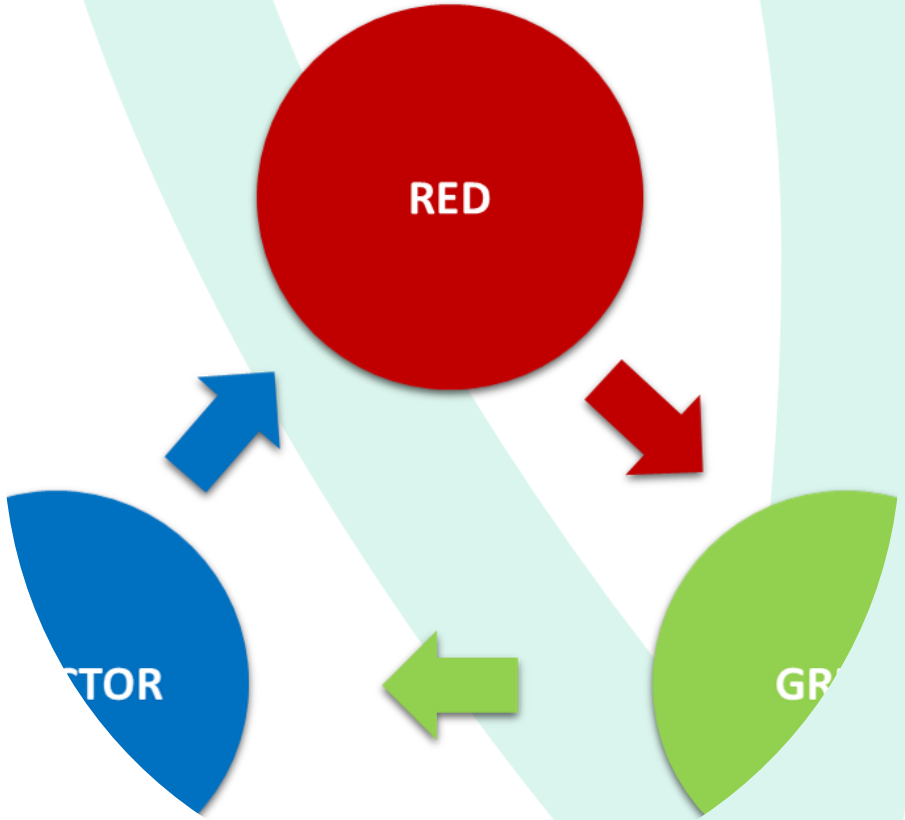
nagarro



TDD
Test-driven development

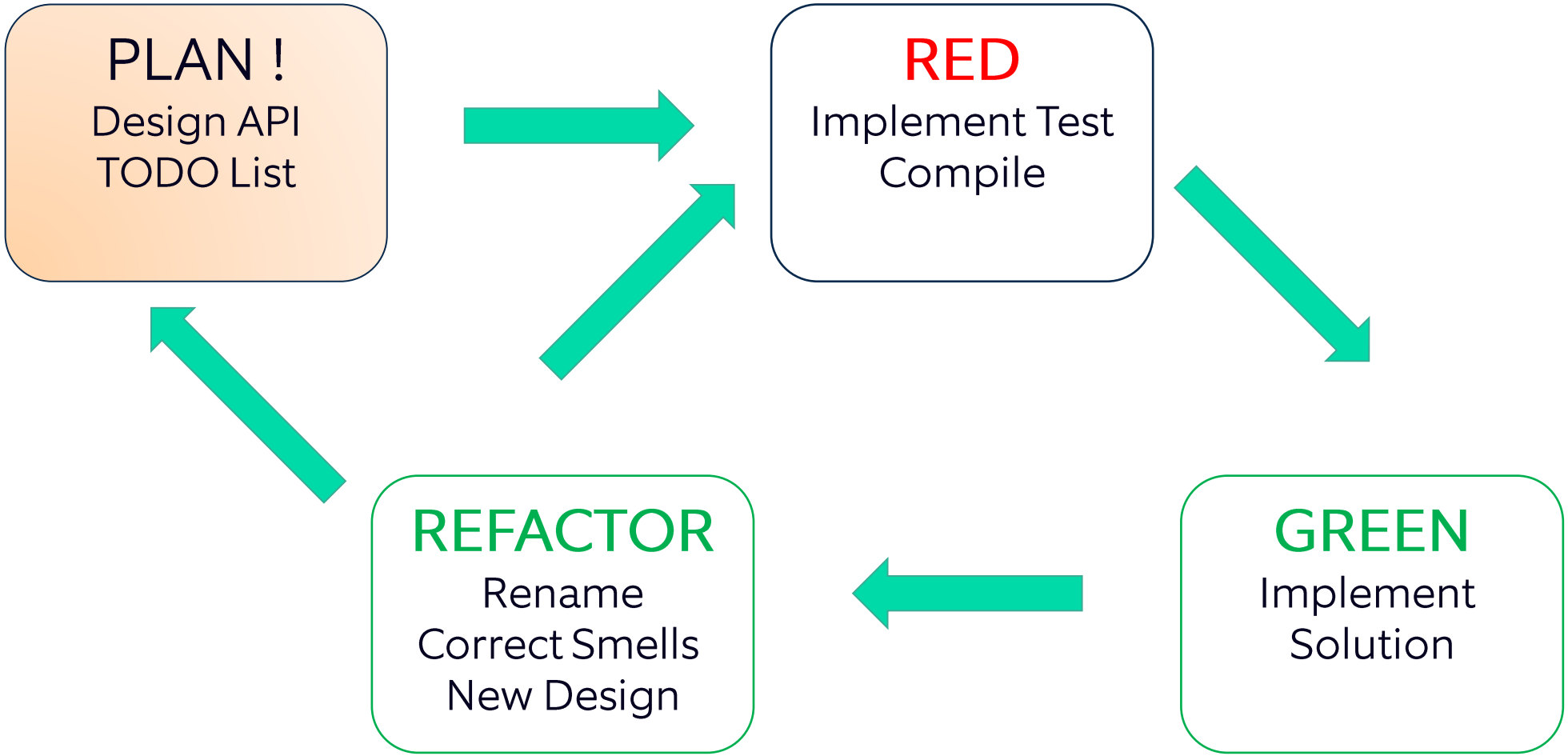
Exercise – What Do You Know About TDD?

Think about TDD and answer the following question:

- Which practices do I know?
- How and when did I use it?
- Which steps do I follow in TDD?
- Which questions come into my mind?
- ...

