

# TESTES AUTOMATIZADOS

Usando código  
para testar  
código

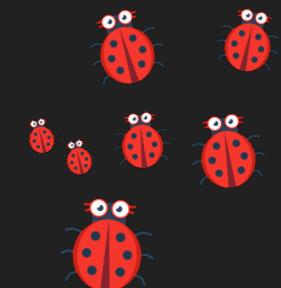
# PHELIPE PERBOIRES



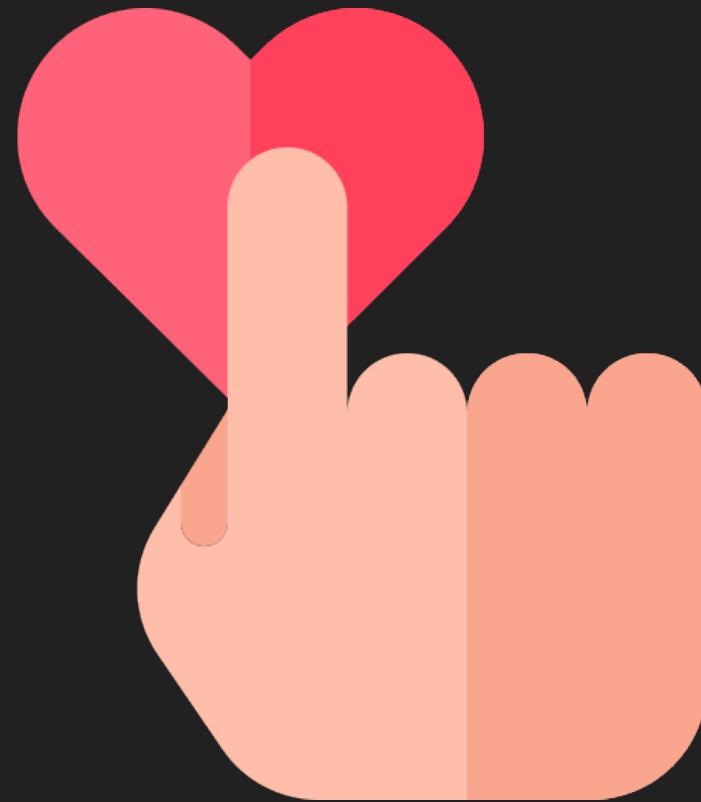
pperboires

Arquiteto de Software  
Desenvolvedor desde 2003  
Cofundador Elleva

Só escrevi meu primeiro teste  
automatizado 5 anos depois



# AGENDA



# AGENDA

- Algumas definições e comparações
- Mitos sobre testes automatizados
- Característica de um teste automatizado
- TDD - Test Driven Development
- Tipos de testes automatizados
- Ferramentas
- Hands on
- Considerações finais

# O que é TESTE?

Mecanismo que busca **verificar ou provar** a **verdade sobre algo ou alguém**.

Procedimento para **avaliar** as **características ou qualidades** de **algo ou de alguém**



Teste de motor



Teste de airbag



Teste do pneu



Teste de colisão



Teste de rodagem

# Indústria automobilística (1860)

E na indústria de  
desenvolvimento  
de software  
(1955)?



<https://sou.gohorseprocess.com.br/extreme-go-horse-xgh/>

# Curiosidade...



- Quem aqui **trabalha com desenvolvimento de software**?
- Quem aqui **já escreve testes**?
- Quem aqui **sempre escreve testes**?
- Quem aqui tem uma **suite de testes que considera satisfatória**?
- Quem aqui tem **coragem de atualizar o sistema no meio do expediente**?

