

Introdução a teste unitário / Teste componentes

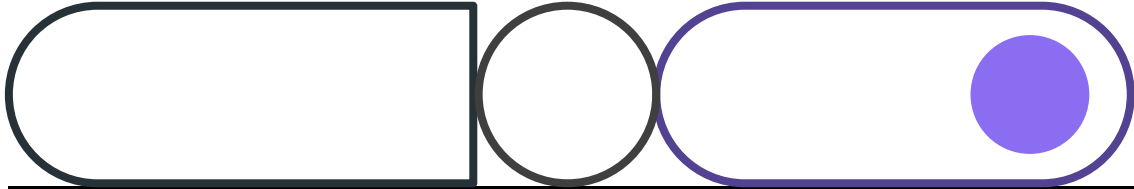
Índice

- 01** Generalidades
- 02** Frameworks

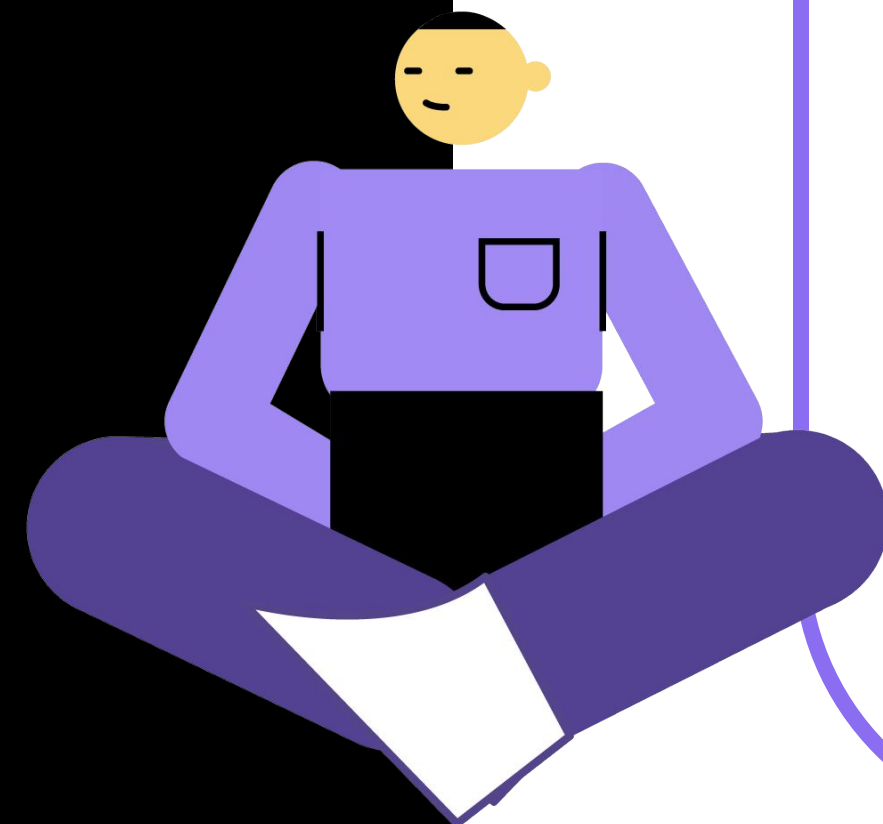


01

Generalidades



"O objetivo principal é isolar
cada unidade do sistema
para identificar, analisar e
corrigir defeitos".



Generalidades

O teste de componentes é geralmente feito isoladamente do resto do sistema, dependendo do modelo de ciclo de vida de desenvolvimento de software, que pode exigir objetos simulados, virtualização de serviço, stubs e drivers.

Este tipo de teste pode abranger:

01

Funcionalidade

Por exemplo, a precisão dos cálculos.

02

Recursos não funcionais

Por exemplo, a busca por vazamentos de memória.

