Desenvolvimento para Dispositivos Móveis

Aula 03

Ambiente de desenvolvimento Android + Kotlin

Prof. Msc. Francisco de Assis de Souza Rodrigues

UniFAP
CENTRO UNIVERSITÁRIO PARAÍSO

Visão Geral

1. Configuração do Ambiente de Desenvolvimento
2. Introdução à Linguagem Kotlin
3. Kotlin no Android
4. Crie um projeto usando o modelo
5. Criar e gerenciar dispositivos virtuais
6. Desafios e Encerramento

UniFAP
CENTRO UNIVERSITÁRIO PARAÍSO



Configuração do Ambiente de Desenvolvimento

UniFAP
CENTRO UNIVERSITÁRIO PARAÍSO

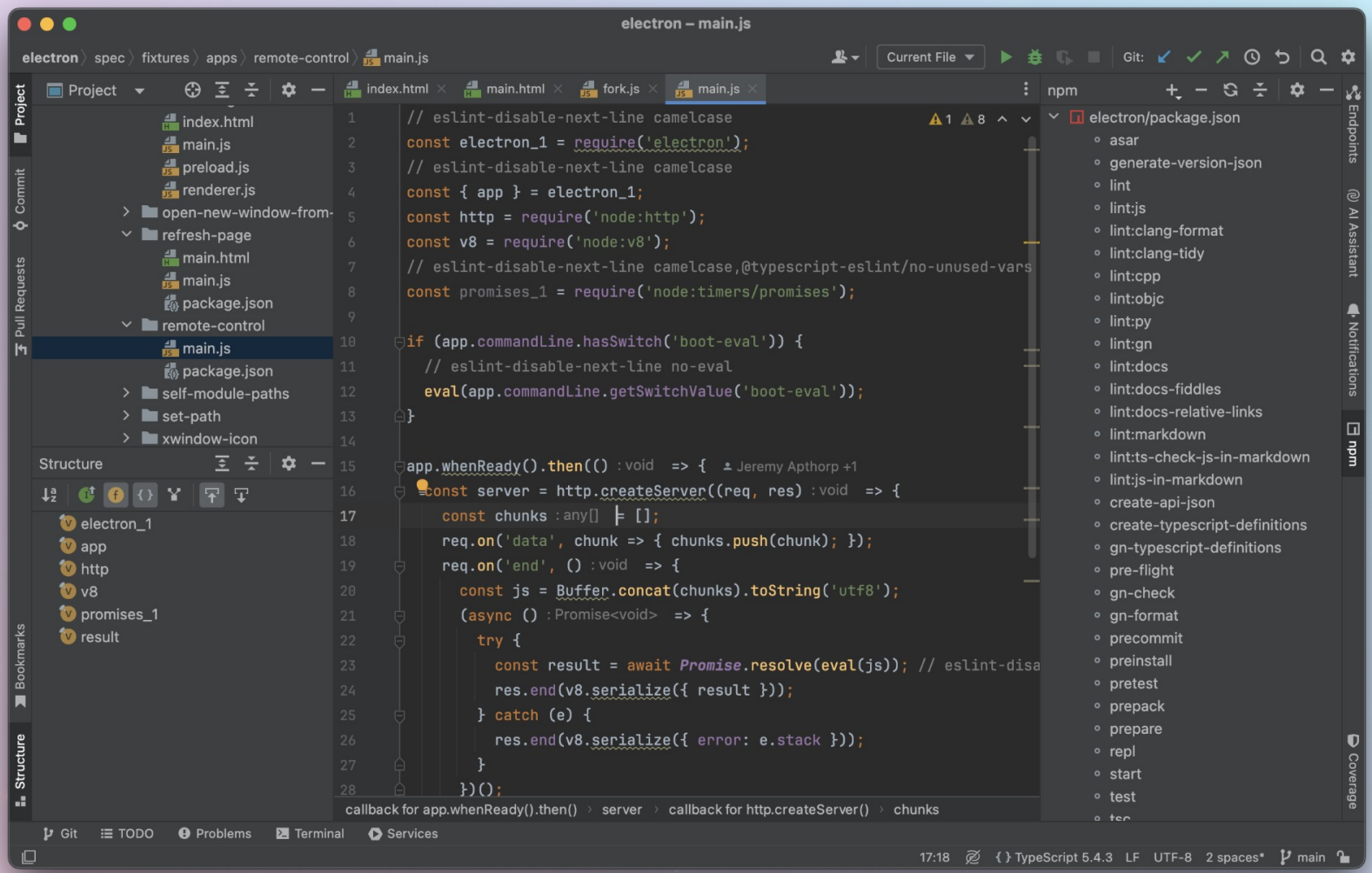
Instalação do Android Studio

1. Requisitos do sistema: https://developer.android.com/studio/install?hl=pt-br
2. Acessa o site: https://developer.android.com/
3. Fazer download da versão do android studio para a versão do seu SO.
4. Inicia o processo de instalação pelo guia.

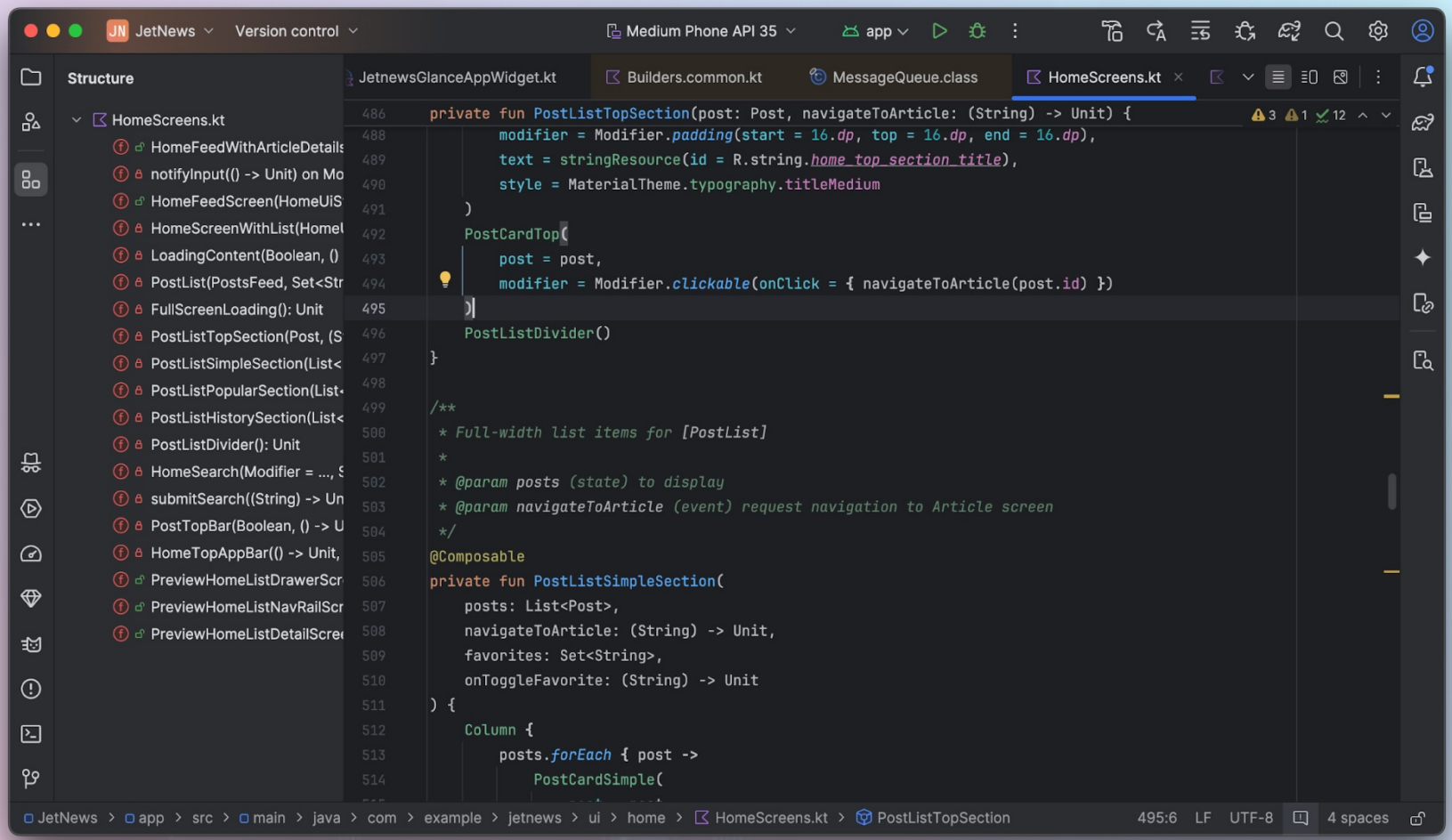
Nova interface no Android Studio

Android Studio Ladybug apresenta a **nova interface** padrão da atualização da **plataforma IntelliJ 2024.2**. https://developer.android.com/studio/intro/new-ui?hl=pt-br

- Temas e ícones
- Barra de ferramentas principal simplificada
- Widgets para Git e projetos
- Executar configurações e ações de criação de perfil
- Janelas de ferramentas reformuladas
- Divisões verticais e horizontais
- Novo local para os controles de ferramentas da interface
- Barra de navegação
- Modo compacto