Afterwards discuss your findings with your neighbour and present it to the participants.

TDD Workflow



TDD Workflow

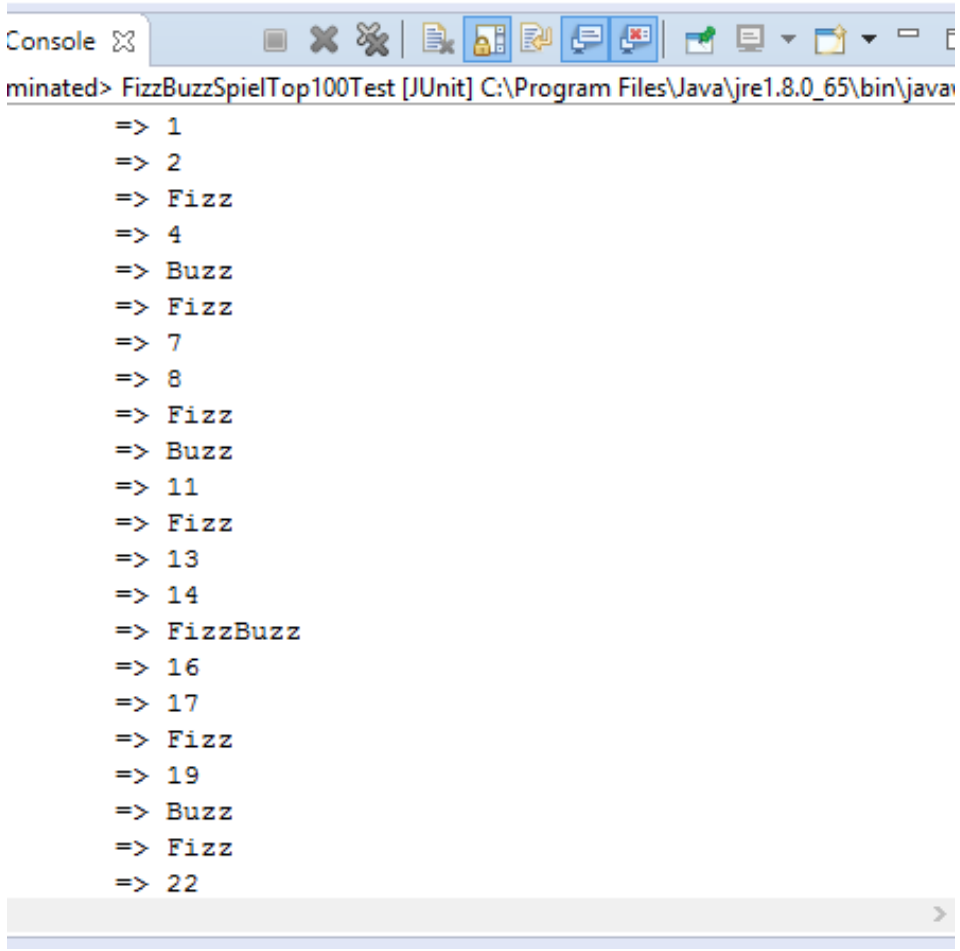
- **Plan**
  - Design and start
  - Input → Output
  - Find examples and create a list of testcases
  - Order testcases from simple to complex
- **Implement**
  - Red
  - Green
  - Refactor
- **Check** your TODO list
  - New examples, testcases and/or order

