Behavior Driven Development

Isaac Aiman Salas
Javier Ramos Fernández

Index

- Testing in programming
- What is BDD?
- Differences between BDD and TDD
- Principles of BDD
- Robot class
- AssertJ

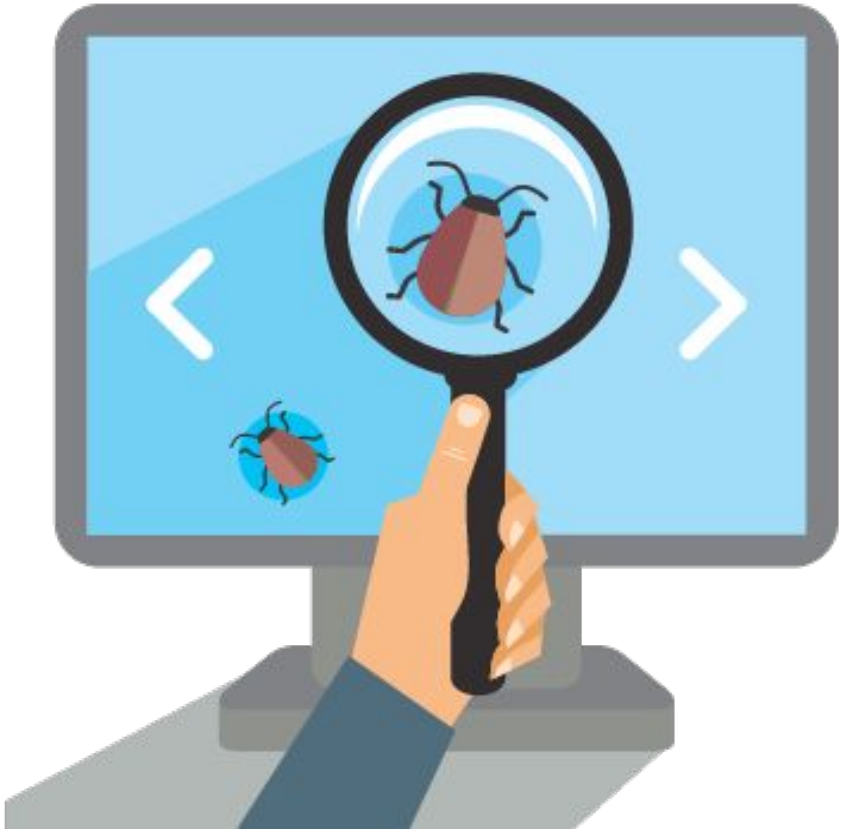
1. Testing in Programming

“If debugging is the process of removing bugs, then programming must be the process of putting them in.”

(Edsger Dijkstra)

What is testing?

Is the process of validating and verifying that a software program or application is working as expected.



Advantages

- Improves your designs.
- Reduces unnecessary code.
- Is motivating.
- Allows more freedom.

