

FPFtech Escola Tecnológica





Desenvolvimento Mobile

Willon Ferreira



QUEM SOU EU?

2018 - UEA, Engenharia Civil

2021 - Desenvolvedor de Software na FPFtech

2024 - Ufam, Engenharia de Software

2025 - Professor na Escola Tecnológica da FPFtech



ACORDOS

Exercícios em todas as aulas (podem valer ponto ou não);

2 avaliações, 10 pontos cada;

Horário da aula:

08h10 às 11h40		Intervalo: 09h30		Chamada: 08h30 e 09h45;
18h10 às 21h30		Intervalo: 19h30		Chamada: 18h10 e 19h45;

Dúvidas são bem-vindas;

Esforce-se para aprender além da aula e não somente depender do professor;

Respeito por quem quer aprender;



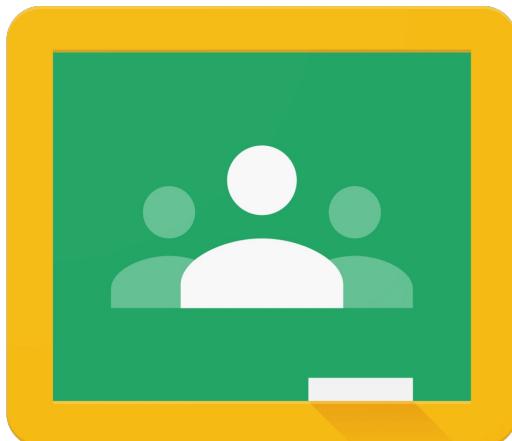
ACORDOS



ACORDOS

Canais de comunicação oficiais:

E-mail: willon.ferreira@fpf-etech.com



PLATAFORMA DE EXERCÍCIOS



beecrowd

codewars 

Achieve mastery through challenge.



LINKS DE APOIO



LINKS DE APOIO

Kotlin in Action - Editora Manning

manning.com/books/kotlin-in-action

Kotlin Docs

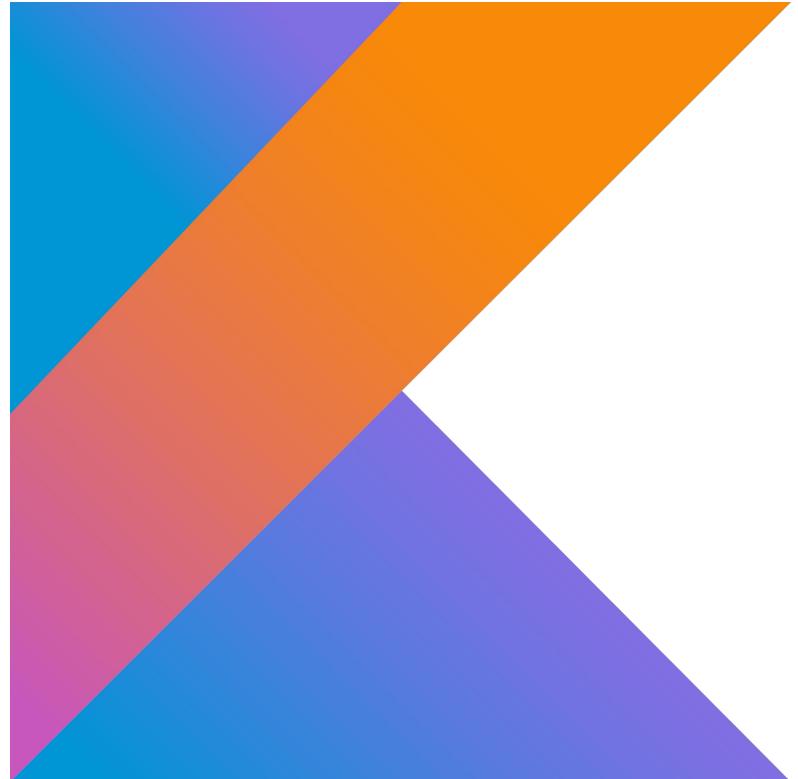
kotlinlang.org/docs/home.html

Kotlin Playground

play.kotlinlang.org/



O QUE VOCÊ
ESPERA DESSA
DISCIPLINA?