# E por que não escrevemos testes?

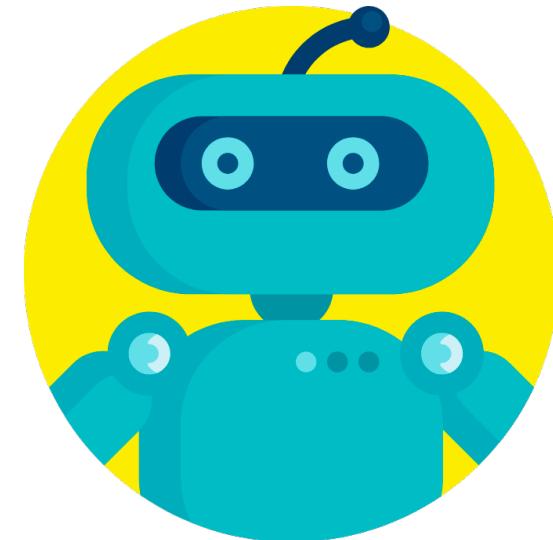


# Existem duas categorias principais de teste



**Testes manuais**

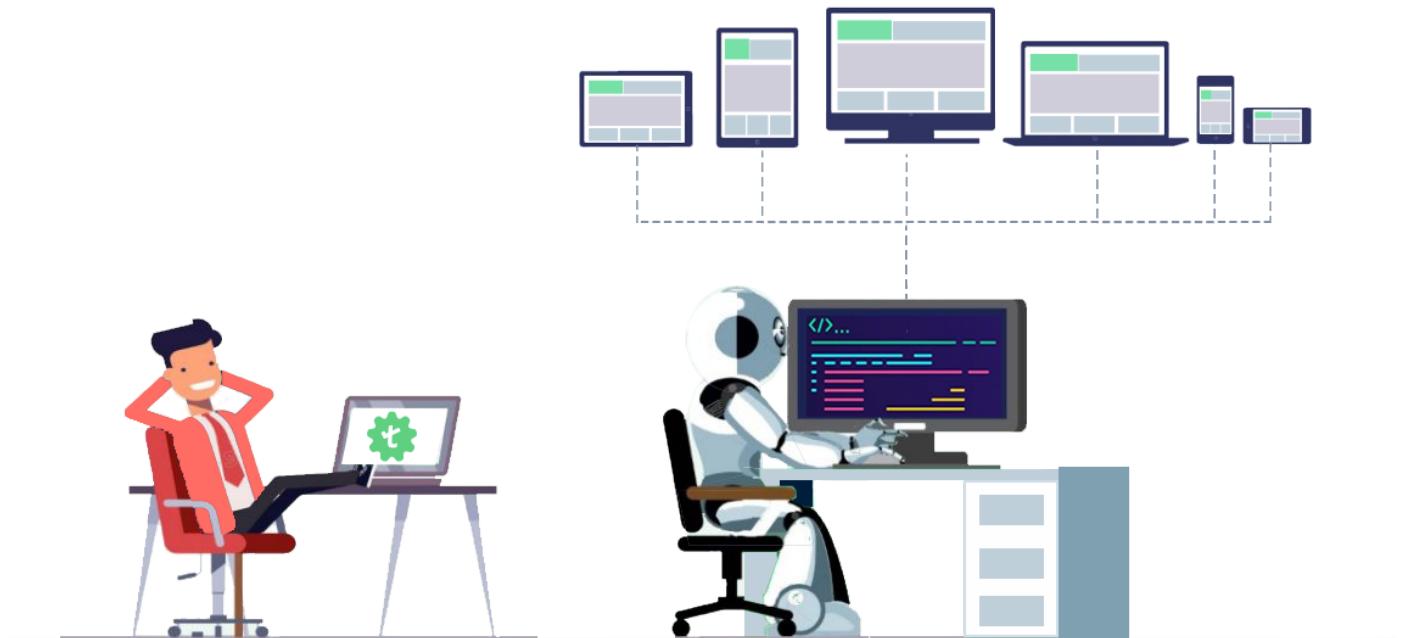
(min)



**Testes automatizados**

(ms/seg)

# O que é TESTE AUTOMATIZADO?



São **rotinas** que **preparam um cenário** específico, **executam** alguma operação e **comparam** os **resultados reais** com os **resultados previstos** ou **esperados**.

# Os três As



# Teste de soma de uma calculadora (2+3)



Ligar a calculadora

- **Digitar número 2**
- **Apertar botão “+”**
- **Digitar número 3**
- **Apertar botão “=”**