2. What is BDD?

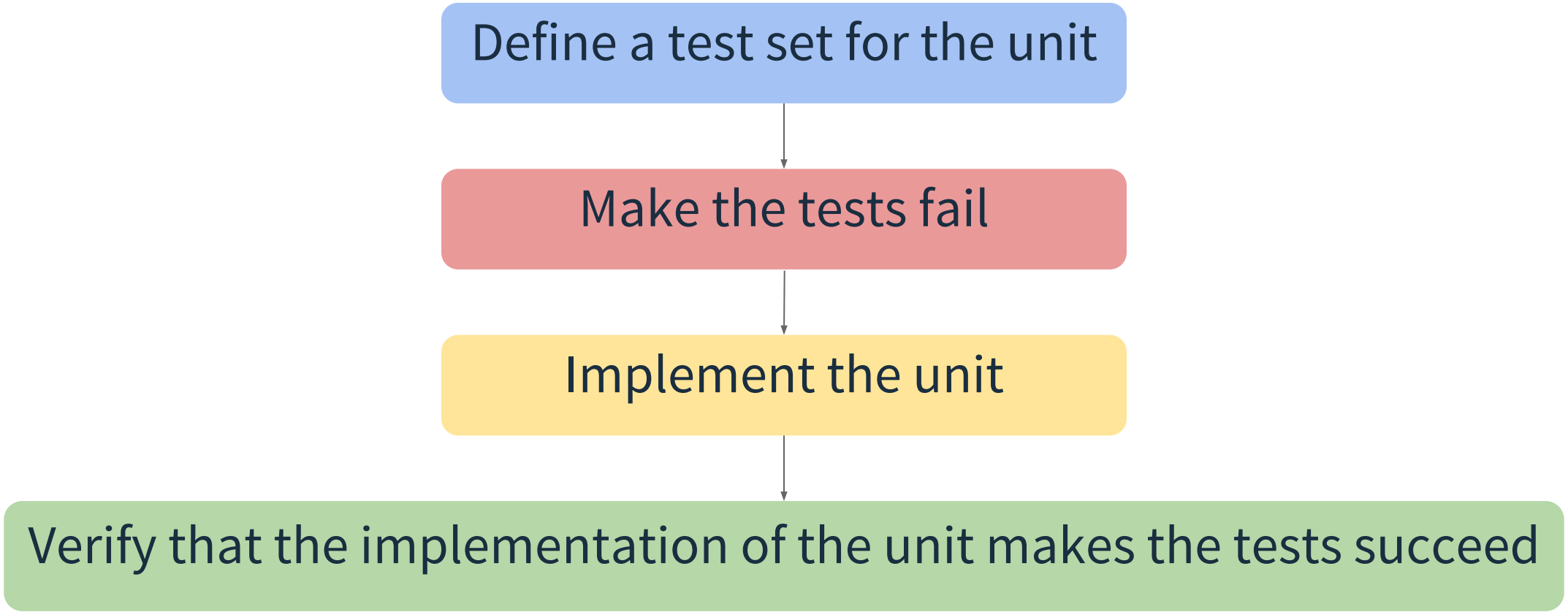
Introduction

- Software development process that emerged from test-driven-development (TDD).
- Should be focused on the business behaviors your code is implementing.
- Makes use of a simple domain-specific scripting language.
- Provides management team with shared tools and a shared process to collaborate on software development.

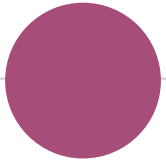
BDD focuses on:

- Where to start in the process.
- What to test and what not to test.
- How much to test in one go.
- How to understand why a test fails.