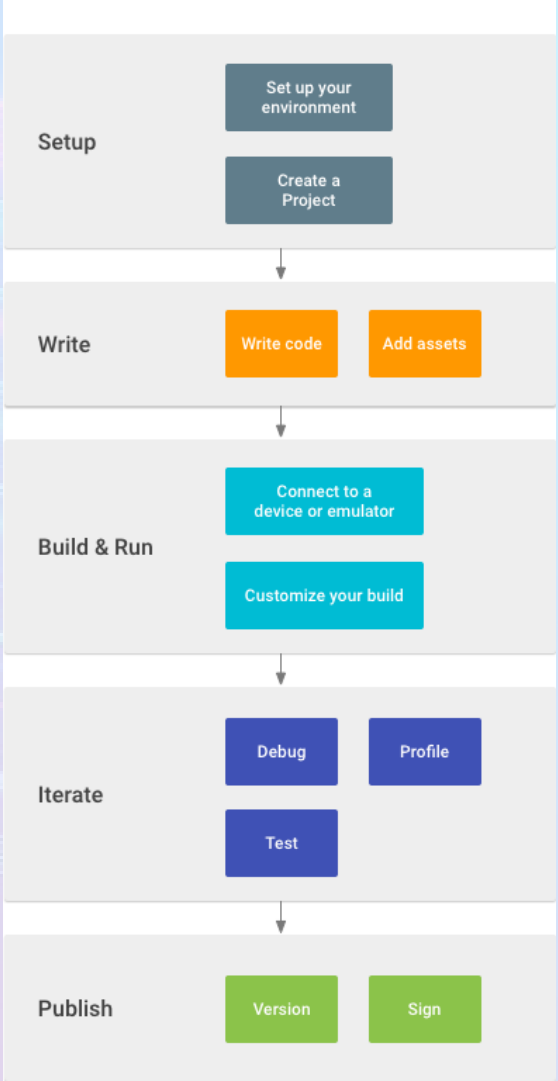
UniFAP
CENTRO UNIVERSITÁRIO PARAÍSO



UniFAP
CENTRO UNIVERSITÁRIO PARAÍSO

Developer workflow basics

https://developer.android.com/studio/workflow

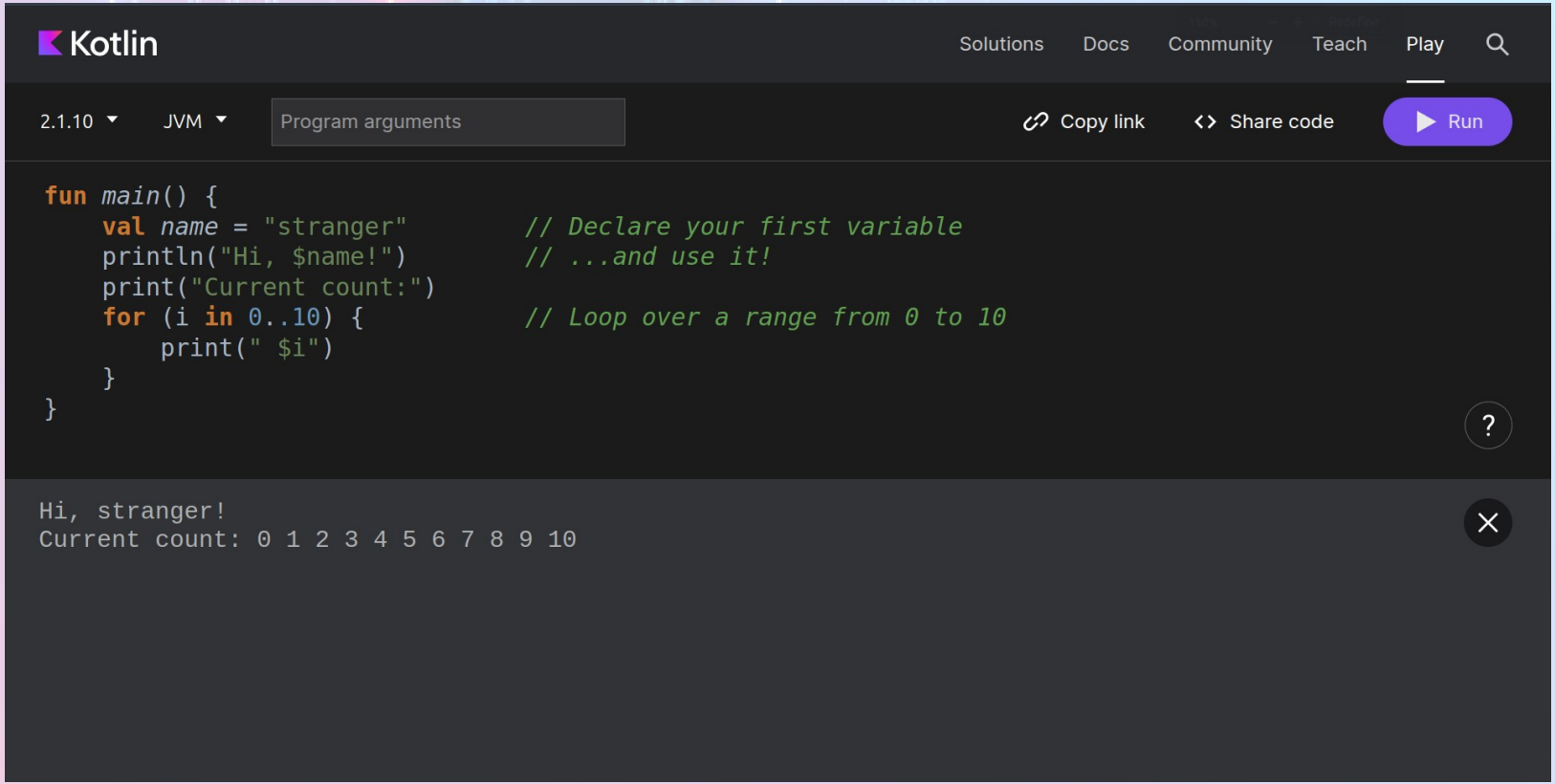




Introdução à Linguagem Kotlin

UniFAP
CENTRO UNIVERSITÁRIO PARAÍSO

Kotlin - Simples



Kotlin - Assíncrono

Kotlin Solutions Docs Community Teach Play

2.1.10 JVM Program arguments Copy link Share code Run

```kotlin
import kotlinx.coroutines.*

suspend fun main() {                            // A function that can be suspended and resumed later
    val start = System.currentTimeMillis()
    coroutineScope {                            // Create a scope for starting coroutines
        for (i in 1..10) {
            launch {                            // Start 10 concurrent tasks
                delay(3000L - i * 300)          // Pause their execution
                log(start, "Countdown: $i")
            }
        }
    }
    // Execution continues when all coroutines in the scope have finished
    log(start, "Liftoff!")
}

fun log(start: Long, msg: String) {
    println("$msg " +
            "(on ${Thread.currentThread().name}) " +
            "after ${(System.currentTimeMillis() - start)/1000F}s")
}
```

```
Countdown: 9 (on DefaultDispatcher-worker-1 @coroutine#9) after 0.417s
Countdown: 8 (on DefaultDispatcher-worker-1 @coroutine#8) after 0.717s
Countdown: 7 (on DefaultDispatcher-worker-1 @coroutine#7) after 1.016s
Countdown: 6 (on DefaultDispatcher-worker-1 @coroutine#6) after 1.316s
Countdown: 5 (on DefaultDispatcher-worker-1 @coroutine#5) after 1.616s
Countdown: 4 (on DefaultDispatcher-worker-1 @coroutine#4) after 1.916s
Countdown: 3 (on DefaultDispatcher-worker-1 @coroutine#3) after 2.216s
Countdown: 2 (on DefaultDispatcher-worker-1 @coroutine#2) after 2.514s
Countdown: 1 (on DefaultDispatcher-worker-1 @coroutine#1) after 2.815s
Liftoff! (on DefaultDispatcher-worker-1 @coroutine#1) after 2.816s
```

Kotlin - Orientado a Objetos

Kotlin Solutions Docs Community Teach Play

2.1.10 JVM Program arguments Copy link Share code Run

```kotlin
abstract class Person(val name: String) {
    abstract fun greet()
}

interface FoodConsumer {
    fun eat()
    fun pay(amount: Int) = println("Delicious! Here's $amount bucks!")
}

class RestaurantCustomer(name: String, val dish: String) : Person(name), FoodConsumer {
    fun order() = println("$dish, please!")
    override fun eat() = println("*Eats $dish*")
    override fun greet() = println("It's me, $name.")
}

fun main() {
    val sam = RestaurantCustomer("Sam", "Mixed salad")
    sam.greet() // An implementation of an abstract function
    sam.order() // A member function
    sam.eat()   // An implementation of an interface function
    sam.pay(10) // A default implementation in an interface
}
```

```
It's me, Sam.
Mixed salad, please!
*Eats Mixed salad*
Delicious! Here's 10 bucks!
```

Kotlin - Funcional

Kotlin

Solutions Docs Community Teach Play

2.1.10 JVM Program arguments

Copy link Share code Run

```kotlin
fun main() {
    // Who sent the most messages?
    val frequentSender = messages
        .groupBy(Message::sender)
        .maxByOrNull { (_, messages) -> messages.size }
        ?.key                                           // Get their names
    println(frequentSender) // [Ma]

    // Who are the senders?
    val senders = messages
        .asSequence()                                   // Make operations lazy (for a long call chain)
        .filter { it.body.isNotBlank() && !it.isRead }  // Use lambdas...
        .map(Message::sender)                           // ...or member references
        .distinct()
        .sorted()
        .toList()                                       // Convert sequence back to a list to get a result
    println(senders) // [Adam, Ma]
}

data class Message(                                     // Create a data class
    val sender: String,
    val body: String,
    val isRead: Boolean = false,                        // Provide a default value for the argument
)

val messages = listOf(                                  // Create a list
    Message("Ma", "Hey! Where are you?"),
    Message("Adam", "Everything going according to plan today?"),
    Message("Ma", "Please reply. I've lost you!"),
)
```

```
Ma
[Adam, Ma]
```