Baby Steps

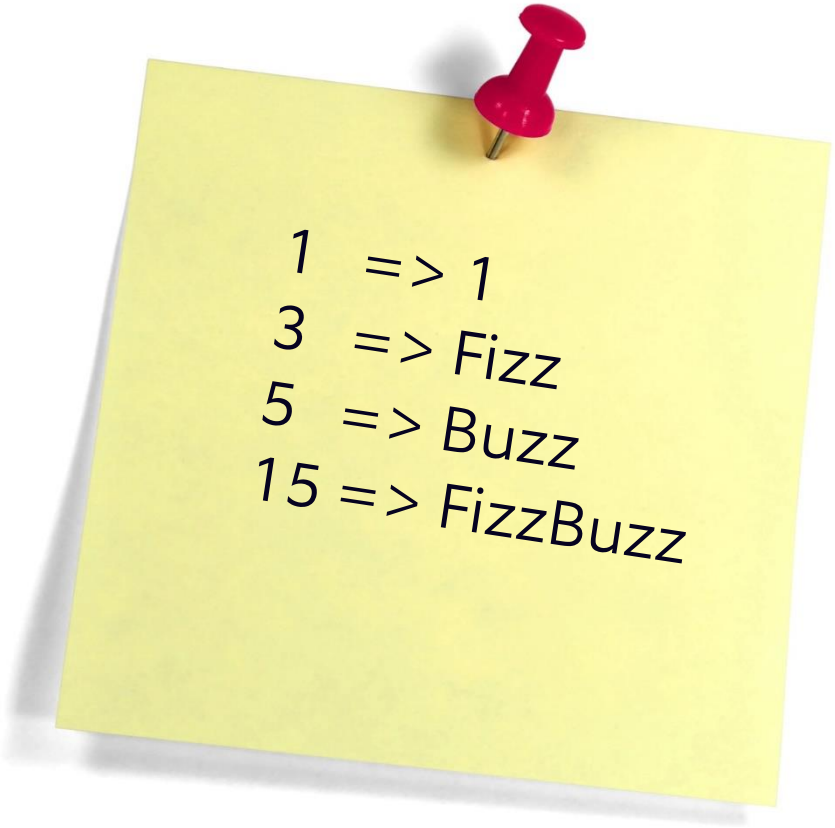
Let‘s implement Fizz-Buzz with TDD

- FizzBuzz is a game counting numbers starting at 1 - sounds simple, or?

- There are some additional rules:
  - If the number is divisible by 3 => „Fizz“
  - If the number divisible by 5 => „Buzz“
  - If the number divisible by 3 and 5 => „FizzBuzz“