INTRODUÇÃO



in Brasil

Vagas | Data do anúncio | Nível de experiência | Empresa | Remoto | Candidatura simplificada | Todos os filtros

Consiga emprego mais rápido. Experimente Premium grátis.

kotlin em: Brasil 1.488 resultados Configurar alerta

BRQ Digital Solutions

Profissional Android 

São Paulo, São Paulo, Brasil · há 1 semana · 49 candidaturas

Híbrido · Contrato · Pleno-sênior

Competências: Java, C#, e mais 5

Veja como você se compara a 49 candidatos. Experimente Premium novamente por [BRLO](#)

[Candidatura simplificada](#) [Salvar](#)

Sobre a vaga

Código da vaga: [41562](#)

Sobre a BRQ Digital

Há 31 anos no mercado, a BRQ Digital Solutions se consolidou como uma das maiores empresas de transformação digital do país. Com uma plataforma de serviços end to end, oferecemos as mais eficientes e inovadoras soluções, tecnologias e metodologias, promovendo uma jornada de transformação para grandes marcas, de diferentes segmentos, no Brasil e no exterior.

Temos 3 mil funcionários e a empresa é destaque como um dos melhores lugares para trabalhar pelo GPTW e Glassdoor.

BRQ WAY

No nosso BRQ WAY temos a missão de conectar, transformar e empoderar cada um dos nossos profissionais, que por aqui são chamados de feras, para que todos tenham a chance de usar sua paixão para transformar o mundo com tecnologia!

É isso que queremos, com nosso Programa de Diversidade e Inclusão trabalhamos com dois objetivos: equidade e inclusão, buscamos aumentar a participação dos grupos minoritários, quebrar os vieses inconscientes e expandir a conscientização sobre temas tão importantes para que todos sejam livres para serem quem são.

foursys

Desenvolvedor(a) IONIC 

Brasil (Remoto)

2 ex-funcionários trabalham aqui

Visualizado · Promovida · [Candidatura simplificada](#)

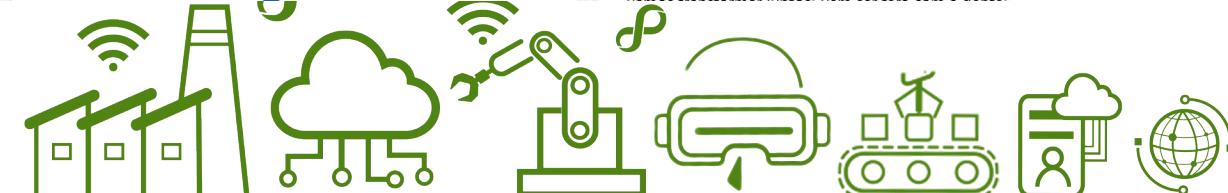
Sil Sistemas

Gerente de Projetos Júnior

Sil Sistemas

Caxias do Sul, Rio Grande do Sul, Brasil (Presencial)

A avaliação da candidatura geralmente leva 4 dias



in Estados Unidos

Vagas | Data do anúncio | Nível de experiência | Salário | Empresa | Remoto | Todos os filtros

Consiga emprego mais rápido. Experimente Premium grátis.

kotlin em: Estados Unidos 1.656 resultados Configurar alerta

Jobot

Engineering Manager (Startup + Hands-On) 

Oklahoma City, OK · Anunciada novamente há 1 semana · 48 candidaturas

US\$ 180K/a - US\$ 210K/a · Remoto · Tempo integral

Competências: Desenvolvimento de software, Ciência da computação, e mais 8

Veja como você se compara a 48 candidatos. Experimente Premium novamente por [BRLO](#)

[Candidatura simplificada](#) [Salvar](#)

Sobre a vaga

Want to learn more about this role and Jobot? Click our Jobot logo and follow our LinkedIn page!

Job details

Fully Remote, well-funded SaaS startup. Excellent Benefits, Unlimited PTO, Room for Growth!

This Jobot Job is hosted by Grant Greenhalgh

Are you a fit? Easy Apply now by clicking the "Easy Apply" button and sending us your resume.