Kotlin - Ideal para Testes

Kotlin

Solutions Docs Community Teach Play

2.1.10 JUnit Copy link Share code Run

```kotlin
import org.junit.Test                   // Tests
import kotlin.test.*                    // The following example works for JVM only

class SampleTest {
    @Test
    fun `test sum`() {                  // Write test names with whitespaces in backticks
        val a = 1
        val b = 41
        assertEquals(42, sum(a, b), "Wrong result for sum($a, $b)")
    }

    @Test
    fun `test computation`() {
        assertTrue("Computation failed") {
            setup()                     // Use lambda returning the test subject
            compute()
        }
    }
}

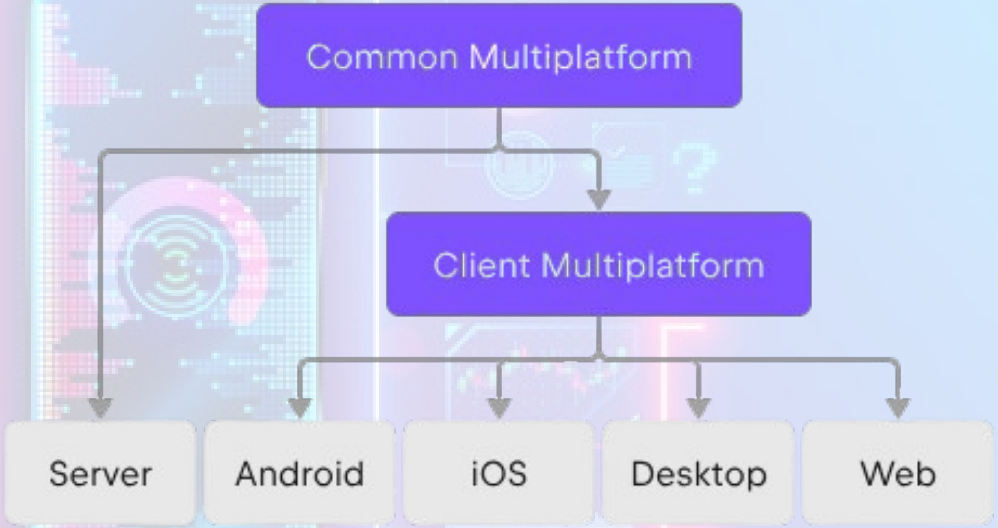
fun sum(a: Int, b: Int) = a + b         // Sources
fun setup() {}
fun compute() = true
```

Total test time: 0.036s

Passed: test sum

Passed: test computation

Kotlin - Multiplataforma



Kotlin - Sintaxe Básica

Variáveis: (**val** e **var**) Kotlin possui dois tipos principais de variáveis:

- **val** → imutável (constante, não pode ser alterada
- **var** → mutável (pode ser alterada depois de declarada)

```
val nome = "Kotlin"  // Imutável

var idade = 25       // Mutável

idade = 26           // Ok!
```

Kotlin - Sintaxe Básica

Tipos de Dados: Os principais tipos de dados em Kotlin são:

Tipo	Descrição	Exemplo
Int	Números inteiros	val numero: Int = 10
Double	Números decimais	val altura: Double = 1.75
Boolean	Verdadeiro ou falso	val ativo: Boolean = true
String	Texto	val nome: String = "João"
Char	Caracter único	val letra: Char = 'A'

Dica: Kotlin tem **inferência de tipo**, então você pode omitir o tipo se o valor for óbvio.

val idade = 30 // Kotlin sabe que é um Int

Kotlin - Sintaxe Básica

Operadores Básicos

Os operadores matemáticos e lógicos são parecidos com os do Java:

- **Aritméticos**: +, -, *, /, %
- **Comparação**: ==, !=, <, >, <=, >=
- **Lógicos**: && (E), | | (OU), ! (NÃO)

Exemplos:

```
val a = 10
val b = 5
println(a + b)


// 15 println(a > b)
// true println(a == b)
// false
```

Kotlin - Sintaxe Básica

Estruturas Condicionais (if, when)

if (comum e como expressão)

kotlin Copiar Editar

```kotlin
val idade = 18

if (idade >= 18) {
    println("Maior de idade")
} else {
    println("Menor de idade")
}

// if como expressão
val resultado = if (idade >= 18) "Maior" else "Menor"
println(resultado)
```

Kotlin - Sintaxe Básica

when (substitui switch do Java)

kotlin

Copiar Editar

```kotlin
val dia = 3

val nomeDoDia = when (dia) {
    1 -> "Domingo"
    2 -> "Segunda"
    3 -> "Terça"
    else -> "Outro dia"
}

println(nomeDoDia) // "Terça"
```

Kotlin - Sintaxe Básica

Estruturas de Repetição (for, while)

Loop for (com ranges e listas)

kotlin

Copiar Editar

```kotlin
for (i in 1..5) {
    println(i) // 1 2 3 4 5
}

// Percorrendo uma lista
val lista = listOf("Kotlin", "Java", "Python")
for (linguagem in lista) {
    println(linguagem)
}
```

Kotlin - Sintaxe Básica

Loop while e do-while

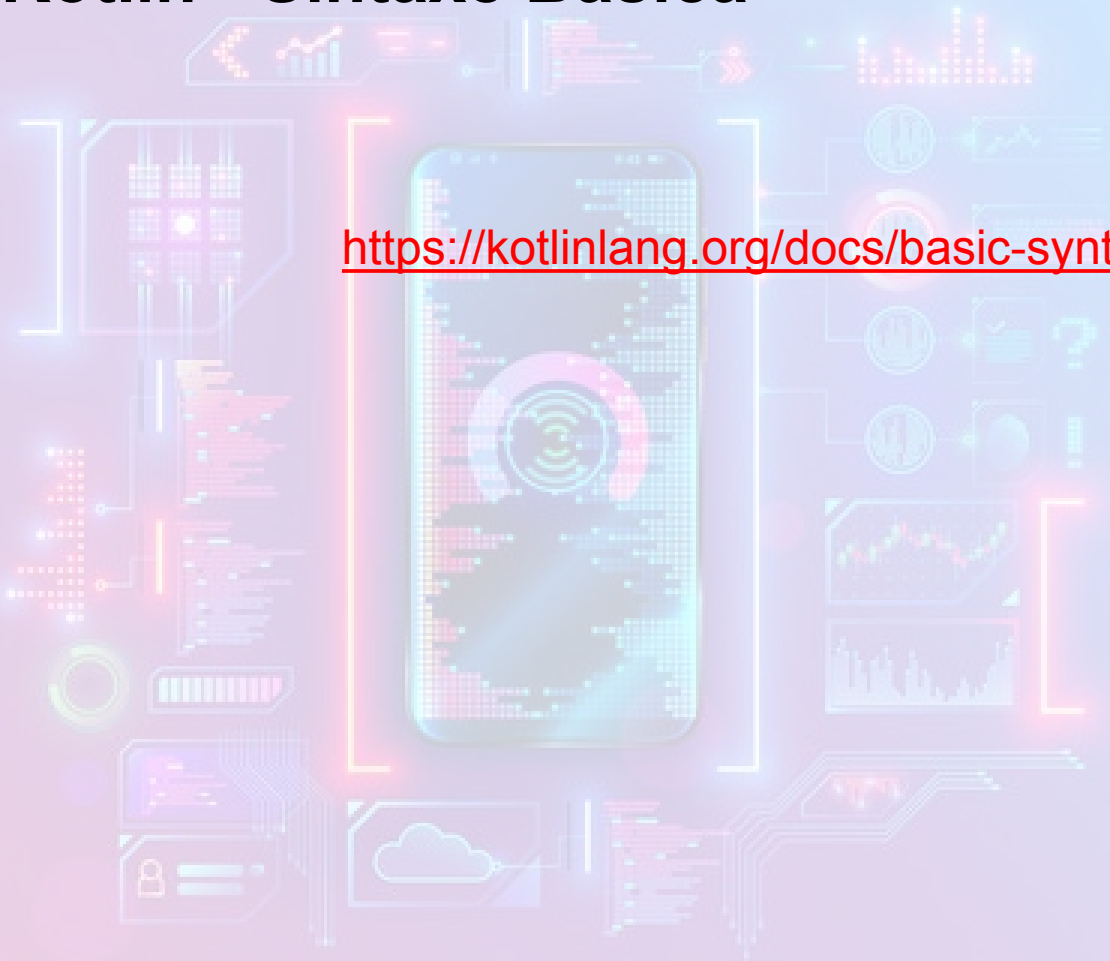
kotlin Copiar Editar

```kotlin
var contador = 3

while (contador > 0) {
    println("Contando: $contador")
    contador--
}

do {
    println("Executa pelo menos uma vez")
} while (contador > 0)
```