03

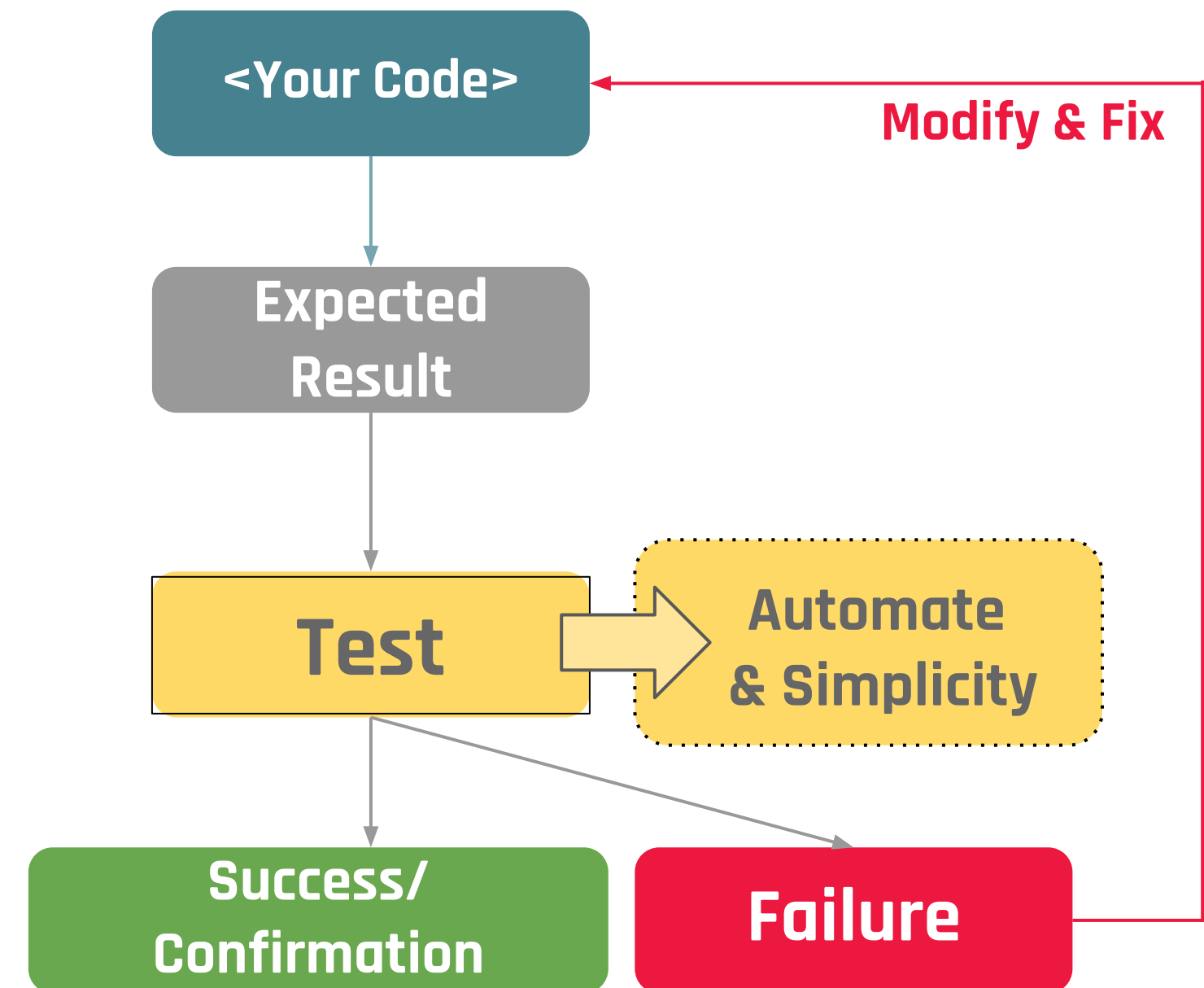
As propriedades estruturais

Por exemplo, testes de decisão.

Processo

Em geral, quando uma abordagem TDD não é seguida, o processo é o seguinte:

- O código do software é criado.
- Os resultados esperados são definidos.
- O teste é executado.
- Se o teste for aprovado, o resultado esperado é confirmado.
- Se o teste falhar, o código é modificado para resolver o defeito encontrado.



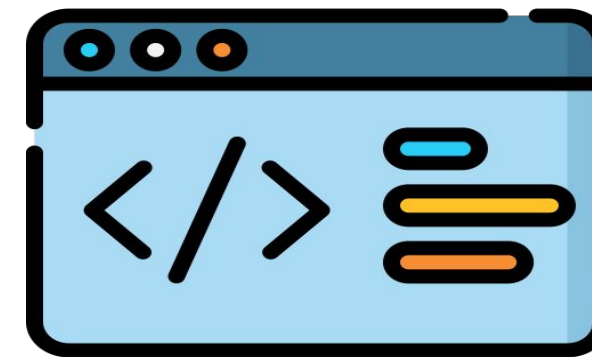
O ideal é automatizar os testes para simplificar o processo de teste.