Salary \$180,000 - \$210,000 per year

A Bit About Us

We're an established SaaS startup with a Platform fueled by our AI solution. This ML technology can automate, optimize and revolutionize critical business needs, and the versatility that our platform provides is the backbone of why we've been so successful since

Spotify

Backend Engineer II - Ads Data Platform 

Nova York, NY

Back-End +1 · Kotlin +7 · Plano de pensão, Plano de saúde

Há 1 dia

Pomelo Care

Software Engineer (all levels) 

Estados Unidos (Remoto)

US\$ 135K/a - US\$ 230K/a

Há 4 meses

Dexcom

Intern II - Software Developer Engineering 

Estados Unidos (Remoto)

Celular · Kotlin +1

Há 6 dias

'GLASSDOOR'

Salários de Mobile Developer

Salários Entrevistas

Salários de Mobile Developer (Brasil)



Confiança muito alta · 318 salários enviados · Atualizado em 7 de jan. de 2025

Experiência

Todos os anos de experiência



Salário base

R\$ 4 mil - R\$ 9 mil/mês

R\$ 6 mil/mês Salário base médio

Remuneração variável 

R\$ 367/mês Média

R\$ 146 - R\$ 2 mil/mês Faixa

A remuneração total mensal estimada para o cargo de Mobile Developer é de R\$ 6.827, com uma média salarial mensal de R\$ 6.460. Esses números representam a mediana, que é o ponto médio dos intervalos do nosso modelo proprietário de Estimativa de Pagamento Total e é baseado nos salários coletados de nossos usuários. A remuneração variável mensal estimada é de R\$ 367, que pode incluir bônus, comissões, gorjetas e participações nos lucros.



O QUE É DESENVOLVIMENTO MÓVEL?

Criação de aplicativos para dispositivos móveis como smartphones ou tablets.



O QUE É DESENVOLVIMENTO MÓVEL?

Primeiros dispositivos móveis:

Limitados a chamadas e mensagens de texto.

Avanço tecnológico -> smartphones

Aplicativos complexos;

Mais recursos para criação de soluções;



PRINCIPAIS PLATAFORMAS

Android (Google)

- Sistema operacional *open-source* e personalizável (AOSP);
- Grande variedade de dispositivos e fabricantes.

iOS (Apple)

- Sistema mais fechado com controle rigoroso de qualidade.
- Foco em desempenho e experiência do usuário.



CRESCIMENTO DO MERCADO DE APPS

- Milhões de aplicativos disponíveis nas lojas (Google Play Store e Apple App Store).

2020

Quanto Total de
Downloads Globais

- Setores como jogos, educação, saúde e comércio eletrônico dominam o mercado.



CRESCIMENTO DO MERCADO DE APPS

- No cotidiano geral é a primeira coisa a se pensar na busca de desenvolvimento de uma solução;



- “Solução na palma da mão”

Existem mais pessoas que sabem mexer em um celular do que em um computador;



IMPORTÂNCIA DO DESENVOLVIMENTO MÓVEL

Conectividade Constante

- Usuários passam a maior parte do tempo conectados via dispositivos móveis.
- Necessidade de acesso imediato a informações e serviços.



IMPORTÂNCIA DO DESENVOLVIMENTO MÓVEL

Oportunidade de Negócio

- Empresas utilizam aplicativos para melhorar a relação com clientes;
- Startups focadas em soluções mobile crescem rapidamente.



Linguagens de programação



Kotlin: Desenvolvimento nativo Android

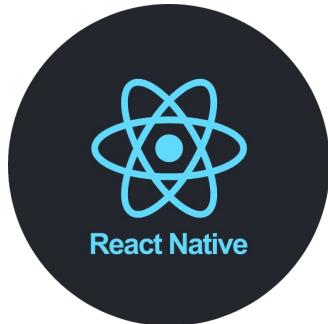
Quanto mais
aprendemos

Swift: Desenvolvimento nativo iOS

Javascript: Desenvolvimento híbrido e
webapps



Frameworks de desenvolvimento



Desenvolvimento nativo usando Javascript;
Multiplataforma;

