

IF.TECH - 2019

FUNDAMENTOS DE TESTES EM JAVA

Neil Prado
Samuel Cabral

VOÇÊ TESTA SUAS APLICAÇÕES?

POR QUÊ É NECESSÁRIO TESTAR?



**SIM,
VOCÊ TESTA**

Suponha que você tem este código:

```
public int subtract(int x, int y){
    return x - y;
}
```

Como você testaria o método acima?

PROBLEMAS CAUSADOS POR UM SOFTWARE MAL FEITO

1

FINANCEIRO

2

REPUTAÇÃO

3

TEMPO

4

VIDA



O QUE É TESTE?

Testes de software são um processo que faz parte do desenvolvimento que tem como objetivo identificar falhas para que sejam relatadas até que o produto atinja a qualidade desejada.

Tipos de Teste de Software

UNITÁRIO

Teste em um nível de componente ou classe. É o teste cujo objetivo é um “pedaço do código”.

INTEGRAÇÃO

Garante que um ou mais componentes combinados funcionam.

REGRESSÃO

Toda vez que algo for mudado, deve ser testada toda a aplicação novamente.

FUNCIONAL

Testar as funcionalidades, requerimentos, regras de negócio presentes na documentação.

Tipos de Teste de Software

INTERFACE

Verifica se a naveabilidade e os objetivos da tela funcionam como especificados e se atendem da melhor forma ao usuário.

PERFORMANCE

Verifica se o tempo de resposta é o desejado para o momento de utilização da aplicação

CARGA

Verifica o funcionamento da aplicação com a utilização de uma quantidade grande de usuários simultâneos.

ACEITAÇÃO

Testa se a solução será bem vista pelo usuário.

AUTOMATIZAÇÃO DE TESTES

O QUE DANADO É ISSO?

AUTOMATIZAÇÃO DE TESTES

O QUE É?

É o uso de software para controlar a execução de testes de software através da aplicação de estratégias e ferramentas, comparando os resultados esperados com os resultados reais