Verificar se o resultado na tela foi 5

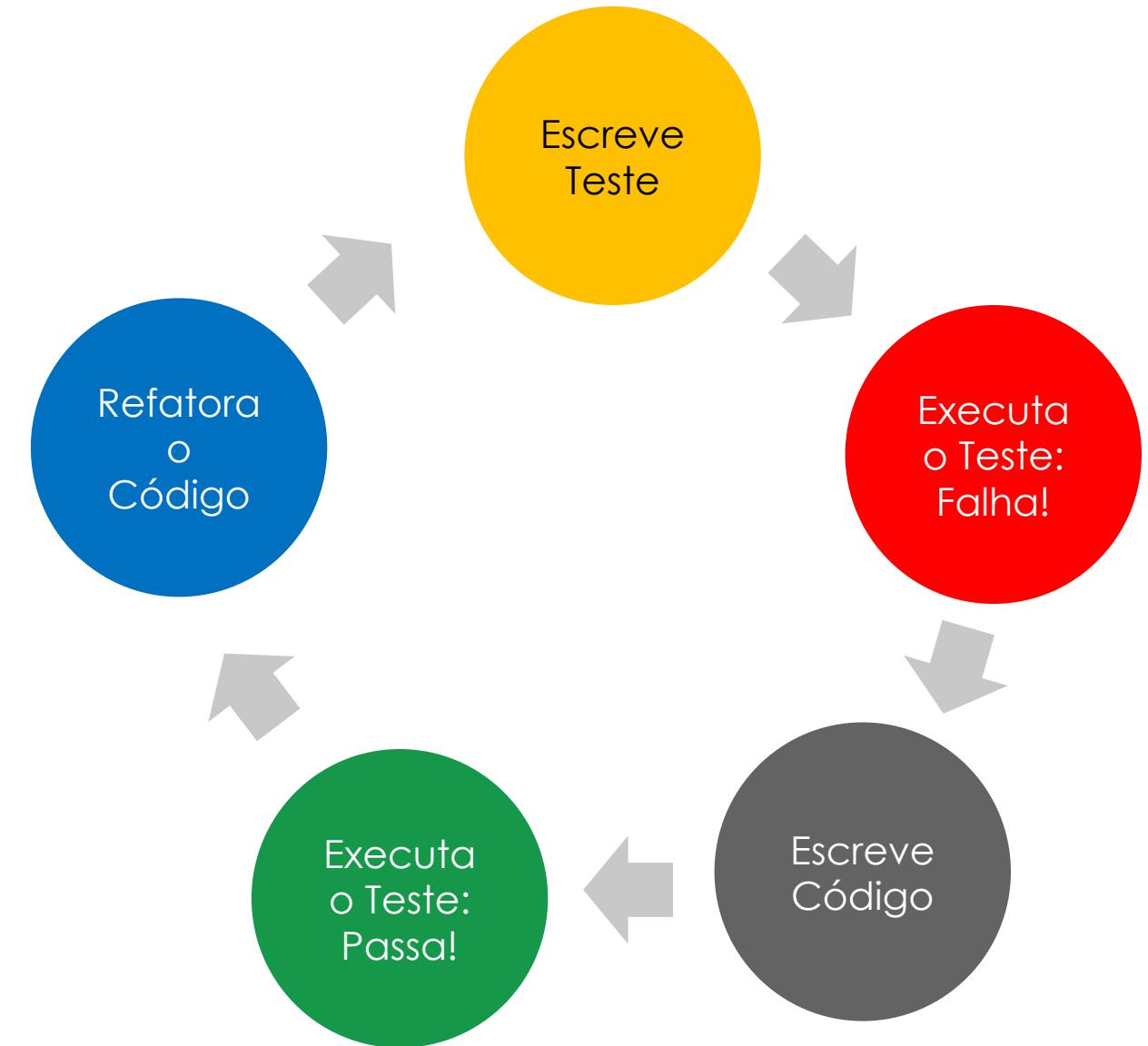
# TDD – Test Driven Development

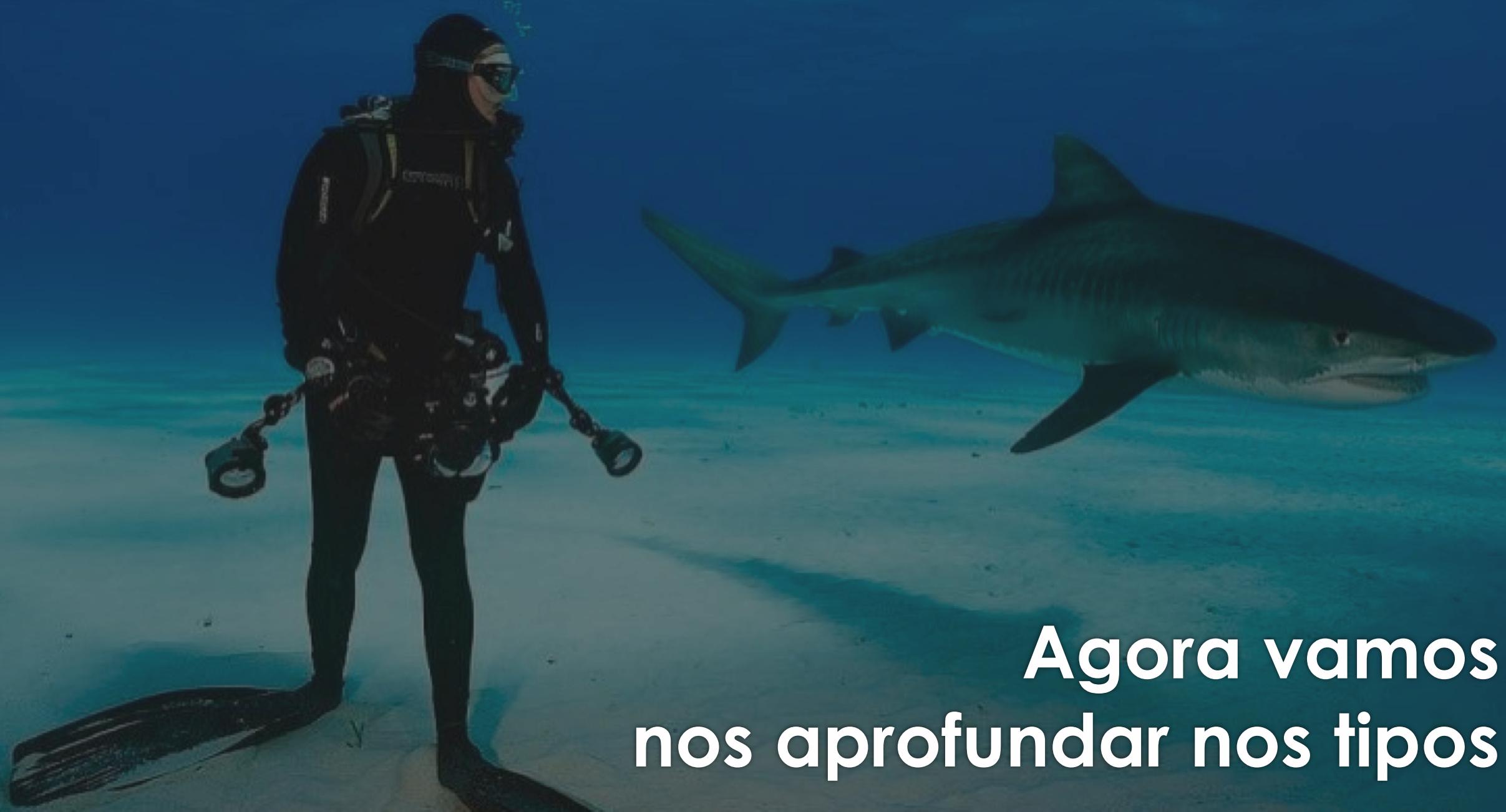
- Kent Back, 2003 (o cara do XP, do Manifesto Agil)
- O **Teste é escrito primeiro** e **guiará** a escrita do código
- Código a ser escrito deve **ter como objetivo fazer o teste** passar