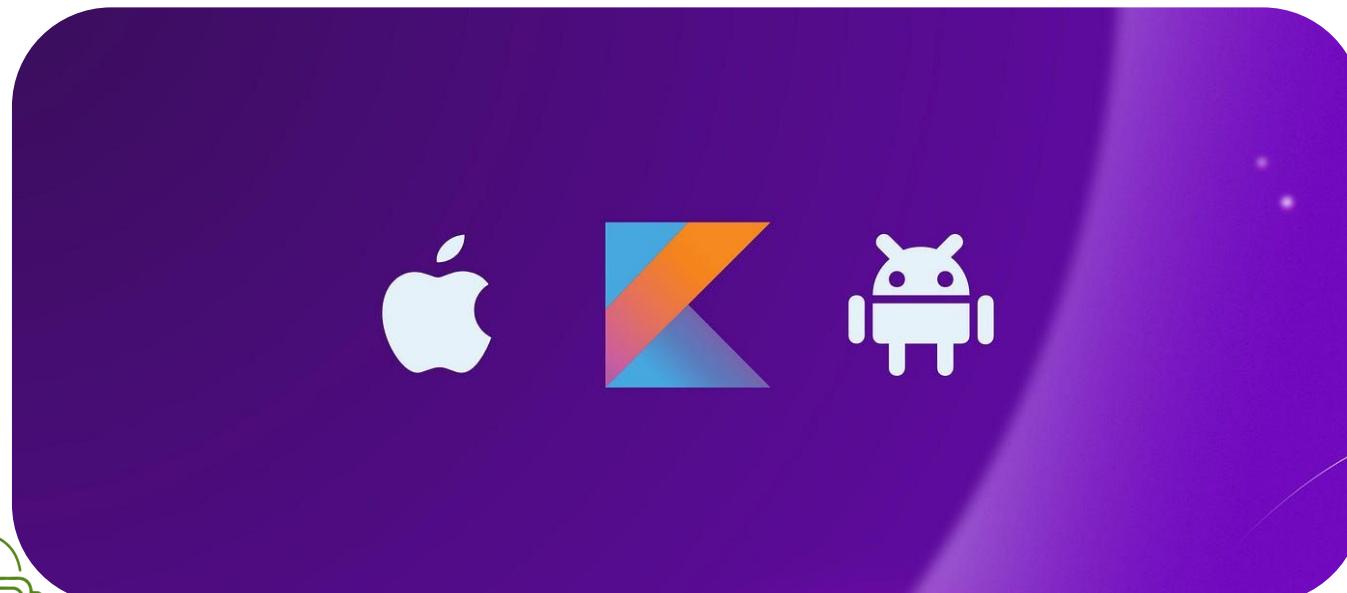


Desenvolvimento nativo usando Dart;
Alto desempenho e widgets personalizáveis;
Multiplataforma;



KMM - Kotlin Multiplatform Mobile (KMM)

Kotlin Multiplatform Mobile (KMM) é um SDK desenvolvido pela JetBrains que simplifica a criação de aplicativos móveis para **Android** e **iOS** com uma base de código compartilhada.



KMM - Kotlin Multiplatform Mobile (KMM)

Permite que você escreva a lógica de negócios, como manipulação de dados, chamadas de rede e operações de armazenamento, em Kotlin comum, enquanto a interface do usuário permanece nativa para cada plataforma.



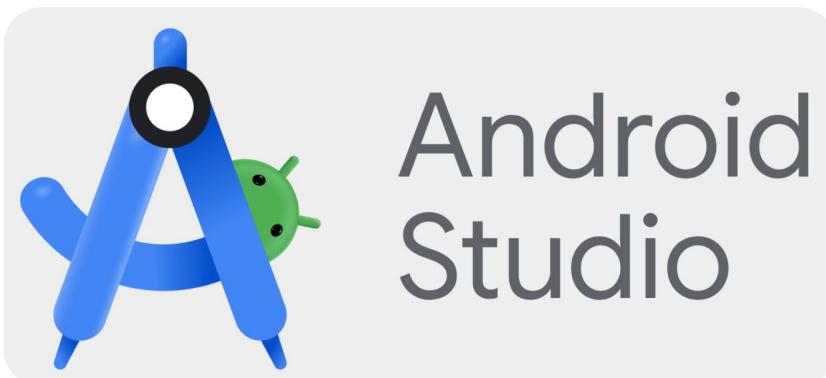
KMM - Kotlin Multiplatform Mobile (KMM)

Código Compartilhado: Você desenvolve a lógica de negócios em Kotlin, que é compartilhada entre as plataformas.

Código Específico da Plataforma: As partes específicas, como a interface do usuário e APIs nativas, são implementadas nas linguagens nativas — **Kotlin/Java** para Android e **Swift/Objective-C** para iOS.