Kotlin - Sintaxe Básica



https://kotlinlang.org/docs/basic-syntax.html

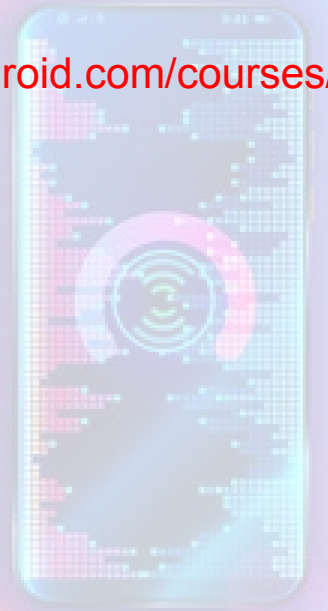


Kotlin no Android

UniFAP
CENTRO UNIVERSITÁRIO PARAÍSO

Kotlin no Android

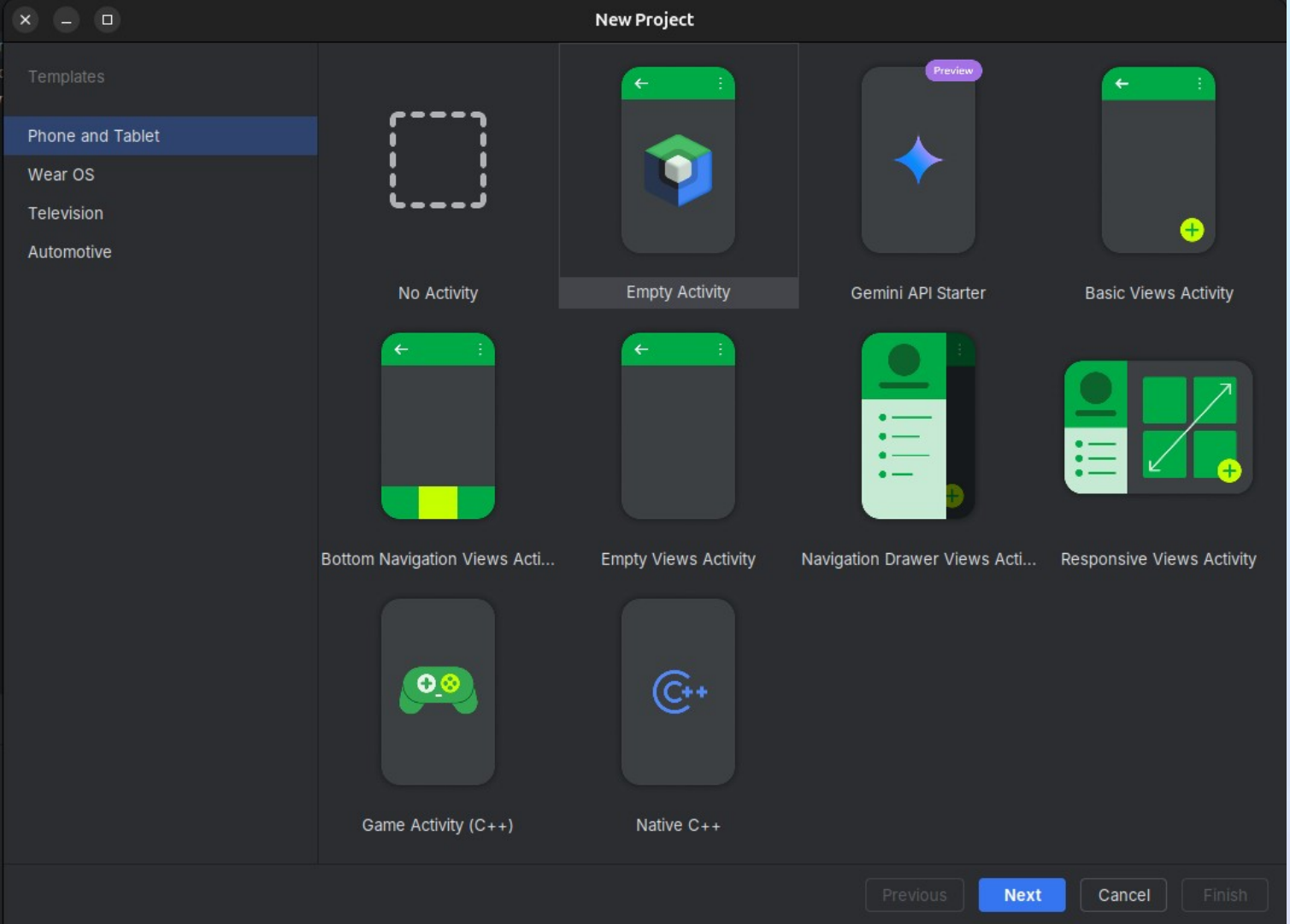
https://developer.android.com/courses/android-basics-compose/course?hl=pt-br





Crie um projeto usando o modelo

UniFAP
CENTRO UNIVERSITÁRIO PARAÍSO

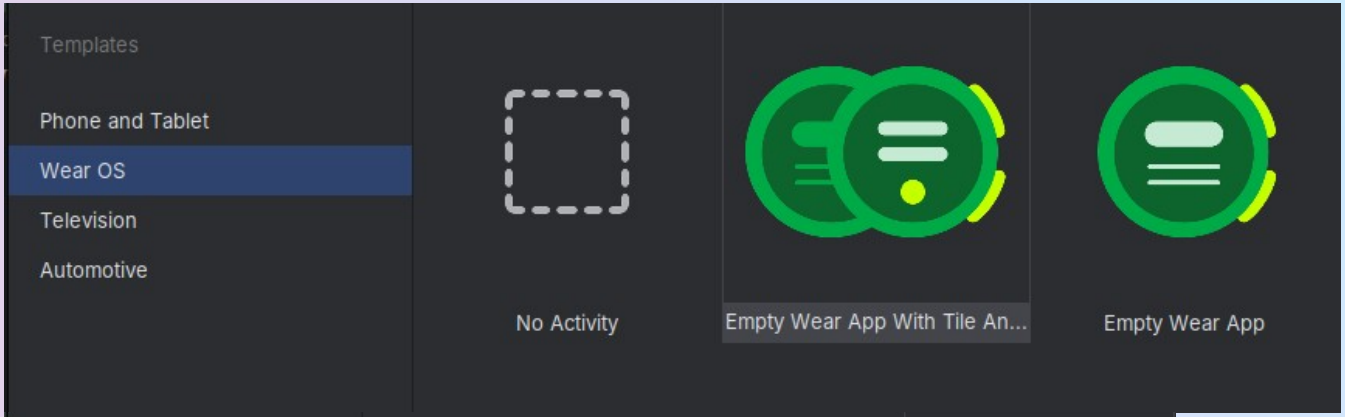


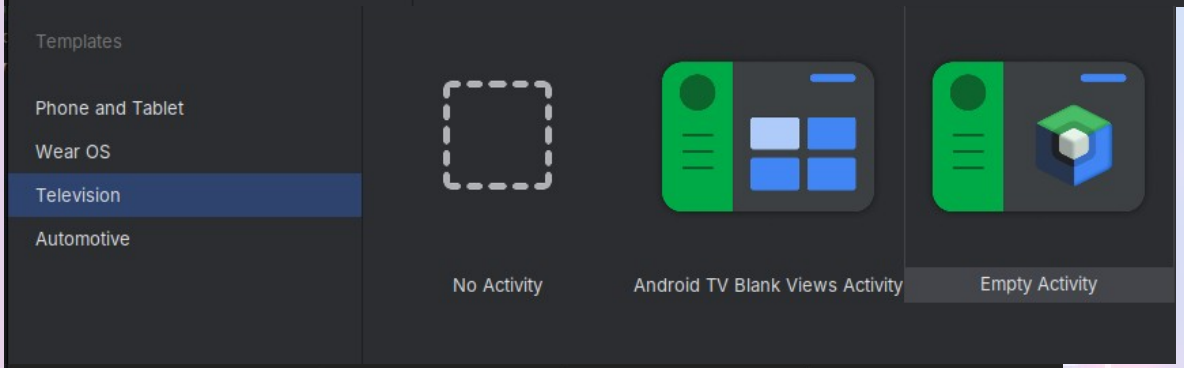
Crie um projeto usando o modelo

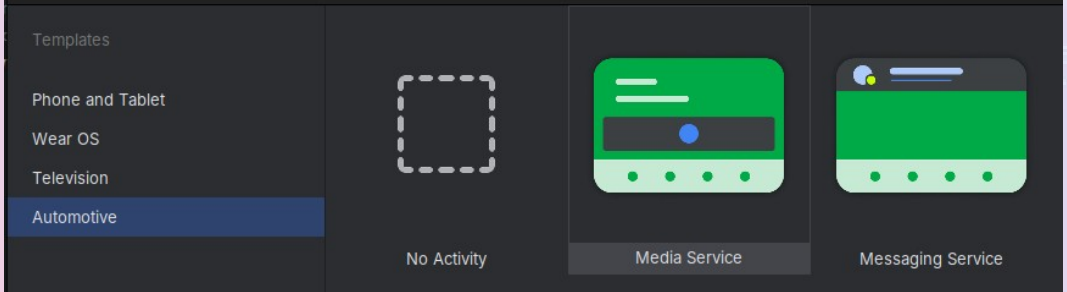
https://developer.android.com/codelabs/basic-android-kotlin-compose-first-app#1