CODING DOJO

by elleva

# Ciclo TDD





Agora vamos  
nos aprofundar nos tipos

# Exemplo que irei usar...



Carrinho de E-Commerce



Produtos



Cupom de Desconto

# Testes de Unidade

**Definição:**

Verifica a **menor parte testável** do software (chamada de unidade)

**Unidade é testada** de forma **isolada**.

Java	.NET	JavaScript	Python
Junit Spock	Xunit NUnit MSTest	Jasmine Jest Mocha	unittest

# Carrinho deve começar zerado



Um **carrinho** deve  
começar zerado.

# Vamos ao código!



# Adicionar item no carrinho

R\$ 1.000,00

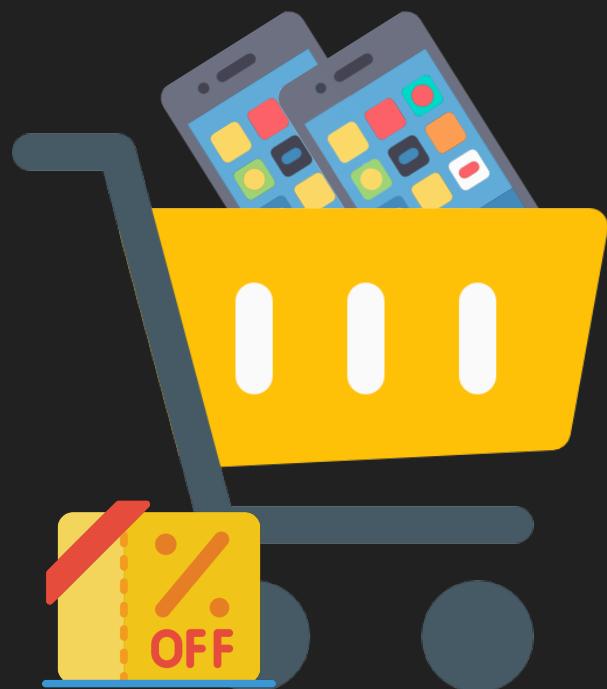


Ao **adicionar item** no carrinho, o seu **valor total** deve ser **atualizado**.

# Vamos ao código!



# Calcular desconto do carrinho



Um **desconto deve ser aplicado** ao valor total do carrinho, **considerando o cupom**.

# Vamos ao código!

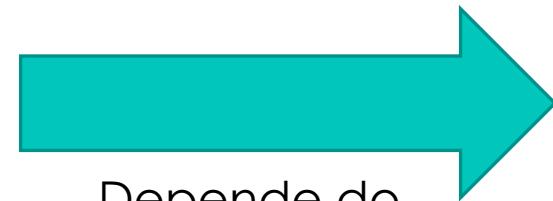


# Tá... mas o que é Mock / Mock Objects?

- São objetos que **imitam objetos reais** (são dublês)
- Permitem que **façamos verificações (asserts)** para saber se **foram chamados ou não**
- **Permitem controlar o seu comportamento**

Java	.NET	JavaScript	Python
JMockit Mockito EasyMock JMock MockCreator	NMockLib Rhino Mocks NMock NSubstitute	JsMockito Sinon.JS Jest	unittest.mock mock

# Dependência entre componentes

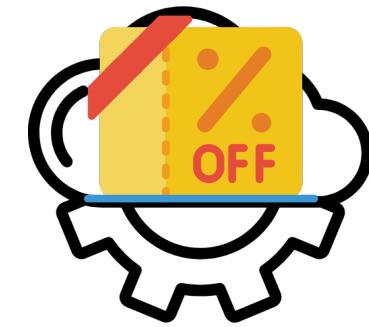
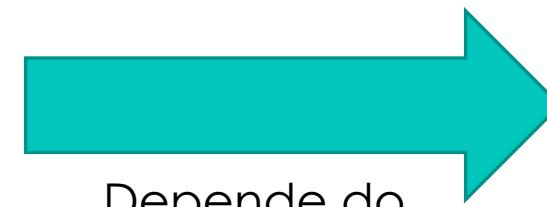


Depende do



Serviço que  
calcula o desconto

# Nos testes de unidade...



Objeto que simula o  
Serviço que  
calcula o desconto

Voltando...

# Calcular desconto do carrinho



Um **desconto deve ser aplicado** ao valor total do carrinho **considerando o cupom**.

# Vamos ao código!



Mas os  
testes de  
unidade  
não pegam  
tudo...



