Kotlin com testes

Fernando Claudiano da Silva
fernandoclaudianosilva@gmail.com
fernando.silva401@fatec.sp.gov.br

Resumo

Nesse capitulo do e-book vamos abordar um cenário de testes, para um projeto de calculadora feita em Kotlin e como deve ser implementado no código.

Os testes são essenciais no desenvolvimento de software, pois ajudam a garantir que o código funcione conforme o esperado, reduzindo a chance de bugs, evitando que um código chegue com defeitos a produção.

Introdução

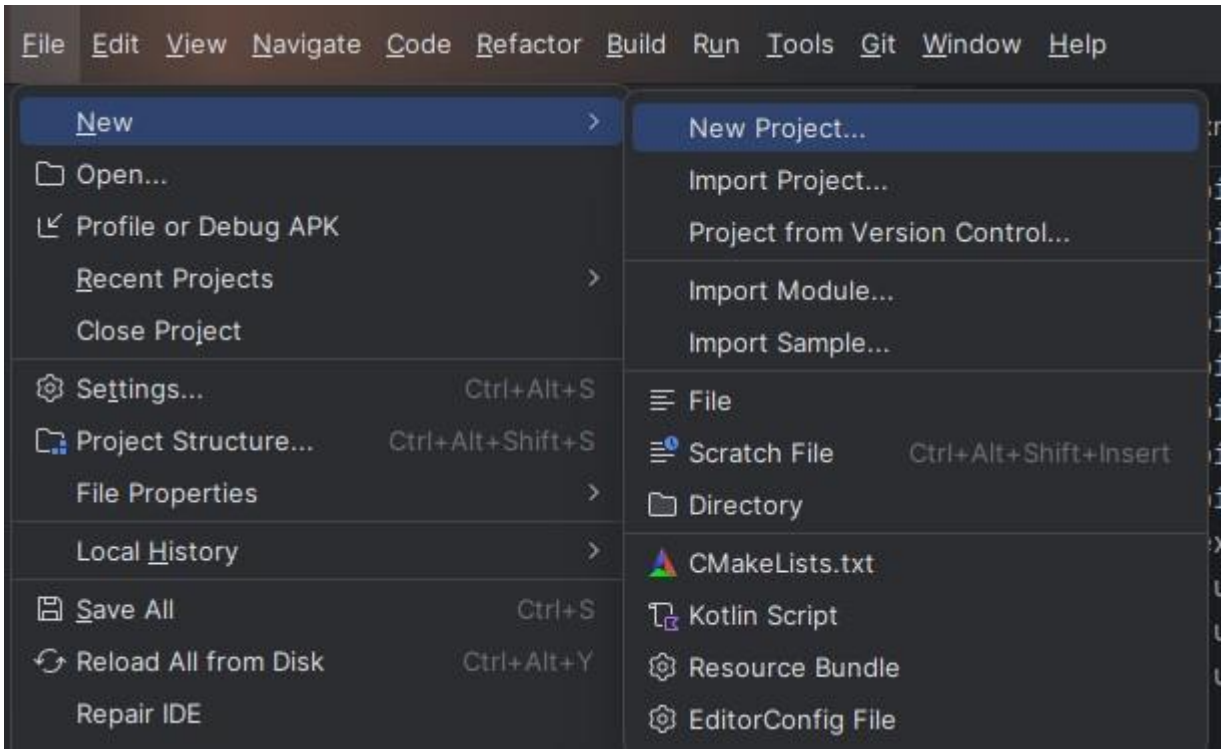
Eu estudei a ferramenta Kotlin, já lecionada na matéria da disciplina de desenvolvimento web multiplataforma, mas decidi me aprofundar nessa ferramenta pelo foco no desenvolvimento de aplicativos Androids, e por ser uma linguagem simples que possibilita fazer as mesmas coisas do que em outras ferramentas, só que em menos linhas de código, como o Java por exemplo.

A linguagem de programação Kotlin, desenvolvida pela JetBrains em 2011 e anunciada oficialmente em 2016, tem ganhado popularidade principalmente no desenvolvimento de aplicativos Android. Kotlin é uma linguagem moderna, concisa, segura e interoperável com Java, tornando-se a escolha oficial para o desenvolvimento Android desde 2017 (JETBRAINS, 2024).

