  test(

    'Calcular o valor do produto com desconto em porcentagem passando o valor do produto zerado',

    () {

      expect(app.calcularDesconto(0, 150, false), -150);

    },

  );

Valor do produto 0

Desconto no produto 150

0 - -150 = -150

Na logica esta certo, mas que empresa venderia um produto por 0 e ainda daria 150 reais de desconto?

Entao para isso podemos gerar uma exceção

Acertando a função para não deixa o valor do produto ser 0

double calcularDesconto(double valor, double desconto, bool percentual) {

  if (valor <= 0) {

    throw ArgumentError("O valor do produto não pode ser zero.");

  }

  if (percentual) {

    // Imagina que estamos dando desconto

    // Imagine a situação

    // Produto 100R$ - Desconto 20%

    return valor - ((valor \* desconto) / 100); // Desconto 20 reais

    // Ai seria 100 - 20 = 80 reais

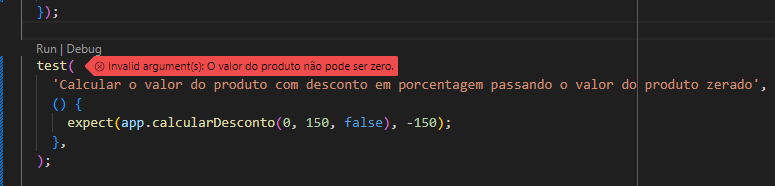
    // O novo valor seria 80 reais.

  }

  return valor - desconto;

}

Fazendo o teste.



Deu erro, mas não quero um erro, eu quero esperar um erro.

Quero um Matcher do tipo argumento

Matcher é oque esperamos receber da função que estamos testando.

Vamos criar uma arrow function

  test(

    'Calcular o valor do produto com desconto em porcentagem passando o valor do produto zerado',

    () {

      expect(() => app.calcularDesconto(0, 150, false), -150);

    },

  );

Em vez de esperar o 150, vamos esperar um Matcher do tipo Argumento

E pra isso vamos usar o ThrowsA(TypeMatcher<ArgumentError>)

 test(

    'Calcular o valor do produto com desconto em porcentagem passando o valor do produto zerado',

    () {

      expect(() => app.calcularDesconto(0, 150, false), throwsA(TypeMatcher<ArgumentError>()));

    },

  );

Resumindo oque ele quer?

Quando passar um valor 0 eu quero que ele espere um erro

E esse erro deve ser do tipo Arguments

Atençao a exceção esta dentro da nossa função, se você quiser outro tipo de exceção você deve mudar la e também no teste.

## Outro exemplo

Agora calculando com o desconto zerado, onde não poderíamos fazer.

double calcularDesconto(double valor, double desconto, bool percentual) {

  if (valor <= 0) {

    throw ArgumentError("O valor do produto não pode ser zero.");

  }

  if (desconto <= 0) {

    throw ArgumentError("O desconto não pode ser 0");

  }

  if (percentual) {

    // Imagina que estamos dando desconto

    // Imagine a situação

    // Produto 100R$ - Desconto 20%

    return valor - ((valor \* desconto) / 100); // Desconto 20 reais

    // Ai seria 100 - 20 = 80 reais

    // O novo valor seria 80 reais.

  }

  return valor - desconto;

}

    test('Calcular o valor do produto com desconto ZERADO!', () {

    expect(() => app.calcularDesconto(100, 0, true), throwsA(TypeMatcher<ArgumentError>()));

  });

Executar em nível de terminal



Lembrar que todos os testes em dart deve acabar com a palavra test