Vantagens

Estas são algumas das vantagens de realizar testes de componentes ou testes de unidade dentro de um projeto:



Reduz o custo de teste, porque os defeitos são detectados antecipadamente.



Melhora o design e o código do software, porque permite uma melhor refatoração do software.

Vantagens

Estas são algumas das vantagens de realizar testes de componentes ou testes de unidade dentro de um projeto:



Reduz defeitos em recursos recém-desenvolvidos ou reduz bugs, alterando a funcionalidade existente.



Em modelos de desenvolvimento incremental e iterativo, em que as alterações de código estão em andamento, o teste de regressão de componentes automatizado desempenha um papel fundamental na construção da confiança de que as alterações não danificaram os componentes existentes.

Testes de unidade **inadequados** farão com que os defeitos se propaguem para testes de nível superior e isso levará a um **alto custo de correção** de defeitos durante os testes do sistema, testes de integração e até testes de aceitação do usuário.

Fazer os testes de unidade **adequados** no início do desenvolvimento **economiza esforço, tempo e dinheiro no final**.



02

Frameworks

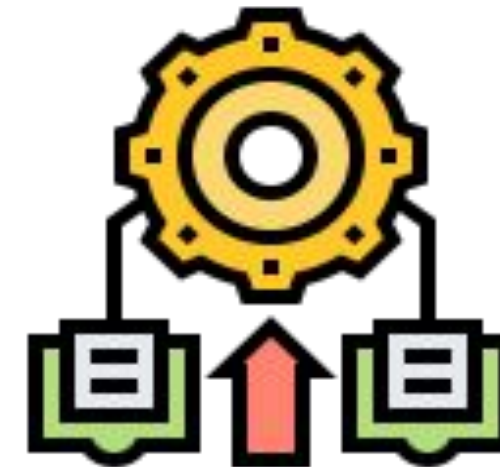
Frameworks

Os testes unitários podem ser de dois tipos:



Manuais

Pode ser usado um documento instrucional com o passo a passo.



Automatizados

Uma estrutura automatizada é necessária para escrever os scripts de teste.

O que é necessário para automatizar testes de unidade?

TEST RUNNER

É uma ferramenta que executa os testes e apresenta os resultados em forma de relatório.

Por exemplo: Mocha
(<https://mochajs.org/>)

ASSERTION LIBRARY

É uma ferramenta usada para validar a lógica de teste, condições e resultados esperados.

Por exemplo: Chai
(<https://www.chaijs.com/guide/>)



JEST é uma estrutura que inclui o test runner e a assertion library.

Frameworks mais utilizados



JUnit

Ferramenta de teste usada para a linguagem de programação Java. Fornece asserções para identificar o método de teste. Essa ferramenta testa os dados primeiro e depois os insere no fragmento de código.



NUnit

É uma estrutura de teste de unidade amplamente usada para todas as linguagens **.net**. É uma ferramenta de código aberto e suporta testes orientados a dados que podem ser executados em paralelo.



JMockit

É uma ferramenta de cobertura de código com métricas de declaração e decisão. Permite simular APIs com sintaxe de registro e verificação.

Frameworks mais utilizados



Emma

É um conjunto de ferramentas de código aberto para analisar e relatar código escrito na linguagem Java. Emma suporta tipos de cobertura como método, instrução, bloco básico.



PHPUnit

É uma ferramenta de teste de unidade para programadores PHP. Permite aos desenvolvedores usar métodos de confirmação predefinidos para afirmar que um sistema se comporta de uma determinada maneira.

Frameworks para Javascript

Para JavaScript, usaremos o framework **JEST**.
Se quisermos configurá-lo, devemos instalar:

1. NodeJS (<https://nodejs.org/>)
2. Uma IDE (o recomendado é o Visual Studio Code
<https://code.visualstudio.com/>)
3. JEST (<https://jestjs.io/>)



Visual Studio Code

Muito obrigado!