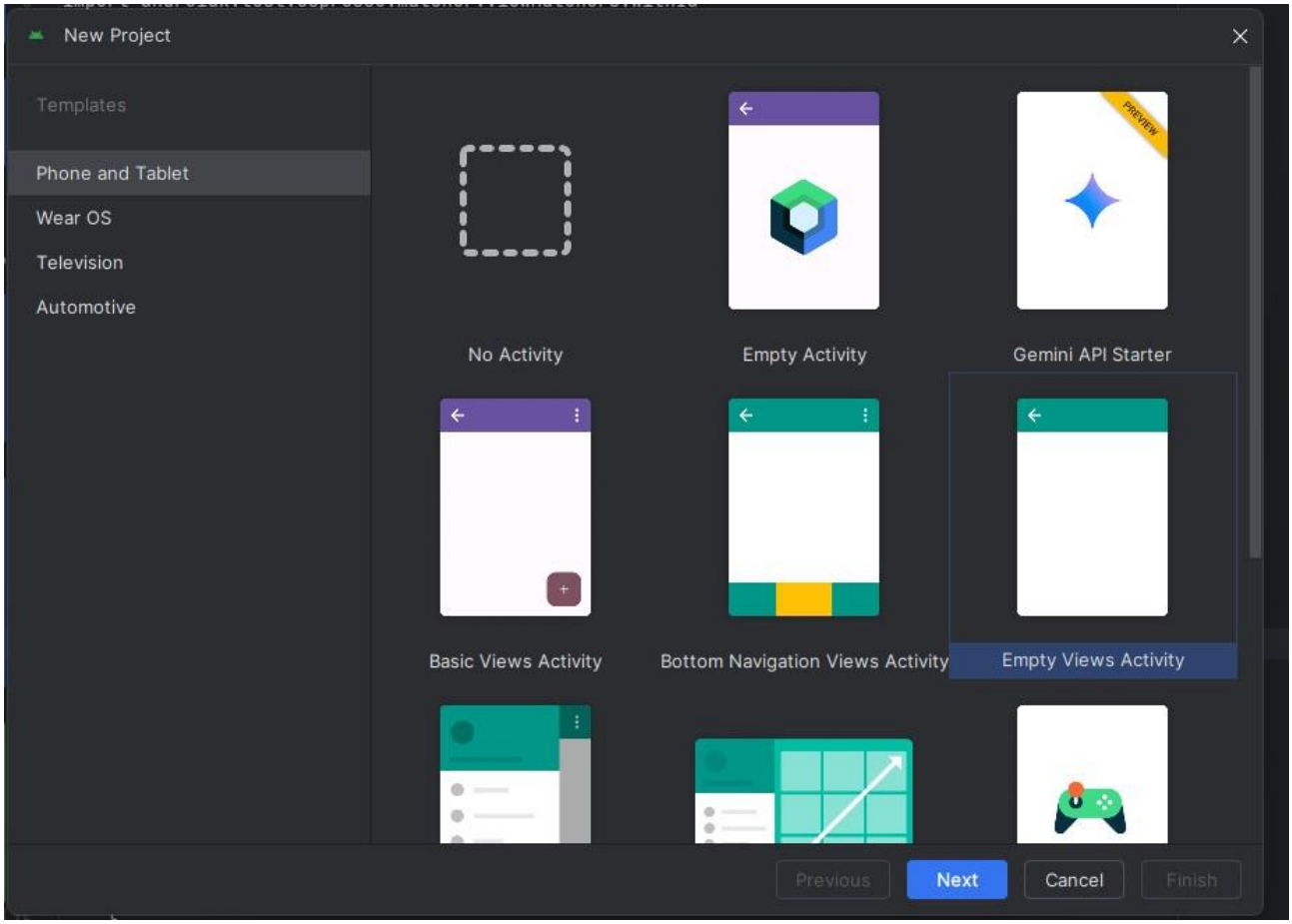
Desenvolvimento

CRIANDO UM PROJETO