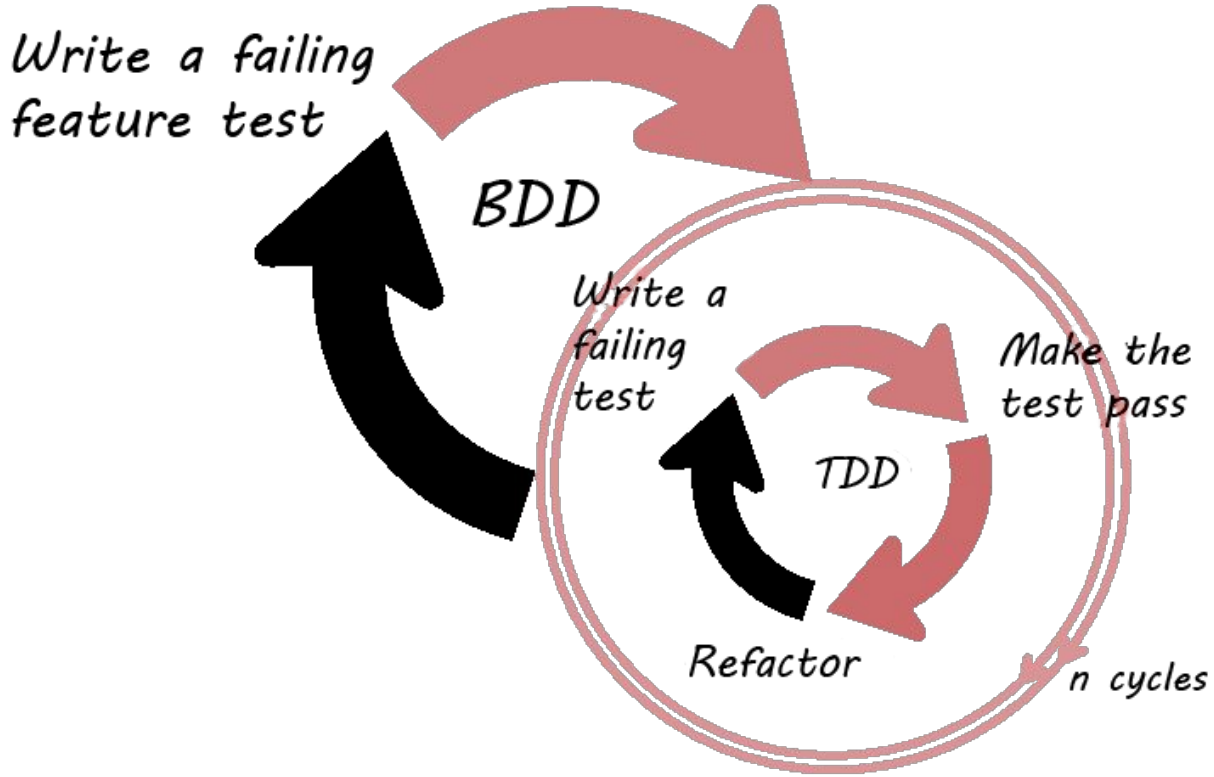
Principles of BDD



3. Differences between BDD and TDD



BDD vs TDD (I)


Write a failing feature test
BDD
Write a failing test
Make the test pass
TDD
Refactor
n cycles

BDD vs TDD (II)

Behaviour Driven Development	Test Driven Development
Teams get feedback from the product owner	Coder gets feedback from code
Thinks about what the scenario is	Thinks about how the code is implemented
High level-software requirements	Low-level technical details
Team scope	Programmer scope

Example

Imagine we want to build a shelf. With a **test driven development**, we should test each and every single hole in the wall in order to check that the shelf is going to be straight.

We set tests for each hole in the wall:

Example

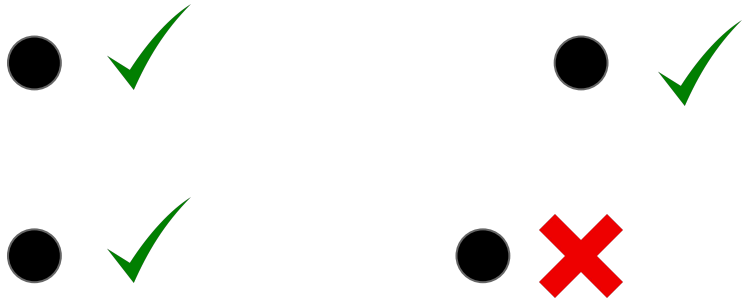
Imagine we want to build a shelf. With a **test driven development**, we should test each and every single hole in the wall in order to check that the shelf is going to be straight.

We set tests for each hole in the wall:

Example

Imagine we want to build a shelf. With a **test driven development**, we should test each and every single hole in the wall in order to check that the shelf is going to be straight.

We set tests for each hole in the wall:



Example

Imagine we want to build a shelf. With a **test driven development**, we should test each and every single hole in the wall in order to check that the shelf is going to be straight.

We set tests for each hole in the wall:

Example

Now we follow a **behavior driven development**. The behavior that we expect from a shelf is that it is straight enough to keep things on its surface without falling.

Now we only have a test that checks that the shelf is behavioring as it should:

Example

Now we follow a **behavior driven development**. The behavior that we expect from a shelf is that it is straight enough to keep things on its surface without falling.

Now we only have a test that checks that the shelf is behavioring as it should:

Is the shelf straight?

Example

Now we follow a **behavior driven development**. The behavior that we expect from a shelf is that it is straight enough to keep things on its surface without falling.

Now we only have a test that checks that the shelf is behavioring as it should:

Is the shelf straight?

Example

Now we follow a **behavior driven development**. The behavior that we expect from a shelf is that it is straight enough to keep things on its surface without falling.

Now we only have a test that checks that the shelf is behavioring as it should:

Is the shelf straight?

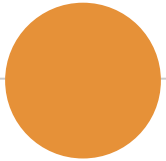
Example

Now we follow a **behavior driven development**. The behavior that we expect from a shelf is that it is straight enough to keep things on its surface without falling.

Now we only have a test that checks that the shelf is behavioring as it should:

Is the shelf straight?

4. Robot class



Test automation

Use of special software to control the execution of tests and the comparison of actual outcomes with predicted outcomes using an automation tool.

Benefits:

- Increases speed of test execution.
- Vastly increases Test Coverage.
- Does not require human intervention.
- Saves time and money.
- Critical for continuous delivery and continuous testing.
- Perform additional testing that would be difficult to do manually.