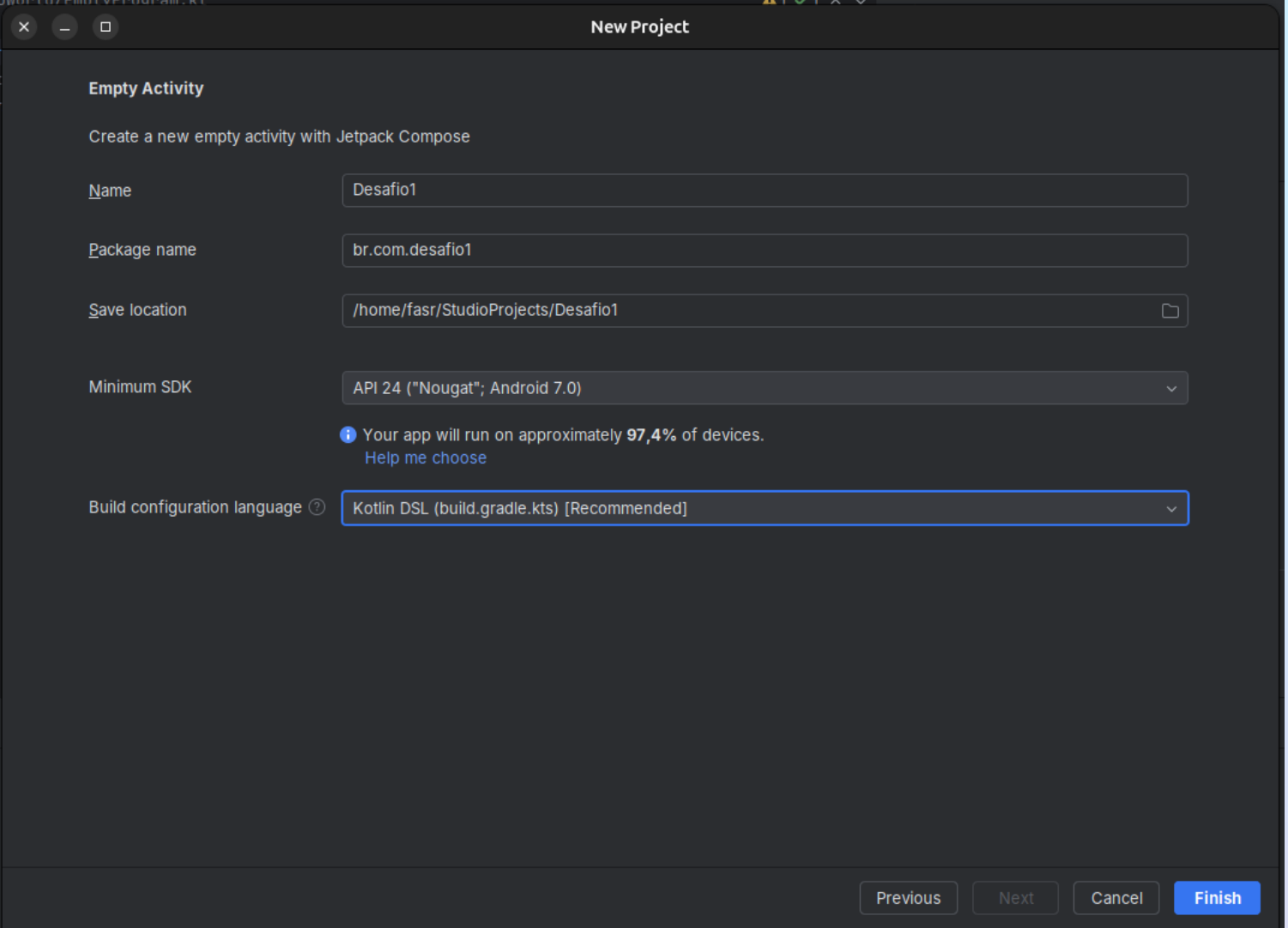
UniFAP
CENTRO UNIVERSITÁRIO PARAÍSO

Outros Modelos



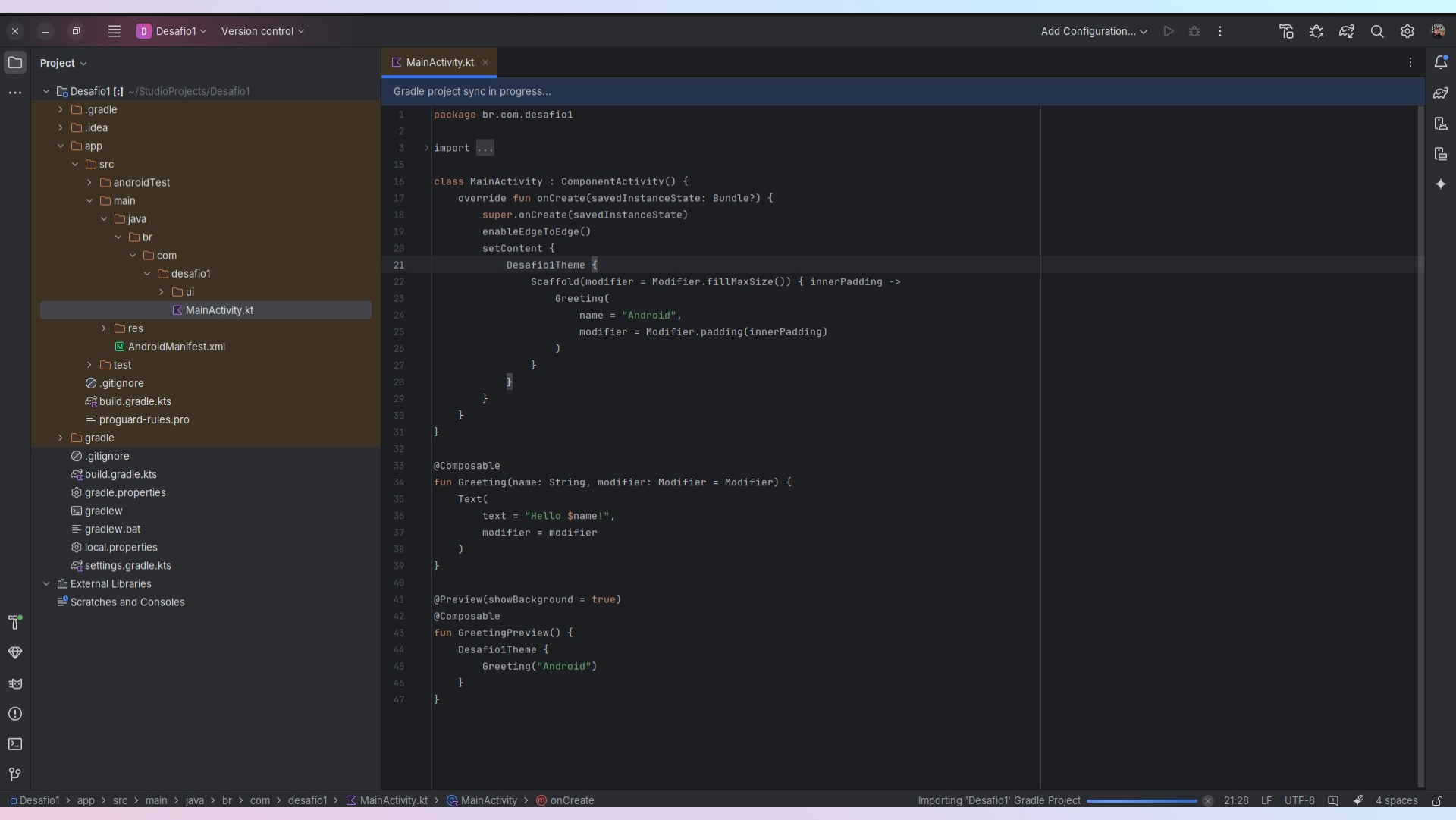


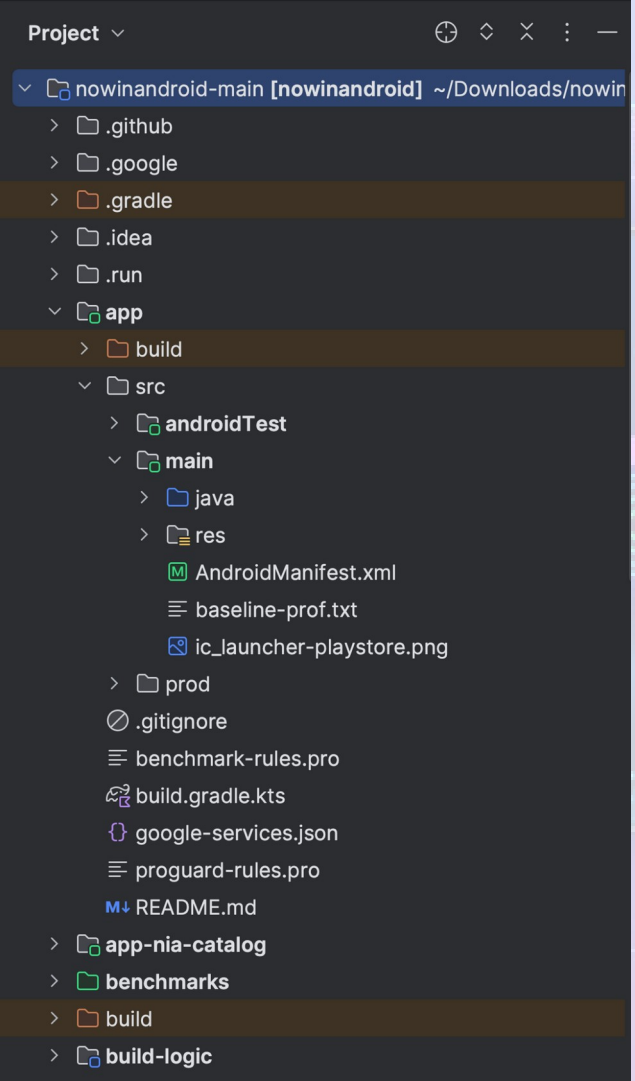




Crie um projeto usando o modelo

UniFAP
CENTRO UNIVERSITÁRIO PARAÍSO


Desafio1
Version control
Add Configuration...
Project
Desafio1 [:] ~/StudioProjects/Desafio1
.gradle
.idea
app
src
androidTest
main
java
br
com
desafio1
ui
MainActivity.kt
res
AndroidManifest.xml
test
.gitignore
build.gradle.kts
proguard-rules.pro
gradle
.gitignore
build.gradle.kts
gradle.properties
gradlew
gradlew.bat
local.properties
settings.gradle.kts
External Libraries
Scratches and Consoles
MainActivity.kt
Gradle project sync in progress...
package br.com.desafio1
import ...
class MainActivity : ComponentActivity() {
override fun onCreate(savedInstanceState: Bundle?) {
super.onCreate(savedInstanceState)
enableEdgeToEdge()
setContent {
Desafio1Theme {
Scaffold(modifier = Modifier.fillMaxSize()) { innerPadding ->
Greeting(
name = "Android",
modifier = Modifier.padding(innerPadding)
)
}
}
}
}
}
@Composable
fun Greeting(name: String, modifier: Modifier = Modifier) {
Text(
text = "Hello $name!",
modifier = modifier
)
}
@Preview(showBackground = true)
@Composable
fun GreetingPreview() {
Desafio1Theme {
Greeting("Android")
}
}
Desafio1 › app › src › main › java › br › com › desafio1 › MainActivity.kt › MainActivity › onCreate
Importing 'Desafio1' Gradle Project
21:28
LF
UTF-8
4 spaces