# Testes de Integração

## Definição:

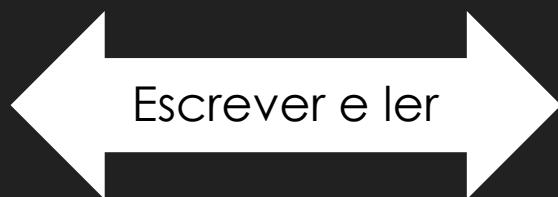
Verifica **se uma unidade tem o comportamento esperado quando funciona de maneira integrada a outros elementos de software**, como chamada de serviços, APIs e banco de dados.

Java	.NET
DBUnit	NDBUnit TestServer

# Quantos testes podem ser feitos nessa consulta?

```
SELECT U.NOME AS NOME_USUARIO  
      , P.NOME AS NOME_PREFERENCIA  
      , P.VALOR AS VALOR_PREFERENCIA  
  FROM USUARIO U  
 LEFT JOIN PREFERENCIAS P ON U.ID = P.USUARIO_ID  
 WHERE U.PRIMEIRO_ACESSO > @DATA  
   AND U.ESTADO = 'ATIVADO'
```

# Armazenar o carrinho em um BD



Postgres

# Passo-a-passo de um teste de integração (que envolve persistencia em banco de dados)



- Criar um database
- Criar tabelas
- Criar views
- Criar massa de dados

**Executar alguma operação que altera o estado do banco**

**Fazer uma consulta e verificar se o resultado é o esperado**

E no final de tudo, apagar o database!

# Vamos ao código!



# Testes Funcionais (e2e, Aceitação, UI...)

## Definição:

Neste tipo de teste, é feita uma **simulação das ações de um usuário real** interagindo com o software, como por exemplo, o **preenchimento de um cadastro**, selecionando uma opção ou um clique do mouse.

## Ferramentas:

- Frameworks de Automação de Web Browser como SeleniumHQ

# Fluxo de Inscrição do Macaé Tech

The image displays a screenshot of the Macaé Tech website, specifically the registration process. It shows three main stages of the workflow:

- Step 1: Initial Registration Page**

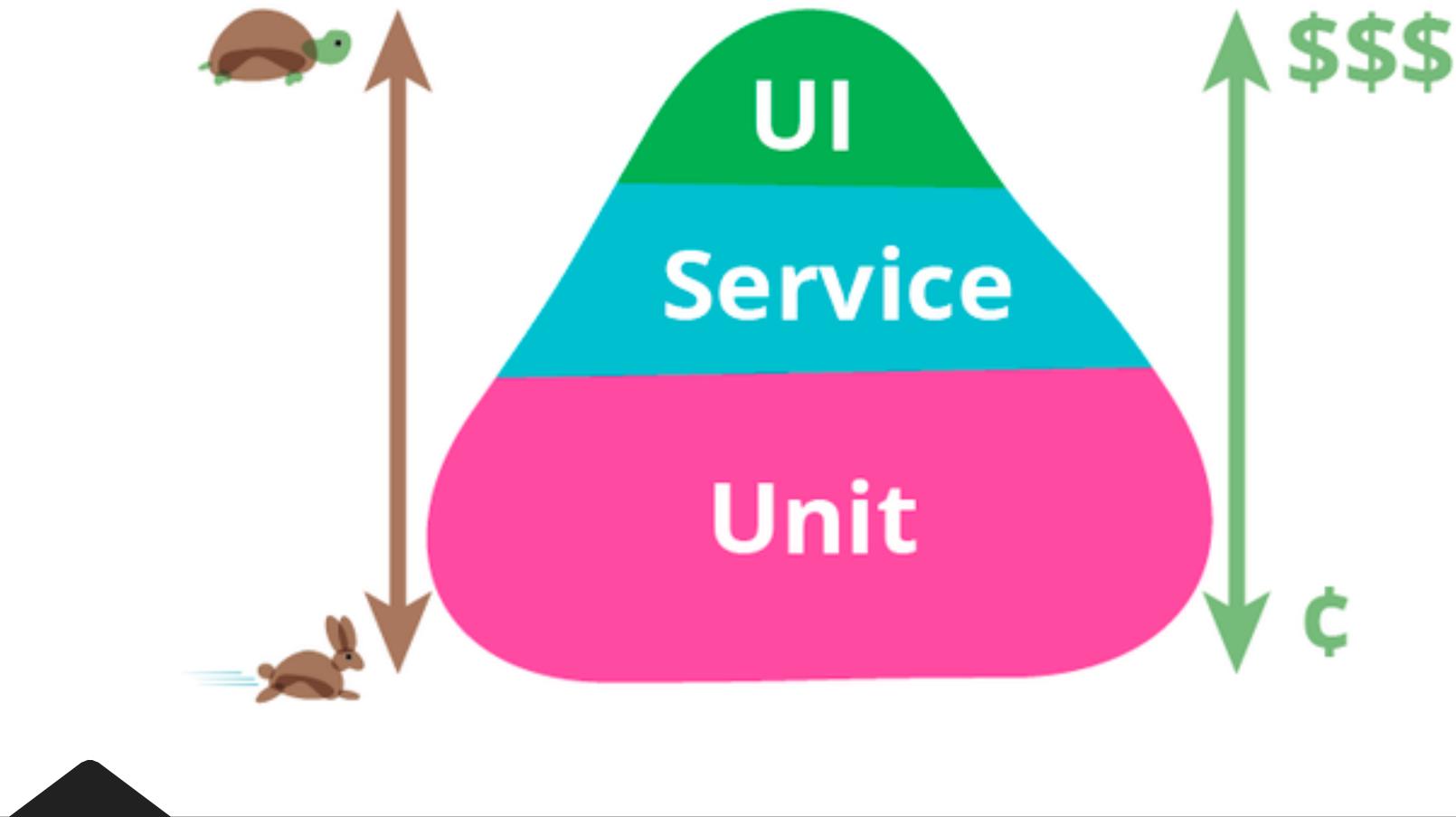
This page features the event logo and a prominent red button labeled "Reserve sua vaga!" (Reserve your spot!). A yellow circle highlights this button.
- Step 2: Participant Information Form**