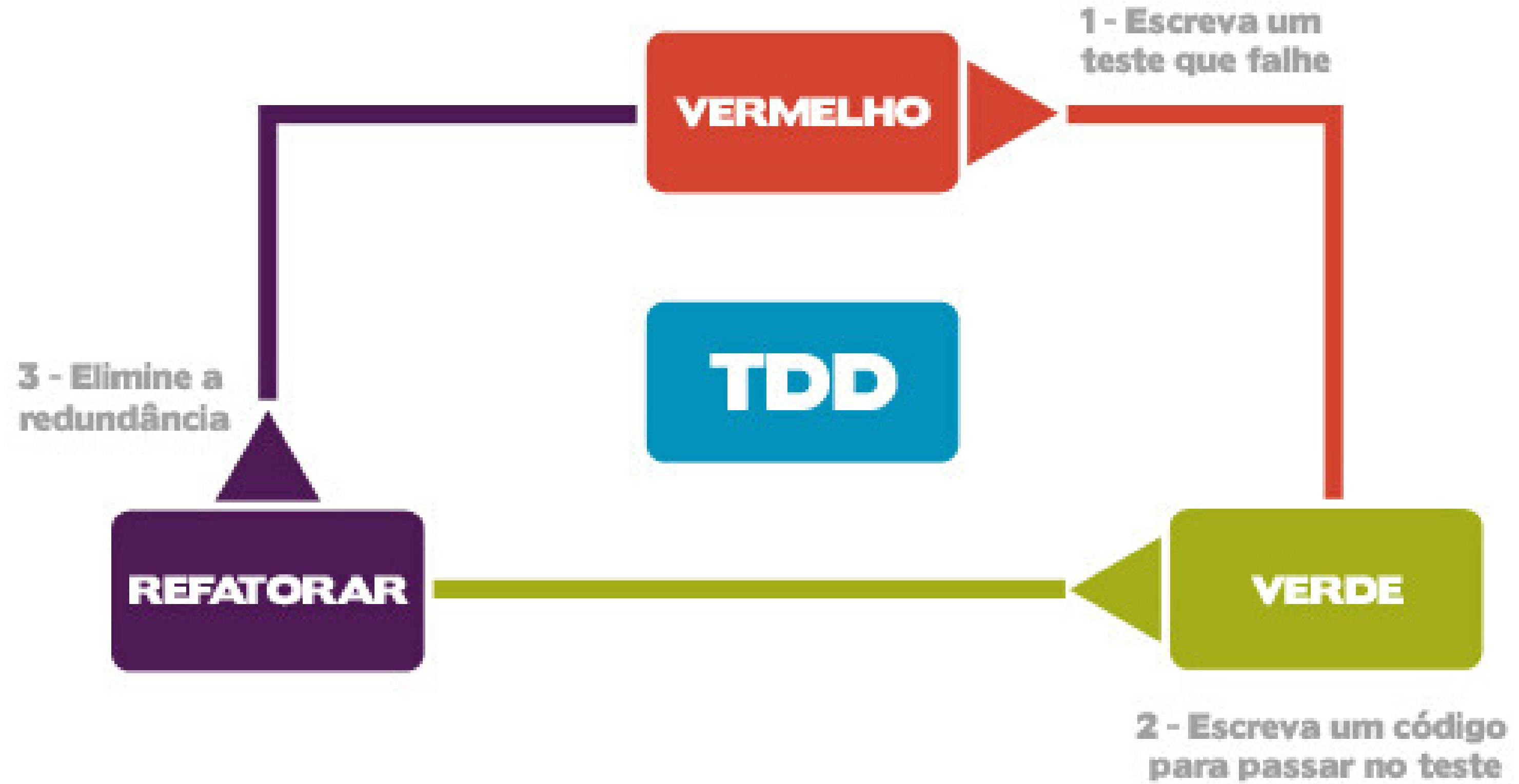
TDD

TEST DRIVEN DEVELOPMENT

O QUE É TDD?

Técnica de desenvolvimento de software que se relaciona com o conceito de verificação e validação e se baseia em um ciclo curto de repetições

Deve-se primeiro escrever os testes, antes de implementar o sistema. Os testes são utilizados para facilitar no entendimento do projeto, que servem para clarear a ideia em relação ao que se deseja em relação ao código.