Estrutura de um projeto Android

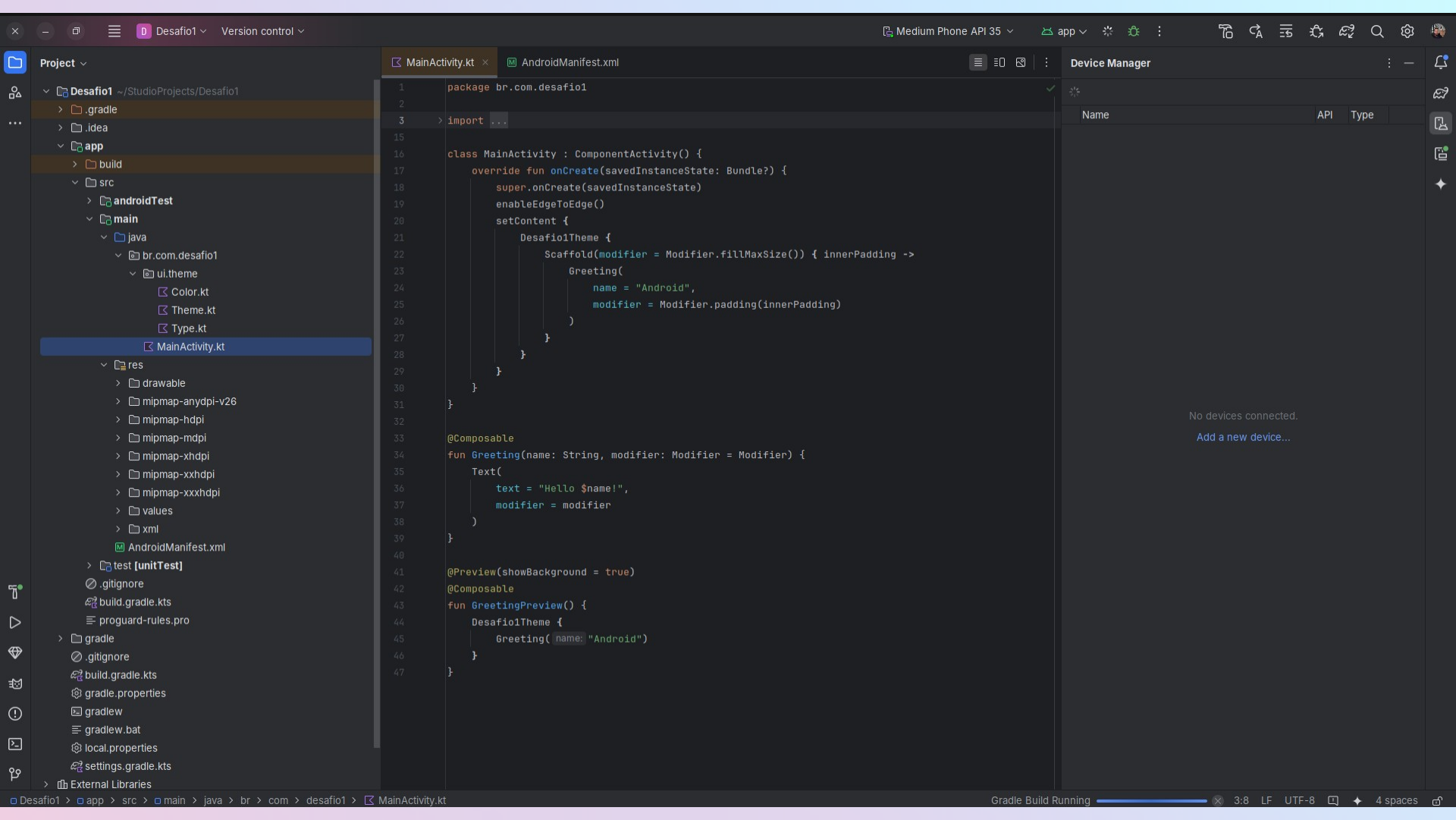
https://developer.android.com/studio/projects?hl=pt-br

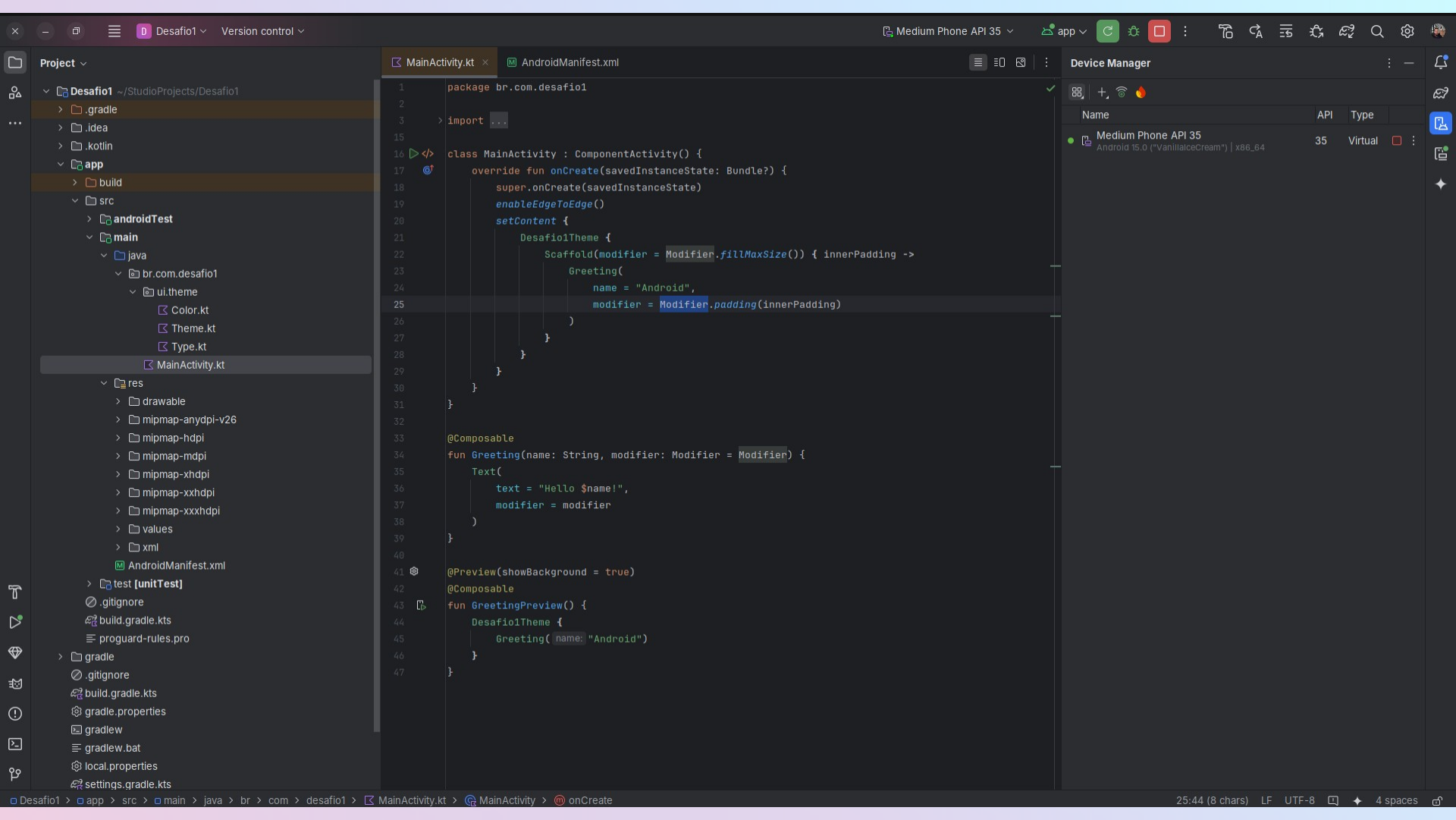
UniFAP
CENTRO UNIVERSITÁRIO PARAÍSO



Criar e gerenciar dispositivos virtuais

UniFAP
CENTRO UNIVERSITÁRIO PARAÍSO


Desafio1
Version control
Medium Phone API 35
app
Project
Desafio1 ~/StudioProjects/Desafio1
.gradle
.idea
app
build
src
androidTest
main
java
br.com.desafio1
ui.theme
Color.kt
Theme.kt
Type.kt
MainActivity.kt
res
drawable
mipmap-anydpi-v26
mipmap-hdpi
mipmap-mdpi
mipmap-xhdpi
mipmap-xxhdpi
mipmap-xxxhdpi
values
xml
AndroidManifest.xml
test [unitTest]
.gitignore
build.gradle.kts
proguard-rules.pro
gradle
.gitignore
build.gradle.kts
gradle.properties
gradlew
gradlew.bat
local.properties
settings.gradle.kts
External Libraries
MainActivity.kt
AndroidManifest.xml
package br.com.desafio1
import ...
class MainActivity : ComponentActivity() {
override fun onCreate(savedInstanceState: Bundle?) {
super.onCreate(savedInstanceState)
enableEdgeToEdge()
setContent {
Desafio1Theme {
Scaffold(modifier = Modifier.fillMaxSize()) { innerPadding ->
Greeting(
name = "Android",
modifier = Modifier.padding(innerPadding)
)
}
}
}
}
}
@Composable
fun Greeting(name: String, modifier: Modifier = Modifier) {
Text(
text = "Hello $name!",
modifier = modifier
)
}
@Preview(showBackground = true)
@Composable
fun GreetingPreview() {
Desafio1Theme {
Greeting(name: "Android")
}
}
Device Manager
Name
API
Type
No devices connected.
Add a new device...
Desafio1 › app › src › main › java › br › com › desafio1 › MainActivity.kt
Gradle Build Running
3:8
LF
UTF-8
4 spaces


Desafio1
Version control
Medium Phone API 35
app
Project
Desafio1 ~/StudioProjects/Desafio1
.gradle
.idea
.kotlin
app
build
src
androidTest
main
java
br.com.desafio1
ui.theme
Color.kt
Theme.kt
Type.kt
MainActivity.kt
res
drawable
mipmap-anydpi-v26
mipmap-hdpi
mipmap-mdpi
mipmap-xhdpi
mipmap-xxhdpi
mipmap-xxxhdpi
values
xml
AndroidManifest.xml
test [unitTest]
.gitignore
build.gradle.kts
proguard-rules.pro
gradle
.gitignore
build.gradle.kts
gradle.properties
gradlew
gradlew.bat
local.properties
settings.gradle.kts
MainActivity.kt
AndroidManifest.xml
1 package br.com.desafio1
2
3 import ...
15
16 class MainActivity : ComponentActivity() {
17 override fun onCreate(savedInstanceState: Bundle?) {
18 super.onCreate(savedInstanceState)
19 enableEdgeToEdge()
20 setContent {
21 Desafio1Theme {
22 Scaffold(modifier = Modifier.fillMaxSize()) { innerPadding ->
23 Greeting(
24 name = "Android",
25 modifier = Modifier.padding(innerPadding)
26)
27 }
28 }
29 }
30 }
31 }
32
33 @Composable
34 fun Greeting(name: String, modifier: Modifier = Modifier) {
35 Text(
36 text = "Hello $name!",
37 modifier = modifier
38)
39 }
40
41 @Preview(showBackground = true)
42 @Composable
43 fun GreetingPreview() {
44 Desafio1Theme {
45 Greeting(name: "Android")
46 }
47 }
Device Manager
Name
API
Type
Medium Phone API 35
Android 15.0 ("VanillaIceCream") | x86_64
35
Virtual
Desafio1 › app › src › main › java › br › com › desafio1 › MainActivity.kt › MainActivity › onCreate
25:44 (8 chars)
LF
UTF-8
4 spaces