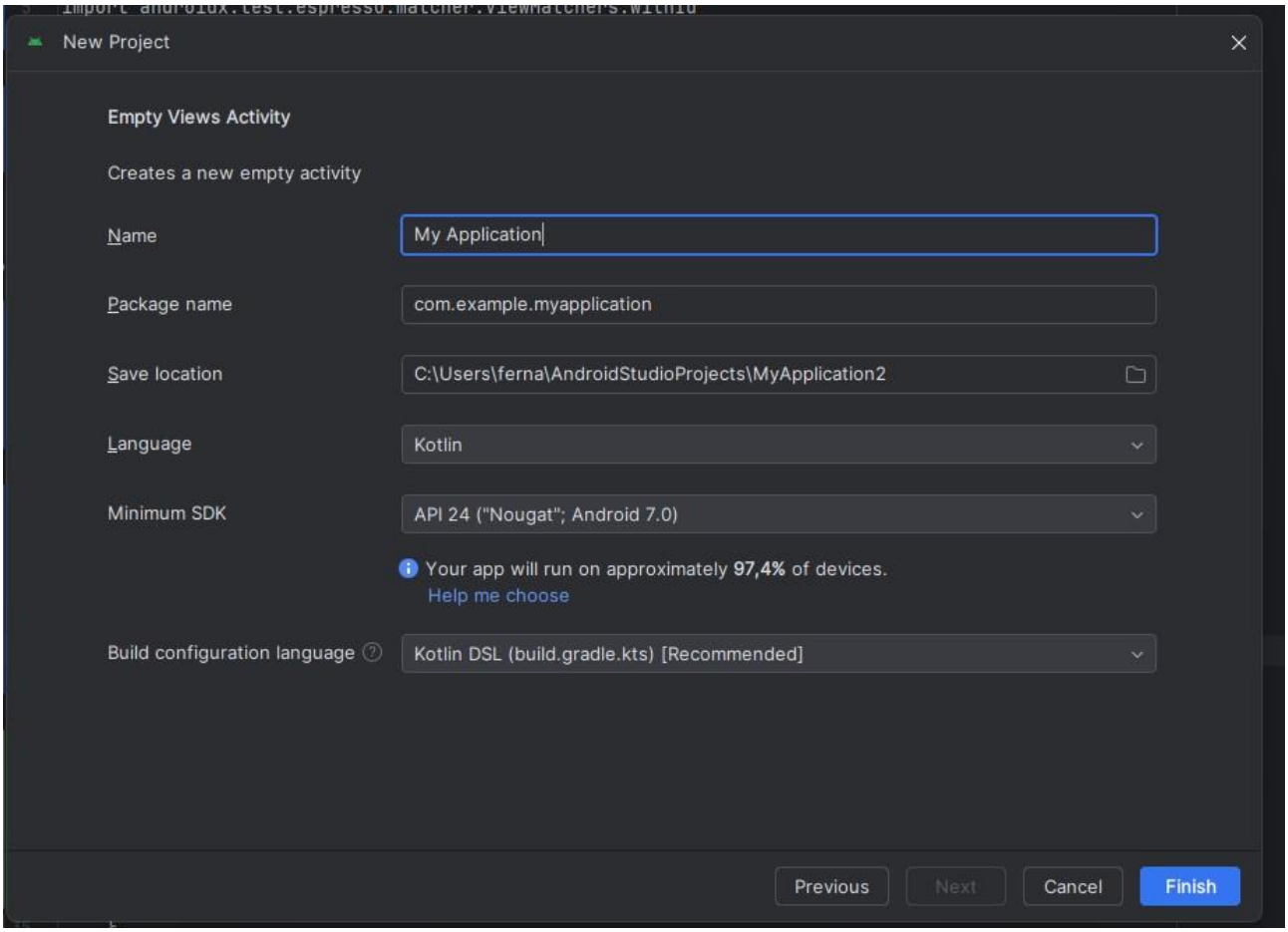
1.No aplicativo android, selecione **file | new | new projetc.**