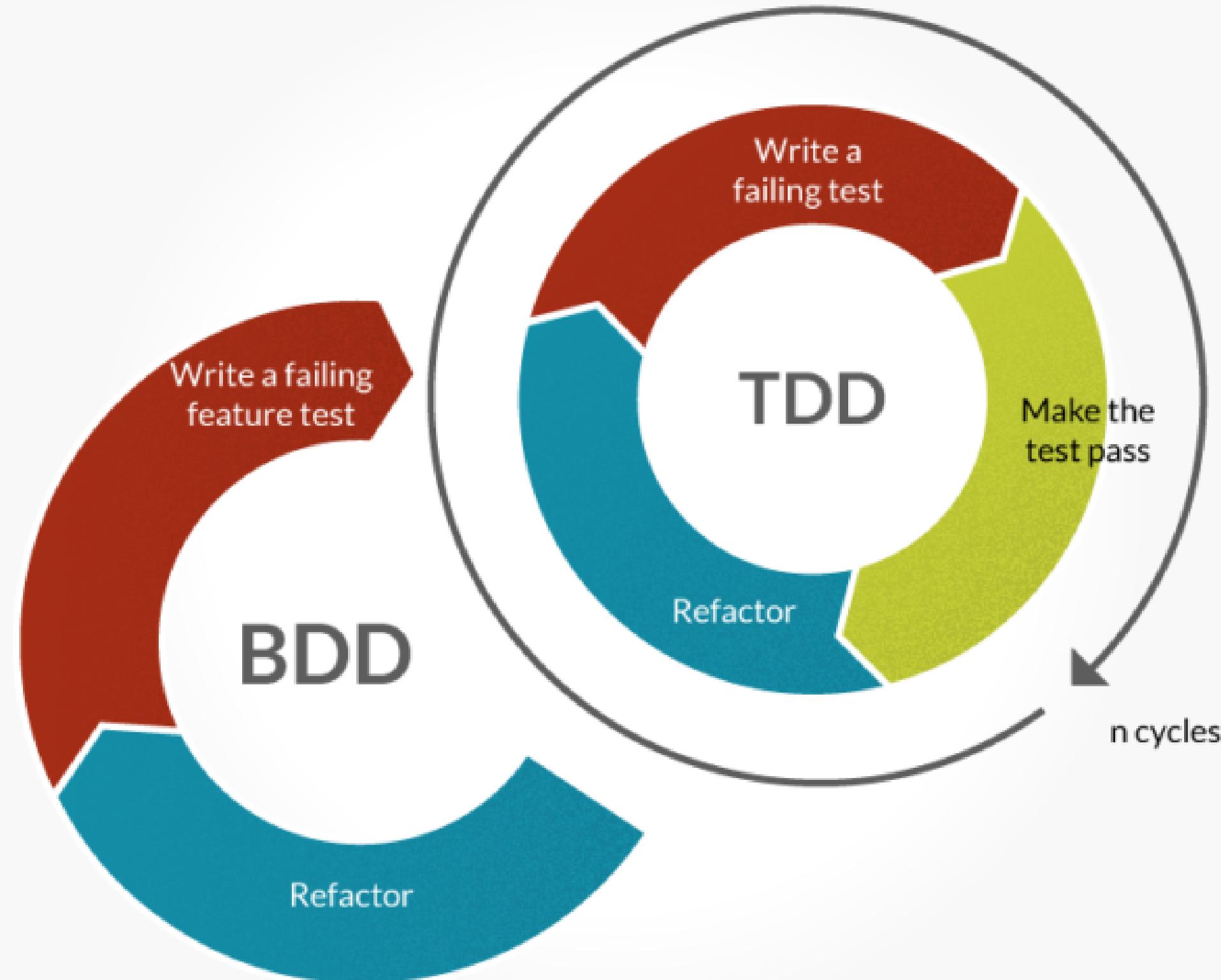
BDD

BEHAVIOR DRIVEN DEVELOPMENT

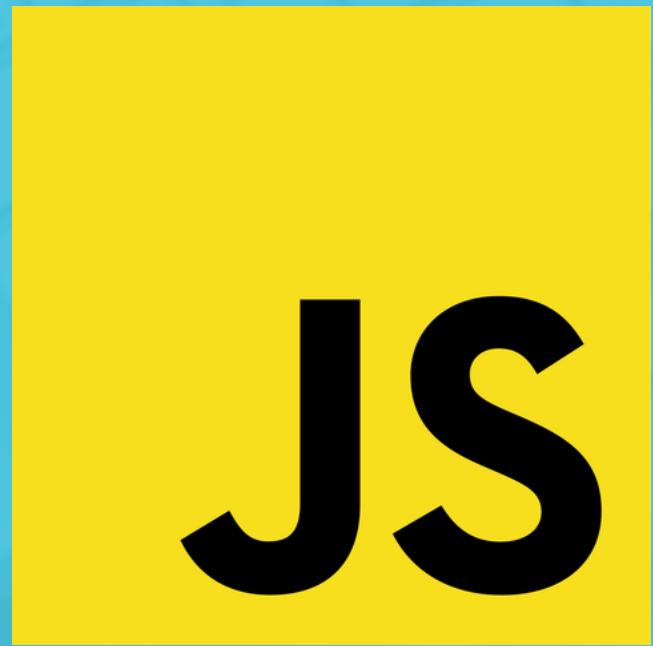
O QUE É BDD?