Desafio1 Version control

Medium Phone API 35 app

Project

Desafio1 ~/StudioProjects/Desafio1
- .gradle
- .idea
- .kotlin
- app
  - build
  - src
    - androidTest
    - main
      - java
        - br.com.desafio1
          - ui.theme
            - Color.kt
            - Theme.kt
            - Type.kt
          - MainActivity.kt
      - res
        - drawable
        - mipmap-anydpi-v26
        - mipmap-hdpi
        - mipmap-mdpi
        - mipmap-xhdpi
        - mipmap-xxhdpi
        - mipmap-xxxhdpi
        - values
        - xml
      - AndroidManifest.xml
    - test [unitTest]
  - .gitignore
  - build.gradle.kts
  - proguard-rules.pro
- gradle
- .gitignore
- build.gradle.kts
- gradle.properties
- gradlew
- gradlew.bat
- local.properties
- settings.gradle.kts

MainActivity.kt | AndroidManifest.xml

```kotlin
package br.com.desafio1

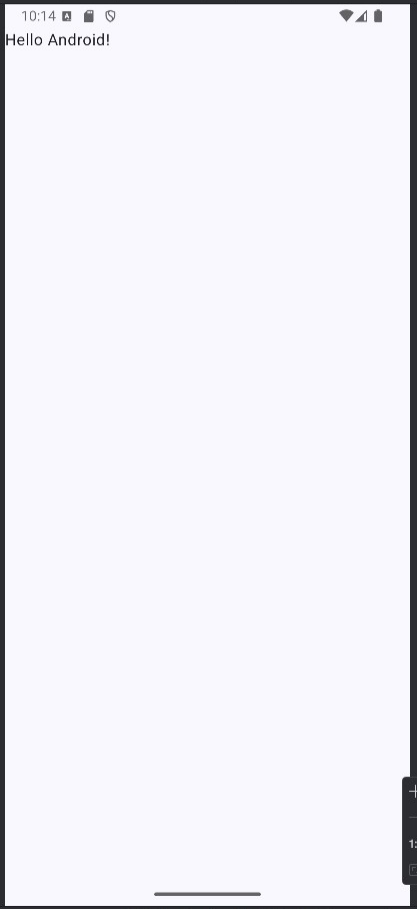
import ...

class MainActivity : ComponentActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        enableEdgeToEdge()
        setContent {
            Desafio1Theme {
                Scaffold(modifier = Modifier.fillMaxSize()) { innerPadding ->
                    Greeting(
                        name = "Android",
                        modifier = Modifier.padding(innerPadding)
                    )
                }
            }
        }
    }
}

@Composable
fun Greeting(name: String, modifier: Modifier = Modifier) {
    Text(
        text = "Hello $name!",
        modifier = modifier
    )
}

@Preview(showBackground = true)
@Composable
fun GreetingPreview() {
    Desafio1Theme {
        Greeting( name: "Android")
    }
}
```

Running Devices | Medium Phone API 35



Desafio1 › app › src › main › java › br › com › desafio1 › MainActivity.kt › MainActivity › onCreate

25:44 (8 chars) LF UTF-8 4 spaces



Configuração de Dispositivo Físico

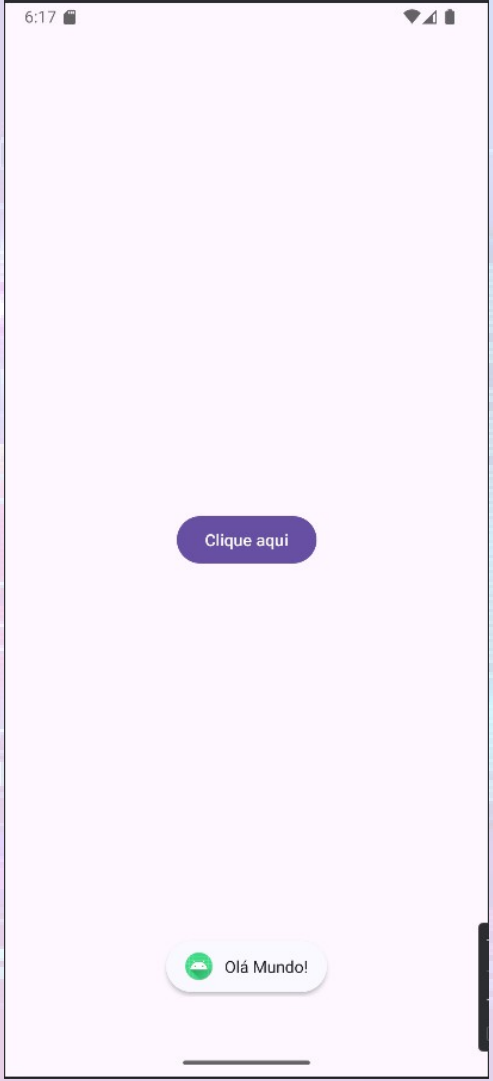
UniFAP
CENTRO UNIVERSITÁRIO PARAÍSO

Configuração de Dispositivo Físico

1. **Chrome OS**: nenhuma outra configuração é necessária.
2. **macOS**: nenhuma outra configuração é necessária.
3. **Windows**: instale um driver USB para o ADB.
4. **Ubuntu Linux**: precisa realizar configuração do plugdev.
5. Conectar ao dispositivo por **USB**
6. Conectar ao dispositivo por **Wi-Fi**
7. Espelhamento de dispositivos

https://developer.android.com/studio/run/device?hl=pt-br

UniFAP
CENTRO UNIVERSITÁRIO PARAÍSO



Desafio 1

Criar um aplicativo simples que exibe uma mensagem ao clicar em um botão.

```
var button: Button = findViewById(R.id.button);
button.setOnClickListener {
   Toast.makeText(this, "Olá Mundo!",
Toast.LENGTH_SHORT).show()
}
```

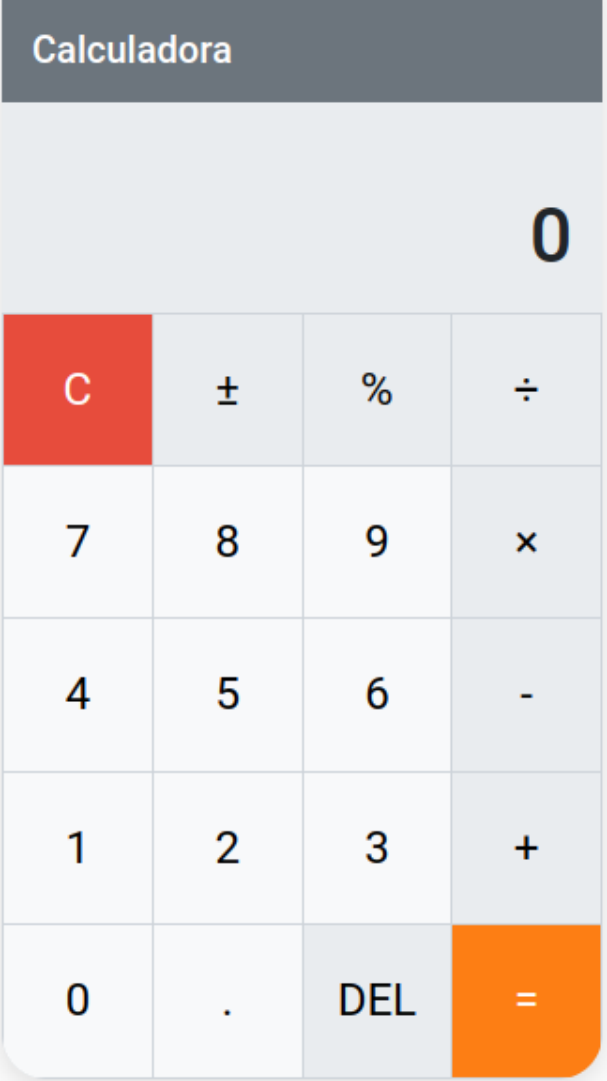


Desafio 2

Modifique o aplicativo para exibir um número que aumenta toda vez que o botão for clicado. O número deve ser exibido em um TextView.

Dicas:

1. Adicione um TextView no layout para exibir o contador.
2. Inicialize um contador (var count = 0).
3. Atualize o TextView dentro do setOnClickListener sempre que o botão for pressionado.

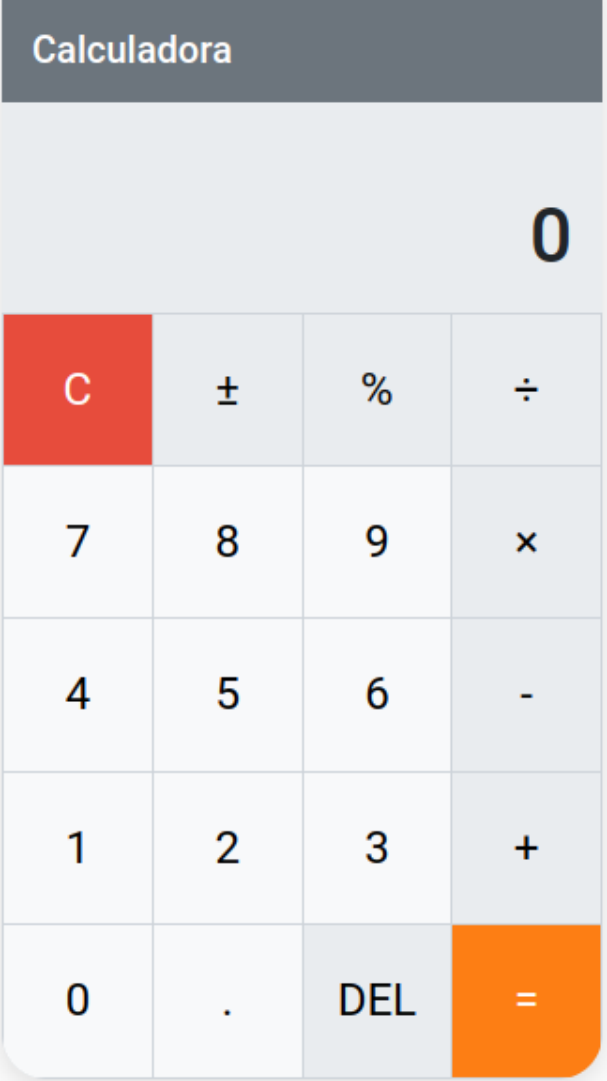


Desafio 3

Criar um aplicativo de um a calculadora simples que possa fazer as 4 operações aritméticas básicas, Soma, subtração, multiplicação e divisão.