This step involves filling out participant details. A yellow box highlights the "E-mail \*" field. Below it, there's a section for "Informação para o recebimento do comprovante" (Information for receiving proof) with a dropdown menu for "Copiar informações".
- Step 3: Confirmation and Error Handling**

This final step shows the confirmation of email fields ("E-mail \*", "Confirmação do e-mail \*") and an error message: "Oops... O campo E-mail é obrigatório." (The email field is mandatory). A yellow oval highlights this error message area.

# Vamos ao código!





## Pirâmide dos Testes

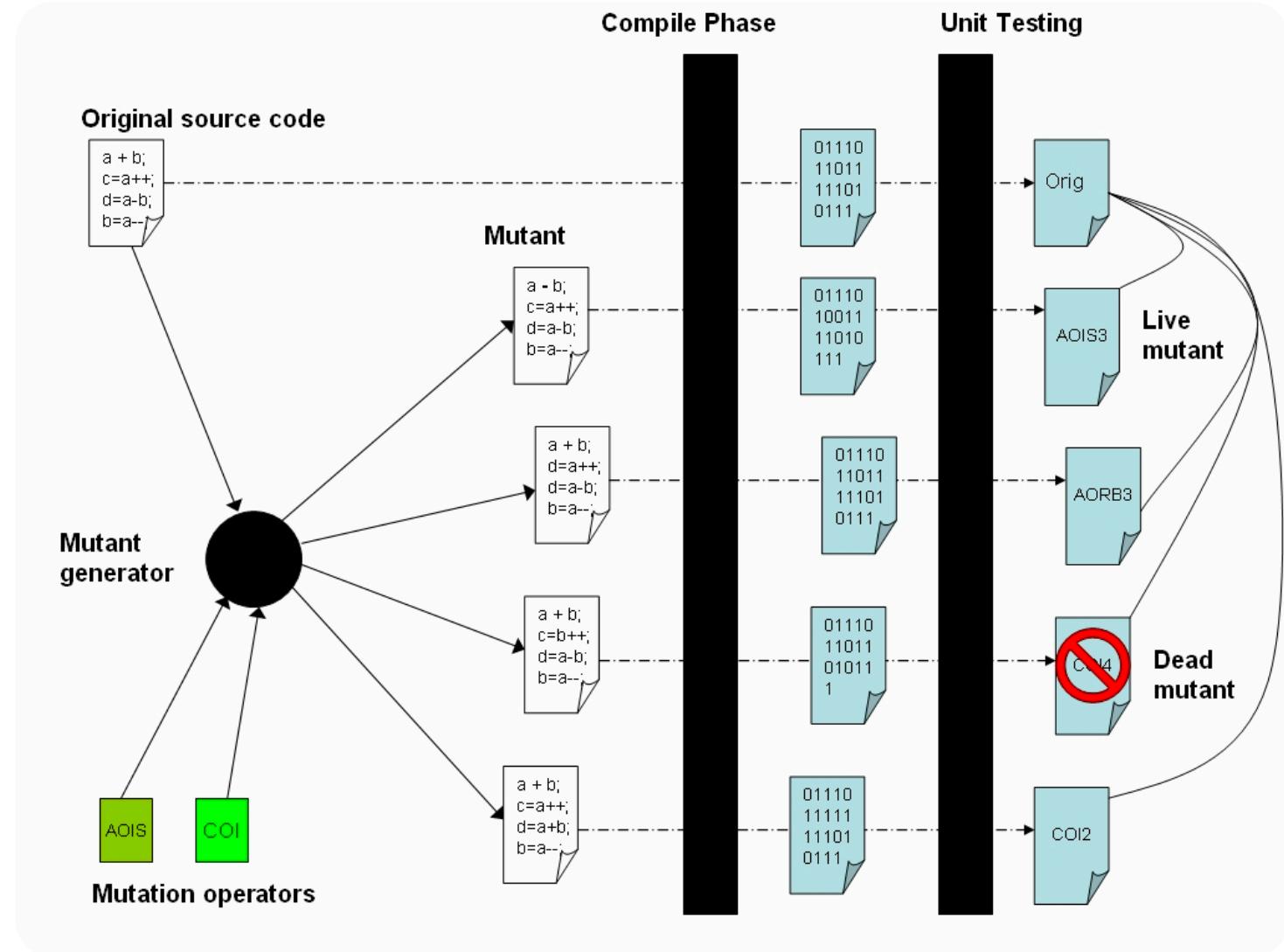
A close-up shot of a man with a shocked or surprised expression. He has light-colored hair and is wearing a dark jacket over a white shirt. His hands are raised to his face, with his fingers covering his mouth and nose, as if he is shouting or reacting to something unexpected.