2.Escolha a opção **Empty View Activity**

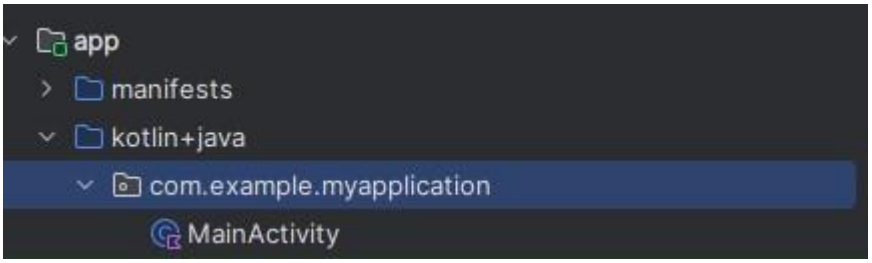


3.Nomeie sua aplicação e selecione a opção **Finish** para terminar de criar



CRIANDO UMA APLICAÇÃO

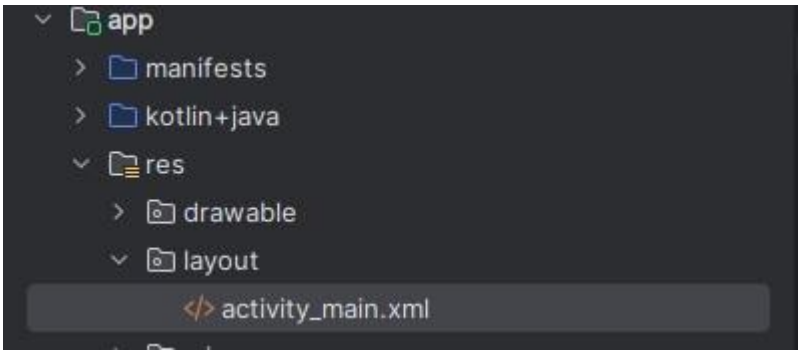
1.Abra o arquivo **MainActivity.kt** em **app/kotlin+java/com.example.myapplication**



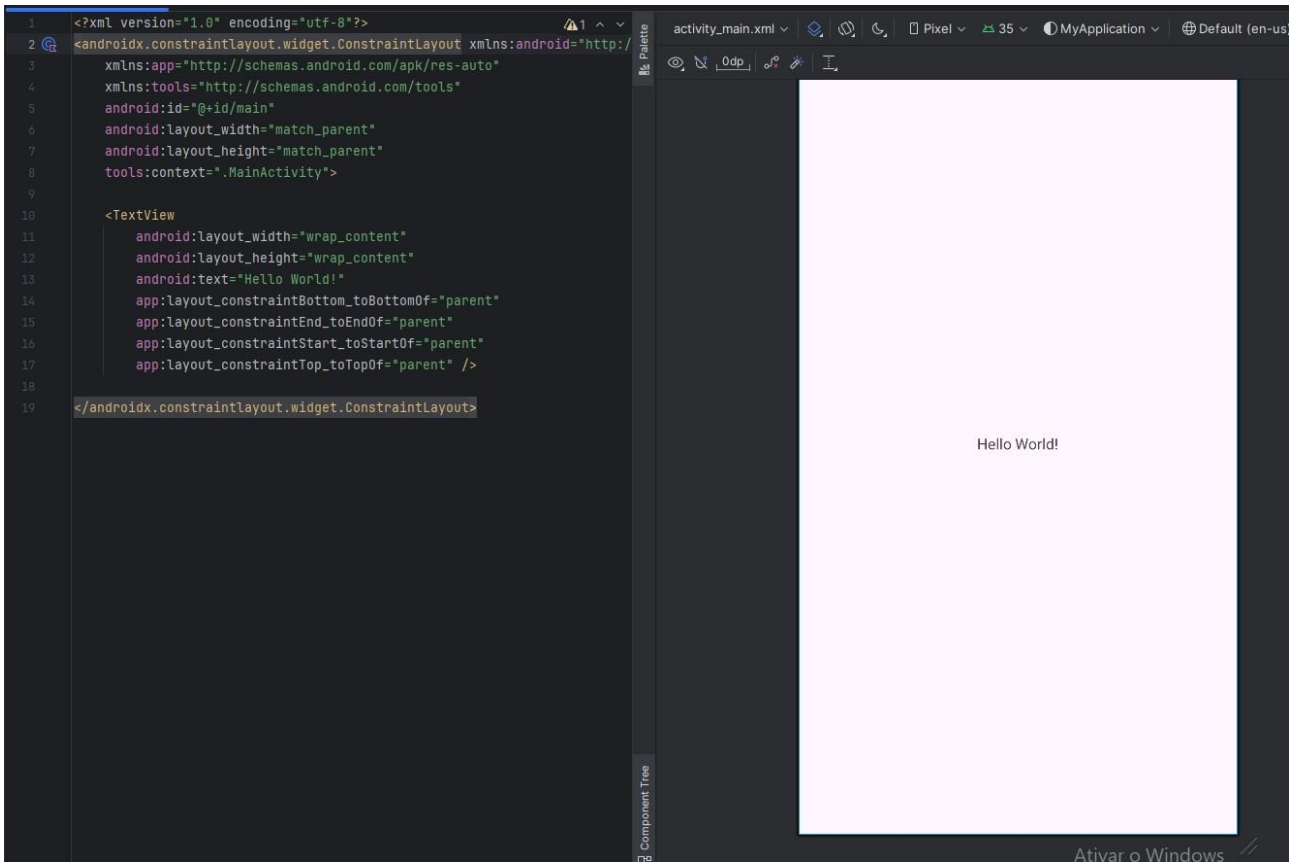
aqui será a parte funcional do aplicativo

```kotlin
package com.example.myapplication

import ...

class MainActivity : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        enableEdgeToEdge()
        setContentView(R.layout.activity_main)
        ViewCompat.setOnApplyWindowInsetsListener(findViewById(R.id.main)) { v, insets ->
            val systemBars = insets.getInsets(WindowInsetsCompat.Type.systemBars())
            v.setPadding(systemBars.left, systemBars.top, systemBars.right, systemBars.bottom)
            insets
        }
    }
}
```