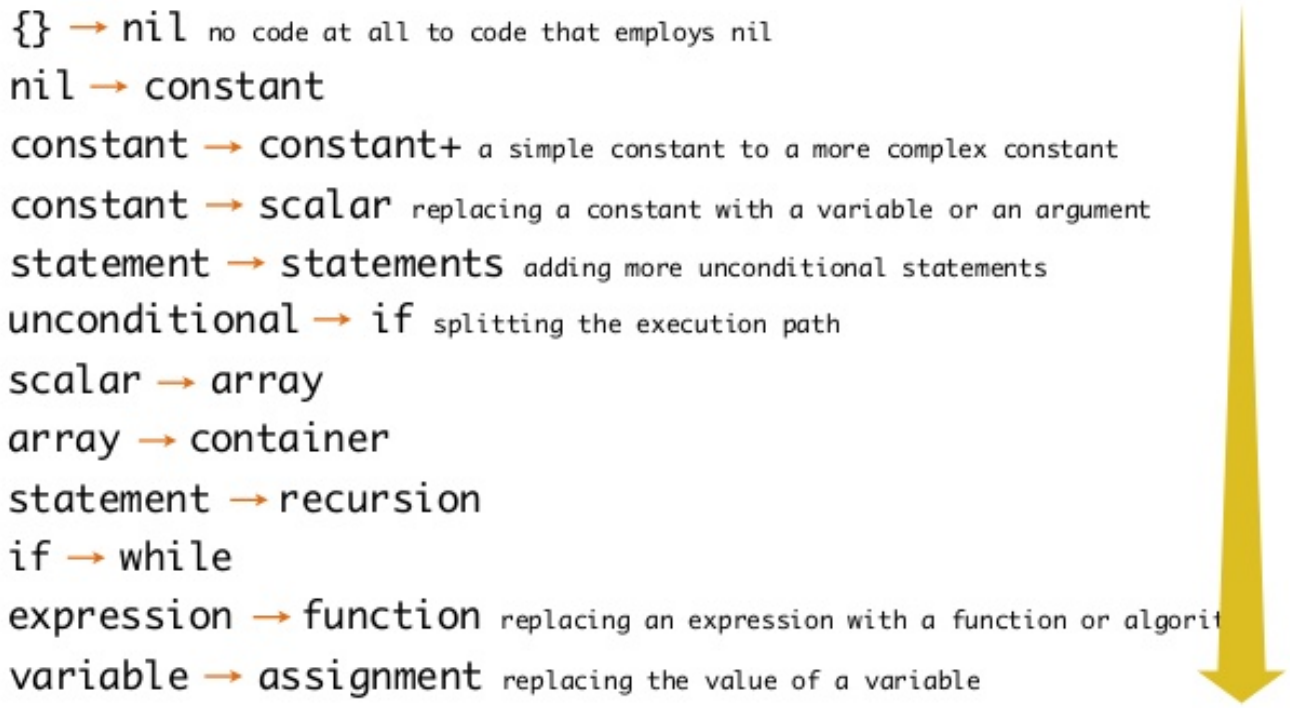


The first step– create a TODO-list and design your API



FizzBuzzGame
+Answer(int): String

The Transformation Priority Premise



There are likely others

https://blog.cleancoder.com/uncle-bob/2013/05/27/TheTransformationPriorityPremise.html

nagarro

Hands on: TDD

String Calcuator Part 1

User Stories

1. Create a simple String calculator with a method:
   int Add(string numbers)
2. Provide two default delimiters for separating the numbers: , and \n
3. Support additional delimiters in the following format
   "//[delimiter]\n[numbers...]"

Examples

„1,2,8“ => 11

„2\n4,5“ => 11

//;\n5;9,3“ => 17

Note: create a new branch for the hands-on sessions

Credits to Roy Osherove http://osherove.com

String Calcuator Part 2

User Stories

1. Calling Add() with a negative number will throw an exception with the message “negative numbers are not allowed”. The exception should contain the negative number that was passed. If there are multiple negative numbers, list all of them in the exception message
2. Numbers bigger than 1000 should be ignored, so adding 2 + 1001 = 2
3. Delimiters can be of any length:
   “//[***]\n1***2***3” should return 6
4. Allow multiple delimiters like this:
   “//[delim1][delim2]\n”
   “//[*][%]\n1*2%3” should return 6.

Credits to Roy Osherove http://osherove.com

What can I achieve with TDD?

- Better understanding of your requirements
  - Avoiding assumptions
- Testable Design
  - Transparent dependencies
  - Small methods and classes
  - New and completely different solutions
- 100% Unit Test Coverage
  - Fast feedback
  - Secure Refactoring
  - No excuse to forget testing
- Less bugs during QA testing and in production

Which skills do you need to be successful?

- Identify the best fitting tests
- Experience in Software Design
  - Design Principles and Patterns
- Clean Code
  - Identify Code Smells
- Refactoring
  - Correct Code Smells
  - Adapt your code continuously
  - Knowledge about long running Refactoring Workflows

Summary

- Take care of RED-GREEN-REFACTOR rhythm
- Baby Steps – small steps
- Create a TODO list with examples at the beginning
- Don‘t forget the Refactoring steps in your loops
- Focus on keeping it simple