BDD serve para criar testes e integrar regras de negócios com a linguagem de programação, focando no comportamento do software. Além disso, ainda melhora a comunicação entre as equipes de desenvolvimento e testes, aumentando o compartilhamento de conhecimento entre elas.

É útil em projetos de software ágeis, que são construídos em várias iterações e estão sofrendo alterações ao longo do seu ciclo de vida. Quanto maior o projeto, mais difícil será a comunicação. Entretanto, BDD propõe uma forma eficaz de resolver estes problemas.



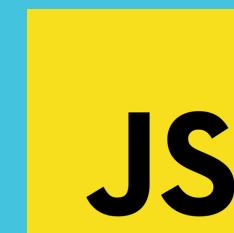
**BELEZA, COMO EU
USO O SOFTWARE
PRA TESTAR?**



Exemplos



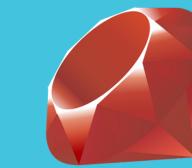
PHPSpec
Behat



Jest
Jasmine
Mocha
RhinoUnit



Vcrpy
Freezegun
Model_mommy



RSpec
Cabybara

E NÓS?



JUNIT

MOCKITO

REST-ASSURED

JUnit

ASSERT

JUNIT API

```
static void assertTrue(String message, boolean condition)
static void assertEquals(String message, Object expected, Object actual)
static void assertNull(String message, Object object)
```

Parâmetros em negrito são opcionais

SINTAXE BÁSICA DE UM TESTE UNITÁRIO

```
public class calculadoraTest(){  
    @Test  
    public void somaTest(){  
        int res = sum(2,3);  
        Assert.assertEquals(5, res);  
    }  
}
```

ANNOTATIONS

@Before

Faz com que o método com esta anotação rode antes de qualquer caso de teste

@After

Faz com que o método com esta anotação rode depois de qualquer caso de teste

@Test()

Determina se um método é de teste. É escrito logo acima do método esta anotação.

@Ignore

Ignora o método de teste, fazendo com que o método não seja executado na suite de testes.

mockito



MOCKS?

Mocks são utilizados para descrever um caso especial de objetos que imitam objetos reais para teste. Esses Mock Objects atualmente podem ser criados através de frameworks, como o Mockito, que facilitam bastante a sua criação.

REST-assured



REST-ASSURED

é um framework open-source que possui suporte para automatizar os métodos GET, POST, PUT, DELETE, HEAD, OPTIONS e TRACE de uma API e permitir fazer validações no header, body e schema da resposta.

SINTAXE BÁSICA DO REST-ASSURED

```
public class exampleTestRestAssured(){
    @Test
    public void testCriaUsuario() {
        Map<String, String> object = new HashMap<>();
        object.put("nome", "João");
        object.put("email", "joao@live.com");
        object.put("endereco", "Avenida Primeiro de Maio");

        RestAssured.given()
            .contentType("application/json")
            .body(myJson)
            .when()
            .post("/")
            .then()
            .statusCode(200)
            .body("message", containsString("Usuário criado com sucesso")));
    }
}
```

DOCUMENTAÇÃO OFICIAL

JUnit

<https://junit.org/junit4/>

mockito

[https://javadoc.io/doc/
org.mockito](https://javadoc.io/doc/org.mockito)



REST-assured

<http://rest-assured.io/>

ABOUT US

Contatos

**NEIL
PRADO**

github.com/neilprado
neil.prado@academico.ifpb.edu.br

**SAMUEL
CABRAL**

github.com/samuelcdcabral
samuel.cabral@academico.ifpb.edu.br

OBRIGADO!!!!

EPA JOVEM, AINDA NÃO ACABOU

E AGORA?

MÃOS À OBRA, JOVENS

NOSSO PROJETO

Sigam os passos do repositório abaixo:

<https://github.com/neilprado/iftech-FundamentosDeTestesJava>