Olhe!  
Faltou escrever um teste aqui!

# Análise de Cobertura de Testes

▼ ◄ Elleva.Macaetech.ECommerce.Shared	82%	<div style="width: 82%; background-color: #2e6b2e; height: 10px;"></div>	15/83
▼ ECommerceException	100%	<div style="width: 100%; background-color: #2e6b2e; height: 10px;"></div>	0/3
* ECommerceException(string)	100%	<div style="width: 100%; background-color: #2e6b2e; height: 10px;"></div>	0/3
▼ ItemCarrinho	100%	<div style="width: 100%; background-color: #2e6b2e; height: 10px;"></div>	0/12
► Quantidade	100%	<div style="width: 100%; background-color: #2e6b2e; height: 10px;"></div>	0/2
► ValorUnitario	100%	<div style="width: 100%; background-color: #2e6b2e; height: 10px;"></div>	0/2
► Produto	100%	<div style="width: 100%; background-color: #2e6b2e; height: 10px;"></div>	0/2
* ItemCarrinho(Produto,int,decimal)	100%	<div style="width: 100%; background-color: #2e6b2e; height: 10px;"></div>	0/6
▼ Produto	100%	<div style="width: 100%; background-color: #2e6b2e; height: 10px;"></div>	0/9
► Nome	100%	<div style="width: 100%; background-color: #2e6b2e; height: 10px;"></div>	0/2
► Valor	100%	<div style="width: 100%; background-color: #2e6b2e; height: 10px;"></div>	0/2
* Produto(string,decimal)	100%	<div style="width: 100%; background-color: #2e6b2e; height: 10px;"></div>	0/5
▼ Carrinho	85%	<div style="width: 85%; background-color: #2e6b2e; height: 10px;"></div>	8/52
► Total	100%	<div style="width: 100%; background-color: #2e6b2e; height: 10px;"></div>	0/2
► TotalProdutos	100%	<div style="width: 100%; background-color: #2e6b2e; height: 10px;"></div>	0/2
* Carrinho(ICalculatorDesconto,Nullable<Guid>)	0%	<div style="width: 0%; background-color: #d9534f; height: 10px;"></div>	0/6
* AdicionarItem(Produto,int)	100%	<div style="width: 100%; background-color: #2e6b2e; height: 10px;"></div>	0/8
* DefineCodigoCupomDesconto(string)	100%	<div style="width: 100%; background-color: #2e6b2e; height: 10px;"></div>	0/3
► RemoverItem(Produto)	100%	<div style="width: 100%; background-color: #2e6b2e; height: 10px;"></div>	0/8
* QuantidadePorProduto(Produto)	100%	<div style="width: 100%; background-color: #2e6b2e; height: 10px;"></div>	0/8
* AtualizarQuantidade(Produto,int)	75%	<div style="width: 75%; background-color: #d9534f; height: 10px;"></div>	2/8
► Código	50%	<div style="width: 50%; background-color: #d9534f; height: 10px;"></div>	1/2
► TotalItens	0%	<div style="width: 0%; background-color: #d9534f; height: 10px;"></div>	1/1
* ObterItemPorProduto(Produto)	0%	<div style="width: 0%; background-color: #d9534f; height: 10px;"></div>	4/4
► CarrinhoRepositorio	0%	<div style="width: 0%; background-color: #d9534f; height: 10px;"></div>	7/7

# Testes de Mutação



# Vamos a DEMO!



# Suite de Testes

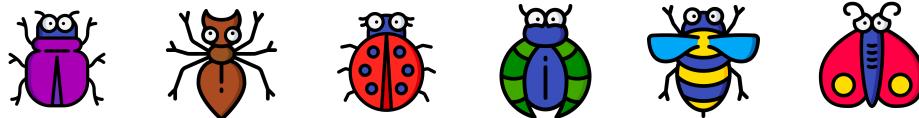


# Quais as consequências quando não temos?

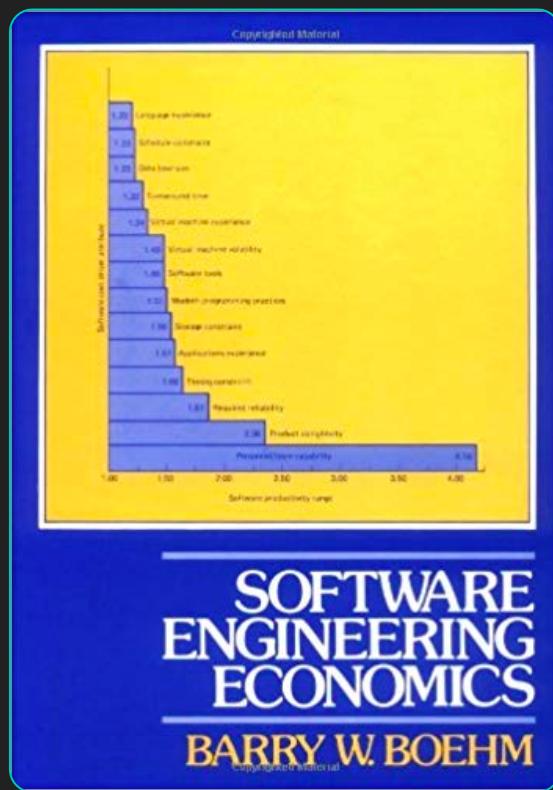
(uma boa suite de testes)