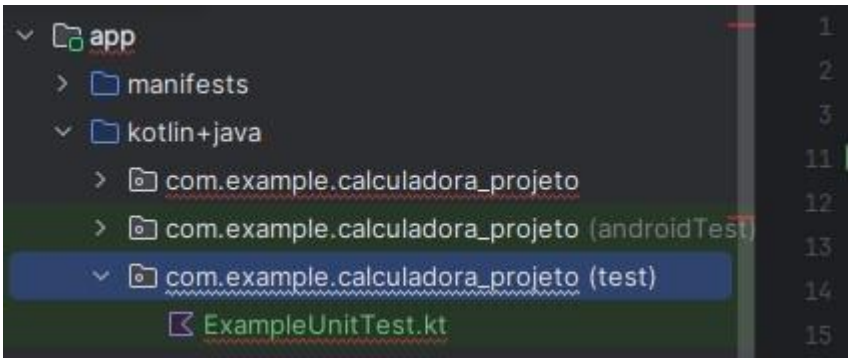
2.Abre o arquivo **acitivty_main.xml** em **app/res/layout**



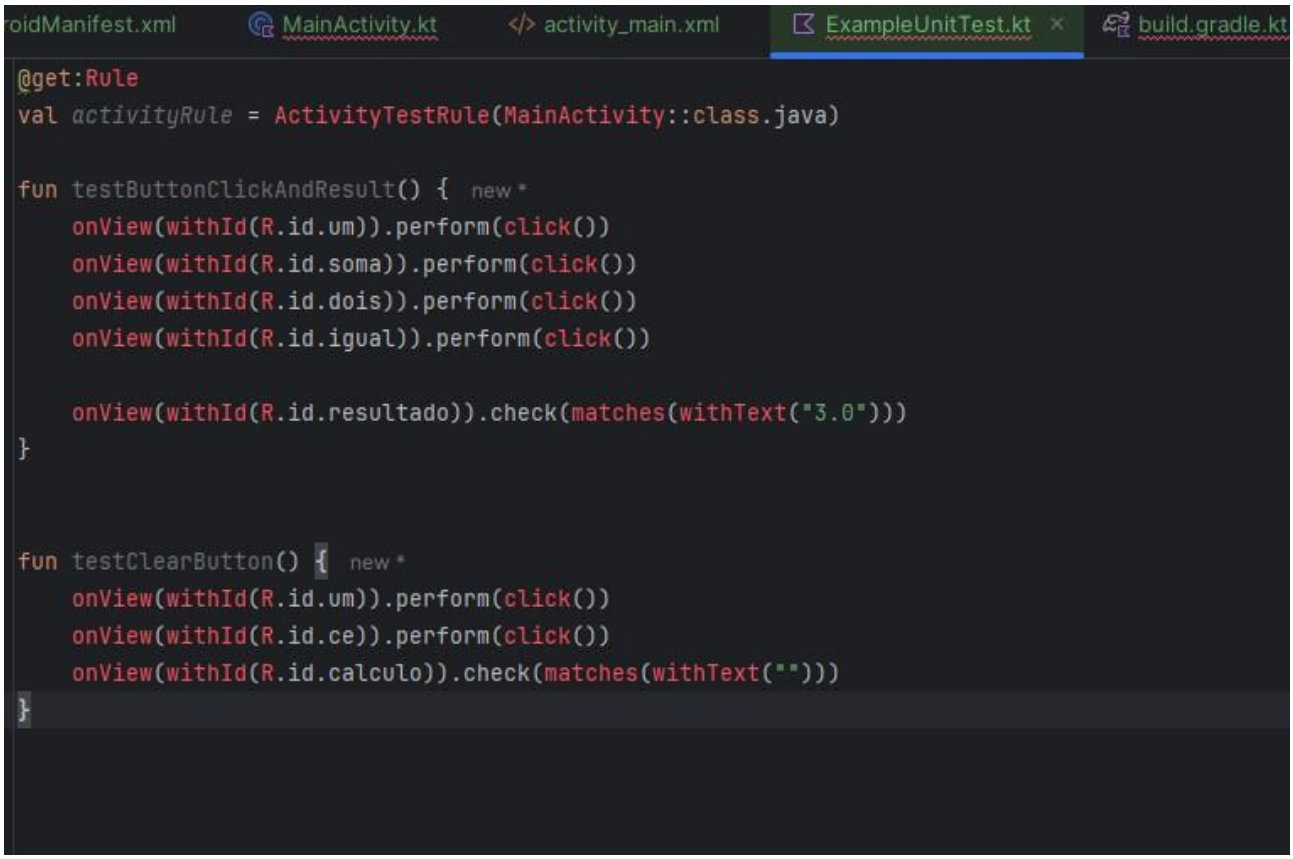
Aqui será criado a parte visual



3.Abra o arquivo **ExampleUnitTest.kt** em **app/kotlin+java/com.calculadoraprojeto**



Aqui será feito os testes unitários



Você pode conferir um projeto que eu criei de uma calculadora em kotlin no meu git no link: https://github.com/fernandoclaudianosilva/calculadora-kotlin

Como funciona os testes?

```
fun testButtonClickAndResult() {  new *
    onView(withId(R.id.um)).perform(click())
    onView(withId(R.id.soma)).perform(click())
    onView(withId(R.id.dois)).perform(click())
    onView(withId(R.id.igual)).perform(click())

    onView(withId(R.id.resultado)).check(matches(withText("3.0")))
}
```

Este teste simula uma operação de soma na calculadora.

1. **Interações com os botões da interface:**

onView(withId(R.id.um)).perform(click()): Localiza o botão com o ID um e simula um clique nele.

onView(withId(R.id.soma)).perform(click()): Localiza o botão de soma (soma) e clica nele.

onView(withId(R.id.dois)).perform(click()): Localiza o botão dois e clica nele.

onView(withId(R.id.igual)).perform(click()): Clica no botão de igual (igual) para calcular o resultado.

```
fun testClearButton() {  new *
    onView(withId(R.id.um)).perform(click())
    onView(withId(R.id.ce)).perform(click())
    onView(withId(R.id.calculo)).check(matches(withText("")))
}
```

Validação do resultado:

onView(withId(R.id.resultado)). check(matches(withText("3.0"))):Verifica se o elemento com o ID resultado contém o texto "3.0", ou seja, a soma foi calculada corretamente.

Considerações Finais

A realização de testes no desenvolvimento de aplicativos Kotlin pelo Android Studio traz inúmeros benefícios, como a identificação precoce de erros, maior estabilidade e garantia de funcionalidades alinhadas aos requisitos. Essa prática reduz o retrabalho, assegurando um desenvolvimento mais ágil e eficaz. O Android Studio oferece ferramentas integradas que facilitam a criação de testes unitários e instrumentados, permitindo validar tanto a lógica de negócios quanto a experiência do usuário em diferentes dispositivos e cenários.

Bibliografia

Documentação Oficial do Kotlin: https://kotlinlang.org