Adicione **Botões (Button)** para os números (0-9) e operações (+, -, ×, ÷, =, C).

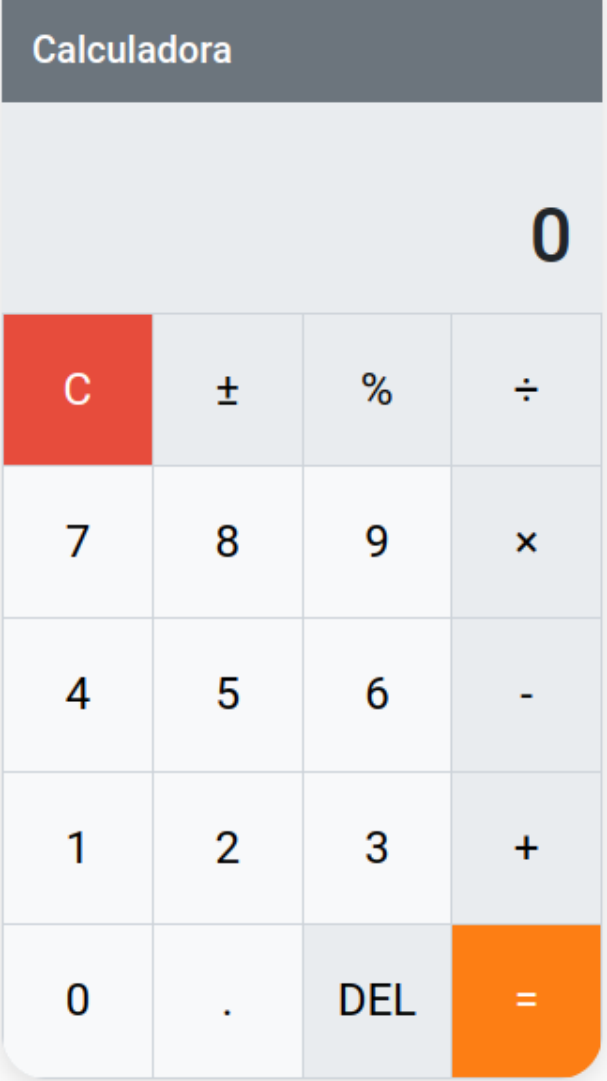
Use um **GridLayout** ou **ConstraintLayout** para organizar os botões.



Desafio 3

Implementar a Lógica da Calculadora

- No MainActivity.kt, crie variáveis para armazenar os números e a operação selecionada.
- Adicione **Listeners** (setOnClickListener) para os botões, de modo que os números e operações sejam processados corretamente.
- Ao pressionar =, exiba o resultado no TextView.
- Lembre-se de tratar **divisão por zero** para evitar erros.



Desafio 3

Testar no Emulador ou Dispositivo Físico

- Execute o app no **Emulador** do Android Studio ou em um celular Android.

Melhorias Opcionais

- Adicionar suporte a **números decimais**.
- Melhorar a interface com **Material Design**.
- Adicionar um botão **"limpar" (C)** para resetar os valores.

Bibliografia

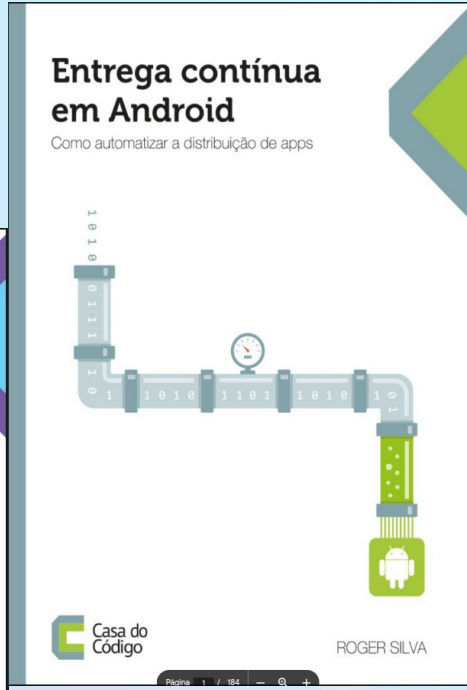
RESENDE, Kassiano. Kotlin com android: crie aplicativos de maneira fácil e divertida. São Paulo, SP: Casa do Código, 2018. E-book. Disponível em: https://plataforma.bvirtual.com.br. Acesso em: 26 fev. 2025.

https://plataforma.bvirtual.com.br/Acervo/Publicacao/213016

SILVA, Roger. Entrega contínua em android: como automatizar a distribuição de apps. São Paulo, SP: Casa do Código, 2016. E-book. Disponível em: https://plataforma.bvirtual.com.br. Acesso em: 26 fev. 2025.

https://plataforma.bvirtual.com.br/Acervo/Publicacao/212670

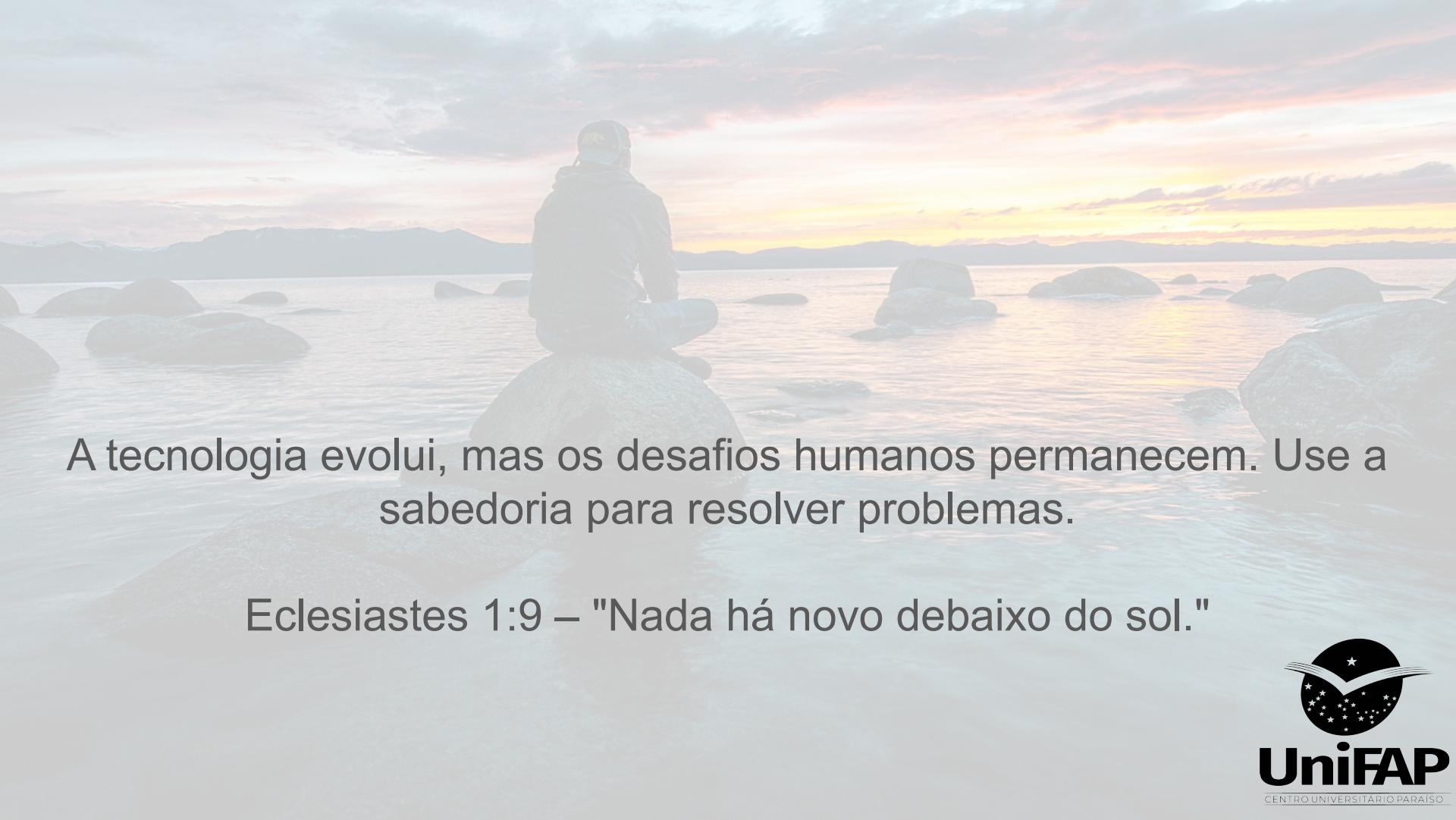




Perguntas?!



UniFAP
CENTRO UNIVERSITÁRIO PARAÍSO



A tecnologia evolui, mas os desafios humanos permanecem. Use a sabedoria para resolver problemas.

Eclesiastes 1:9 – "Nada há novo debaixo do sol."

UniFAP
CENTRO UNIVERSITÁRIO PARAÍSO



Obrigado!

UniFAP
CENTRO UNIVERSITÁRIO PARAÍSO