- Aumenta a carga de trabalho ao longo do tempo
- Cria um time responsável por garantir a qualidade do que está sendo entregue
- Afeta a entrega contínua
- Produz código que ninguém tem coragem de mexer
- Dificulta experimentar novas tecnologias
- Dificulta detectar erros em cenários mais complexos
- Visão míope sobre o que está pronto! (Tá pronto, só falta testar!)

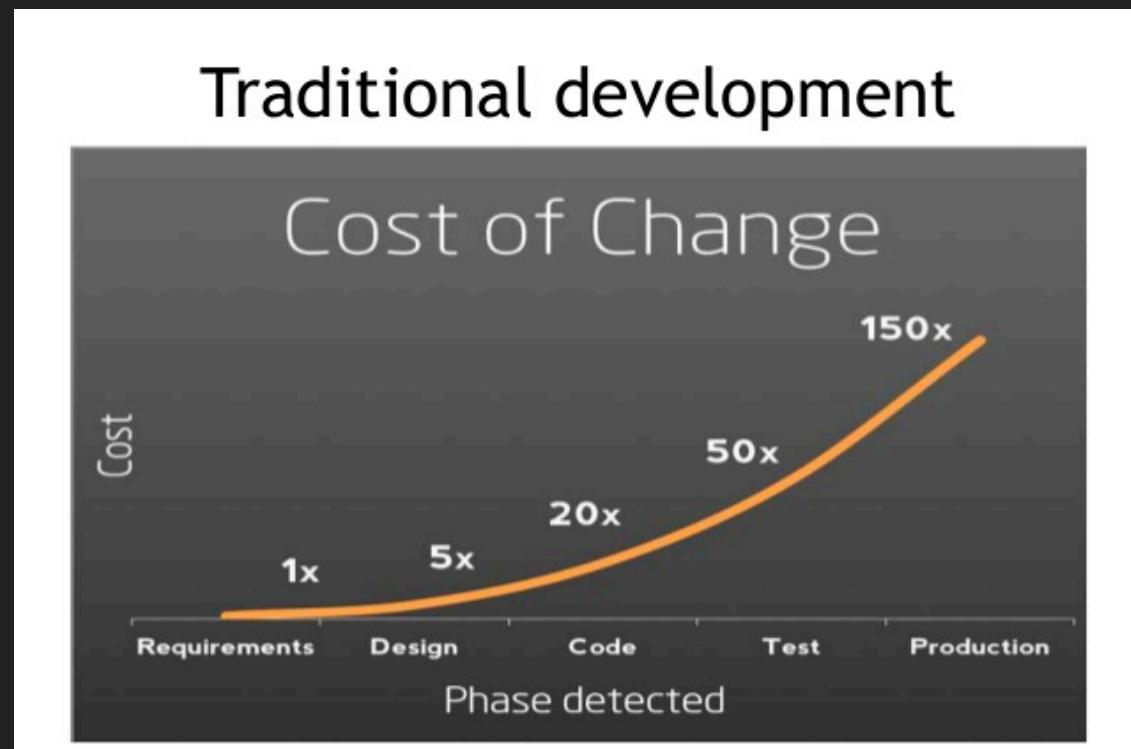
Produção de muuuitos



# Custo da Mudança de um Software



1981



# E comentários no código como esses



```
// Quando eu escrevi isso, só Deus e eu sabíamos o que eu estava fazendo  
// Agora só Deus sabe.
```



```
// Mágico. Não toque!
```



```
// Se você achou que conseguiria melhorar esse código e percebeu que comenteu um erro,  
// incremente o contador abaixo com a quantidade de horas gastas como forma de aviso para  
// o próximo que tentar.  
// Tempo total desperdiçado até agora: 16h
```

E esse que  
eu fui  
testemunha



// Até essa linha eu testei!!!

# Outras coisas que vale a pena estudar...

- BDD com Cucumber
- Selenium GRID
- Page Object Pattern
- Integração Contínua
- Entrega Contínua
- Autofixture
- Uso de IA para testes
- Coesão e Acomplamento
- Injeção de Dependencia e IoC



```
public class ApresentacaoTest {  
  
    [Test]  
    public void ao_concluir_apresentacao_devem_existir_duvidas() {  
  
        // ARRANGE  
        var apresentacao = new Apresentacao();  
  
        // ACT  
        apresentacao.IrParaUltimoSlide();  
  
        var duvidas = apresentacao.ObterDuvidas();  
  
        // ASSERT  
        Assert.IsFalse(0, duvidas.Count, "Ninguém teve dúvida. /=");  
    }  
}
```

Obrigado!