Automate GUI tests for Swing applications

- **User Interface Testing:** technique used to identify the presence of defects is a product/software under test by using Graphical user interface.
- **We need to access Swing components.**
- **Three ways to access Swing components:**
  - getXxx() methods.
  - Test code invokes events on a screen (Robot Class).
  - Test code traverses the component tree and finds a component of a specific signature.

What is the Robot class?

- **Class used to generate native system input events for the purposes of:**
  - Test automation.
  - Self-running demos.
  - Other applications where control of mouse and keyboard are needed.

- **Three main functionalities:**
  - Mouse control
  - Keyboard control
  - Screen capture

Robot class methods

- `mouseMove(int x, int y):` moves the mouse pointer to a set of specified absolute screen coordinates given in pixels.
- `mousePress(int buttons):` presses one or more mouse buttons.
- `mouseRelease(int buttons):` releases one of the buttons pointed by the mouse.
- `keyPress(int keycode):` presses a specified key on the keyboard.
- `keyPress(int keycode):` Releases specified key on the keyboard.

See example

Example 1 Robot Class

CALCULATOR

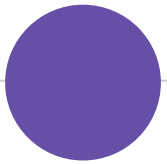
Example 2 Robot Class

BALL

Example 3 Robot Class

TEXT EDITOR

5. AssertJ



What is AssertJ?

- AssertJ provides a rich and intuitive set of strongly-typed assertions to use for unit testing. When writing unit tests, we should have at our disposal assertions specific to the type of the objects we are checking.

- For example, if you're checking the value of a String, you need to use String-specific assertions.

- Although you could work with the `Robot` directly, the `Robot` class is too low-level and requires considerably more code than the fixtures.

Modules

- **Core module:** for JDK types (String, Iterable, Stream, Path, File, Map...).
- **Guava module:** for Guava types (Multimap, Optional...).
- **Joda Time module:** for Joda Time types (DateTime, LocalDateTime) .
- **Neo4J module:** for Neo4J types (Path, Node, Relationship...).
- **DB module :** for relational database types (Table, Row, Column...)
- **Swing module:** for functional testing of Swing user interfaces.

We are going to focus on swing module.

AssertJ swing

- Provides specific methods to simulate user interaction with a GUI component.
- Provides assertion methods that verify the state of such a GUI component.
- Provides one method per Swing component. Each method has the same name as the Swing component they can handle ending with Fixture. For example, a `JButtonFixture` knows how to simulate user interaction and verify the state of a JButton.
- Package needed: `org.assertj.swing.fixture`

Basic Input Simulation (I)

AssertJ Swing uses an AWT Robot to generate user input.

- **Pressing and releasing one or more keys:**
  - `pressAndReleaseKeys(int...)`: can simulate a user pressing and releasing one or more keys.
  - `pressKey(int)`: simulates a user pressing a key.
  - `releaseKey(int)`: simulates a user releasing a key.

Basic Input Simulation (II)

- **Typing text:**
  - `enterText(String)`: it is a method available for text component fixtures that allows to enter text.
- **Simulating clicks:**
  - `click()`: simulates a user clicking a component once, using the left button.
  - `rightClick()`: simulates a user clicking a component once, using the right button.
  - `doubleClick()`: simulates a user double clicking a component once, using the left button.

Examples

Example 1 - CopyText

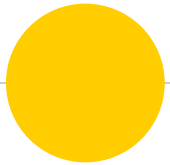
Example 2 - Currency Converter & Copy2Texts

Example 3 - Login Example

Example 4 - Bola móvil

Example 5 - Background color

All examples can be found at:
https://github.com/alu0100841565/CodigoPresentacionBDD



Any questions?

Thanks!

References

Test automation.

https://en.wikipedia.org/wiki/Test_automation

Robot class.

http://www.developer.com/java/other/article.php/2241561/An-Automated-Test-Program-using-the-Java-Robot-Class.htm
http://alvinalexander.com/java/java-robot-class-example-mouse-keystroke
https://www.slideshare.net/oxus20/java-virtual-keyboard-using-robot-toolkit-and-jtogglebutton-classes-39358530

References

AssertJ

https://joel-costigliola.github.io/assertj/

https://joel-costigliola.github.io/assertj/swing/api/index.html