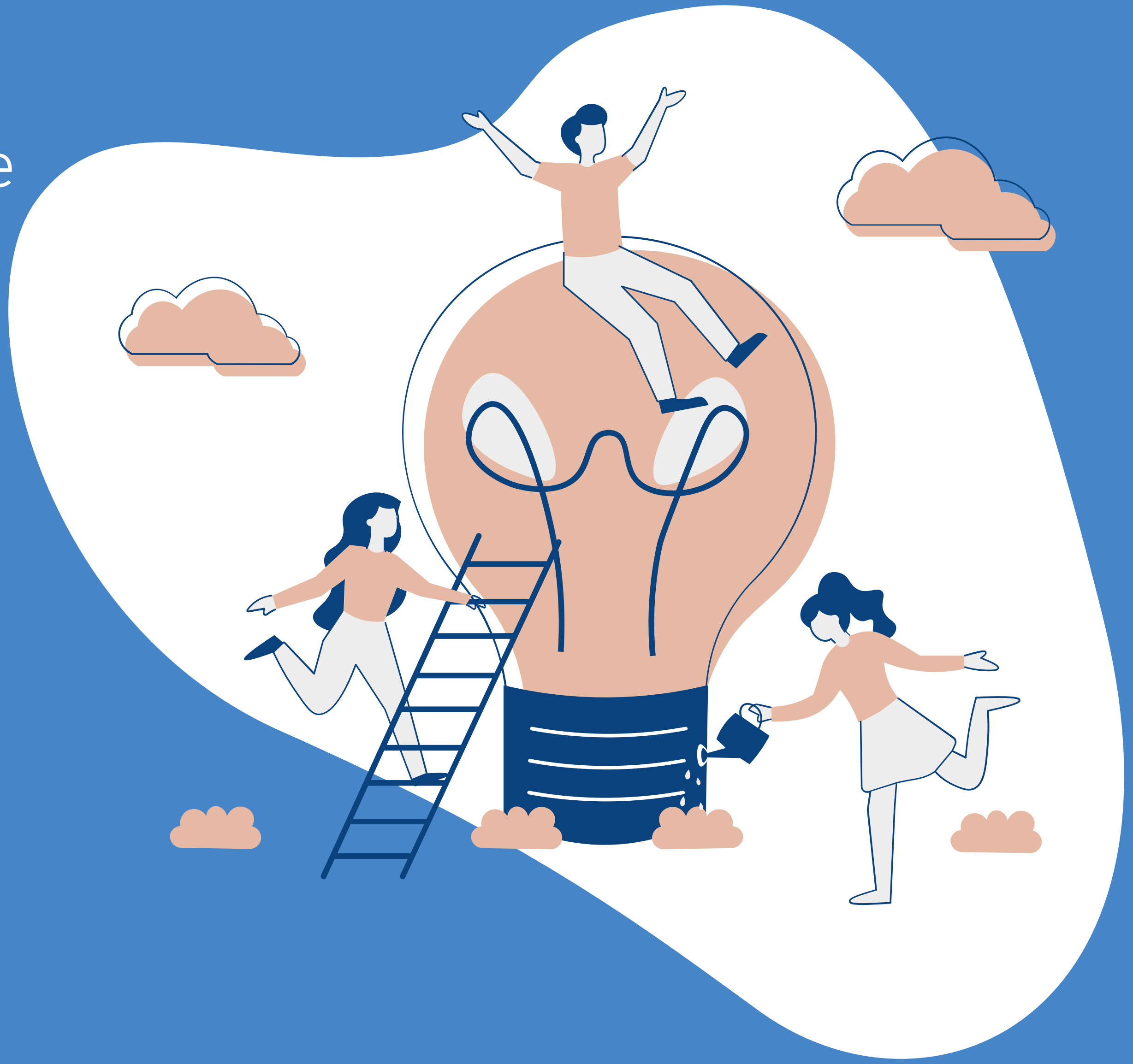
- to add a new feature

DEV team expectations

- understand how existing code works
- add good test coverage
- refactor old code (if needed)
- add new functionality

Step 1 - Freeze the existing logic in a test

As a first step, we decided to write a test for the existing logic. We didn't know exactly how the code worked. Based on TDD principles, we wrote a test for business logic (not the implementation).
In our case the Test must be green at this step (it tests the old logic).



Step 2 - Write new Test for a feature/change

At this step we write a new test for a new change. It must be red.

Step 3 - Implement a change

- Implement a new feature in the existing code.
- Check the Test from step 2 - it must be green.
- If we changed the existing logic, that the test from step 1 must be red.
- Delete the test 1 if not needed.

Step 4 - Integration Test

As we have changed the global logic on which many components depend, we need to check at this point how the components have reacted to the change. Existing integration tests must be red.

Step 5 - Update integration tests

Update integration tests with a new logic. Check if the logic works properly for depedent components.

Step 5 - Refactor and control the tests

Win 1

We understood how the logic of the global component works and were able to modify it.

Win 2

Thanks to the new tests, we are confident that we didn't break something else.

Win 3

We optimised the code and broke one big function into smaller understandable functions

Win 4

Development in Mob - better knowledge transfer.

Questions? Discussion...

I'm excited about the next part of the course!

Thanks!