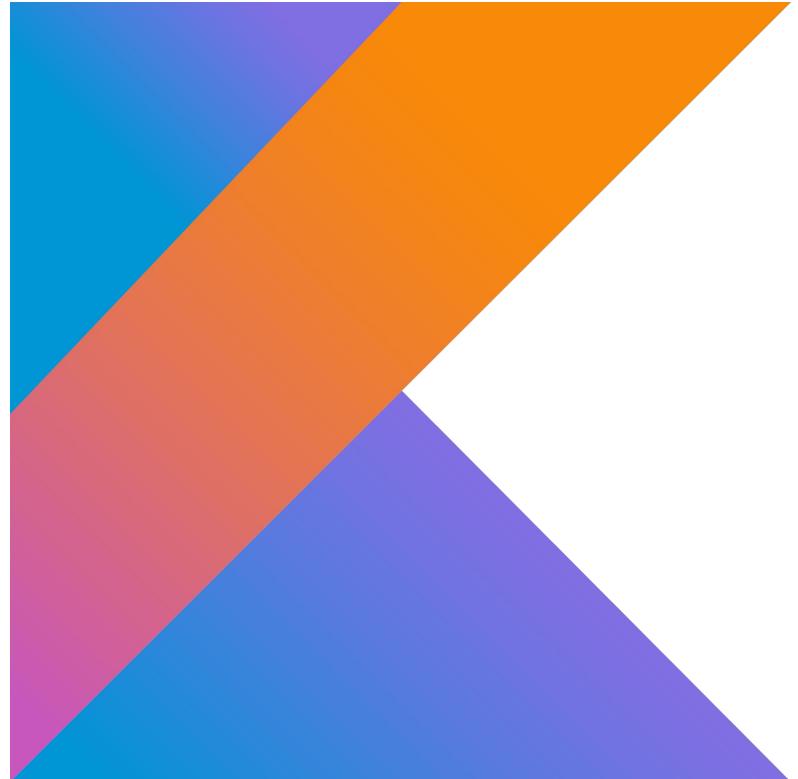
O QUE VAMOS USAR?



Jetpack Compose

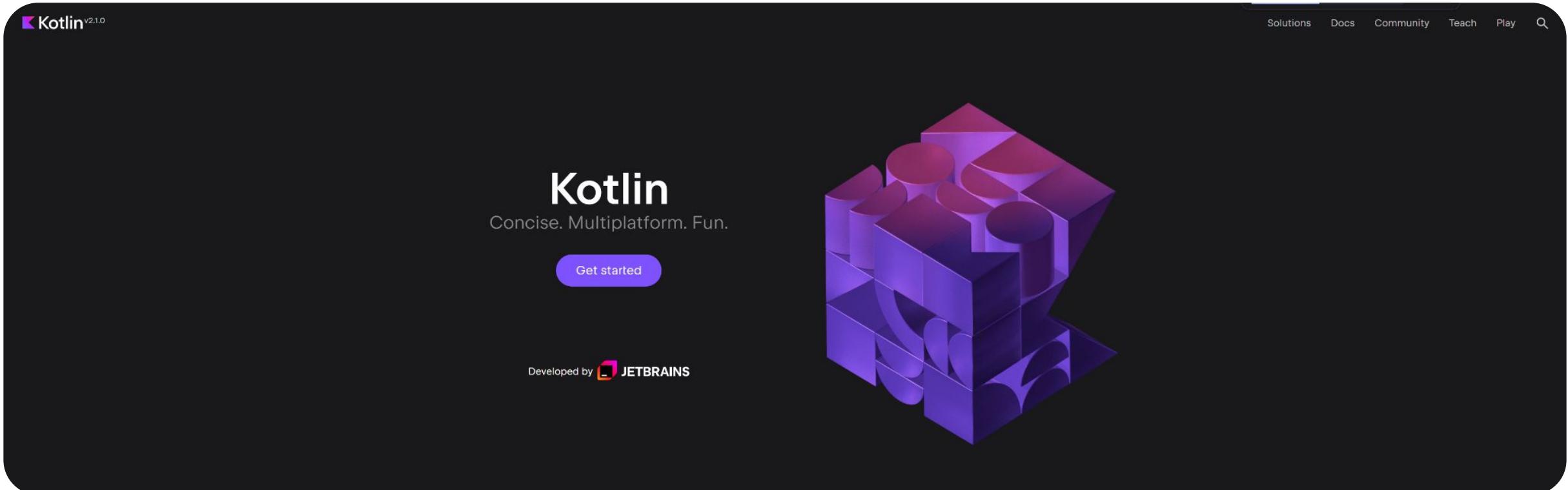


Ambiente de desenvolvimento



AMBIENTE DE DESENVOLVIMENTO

Instalação



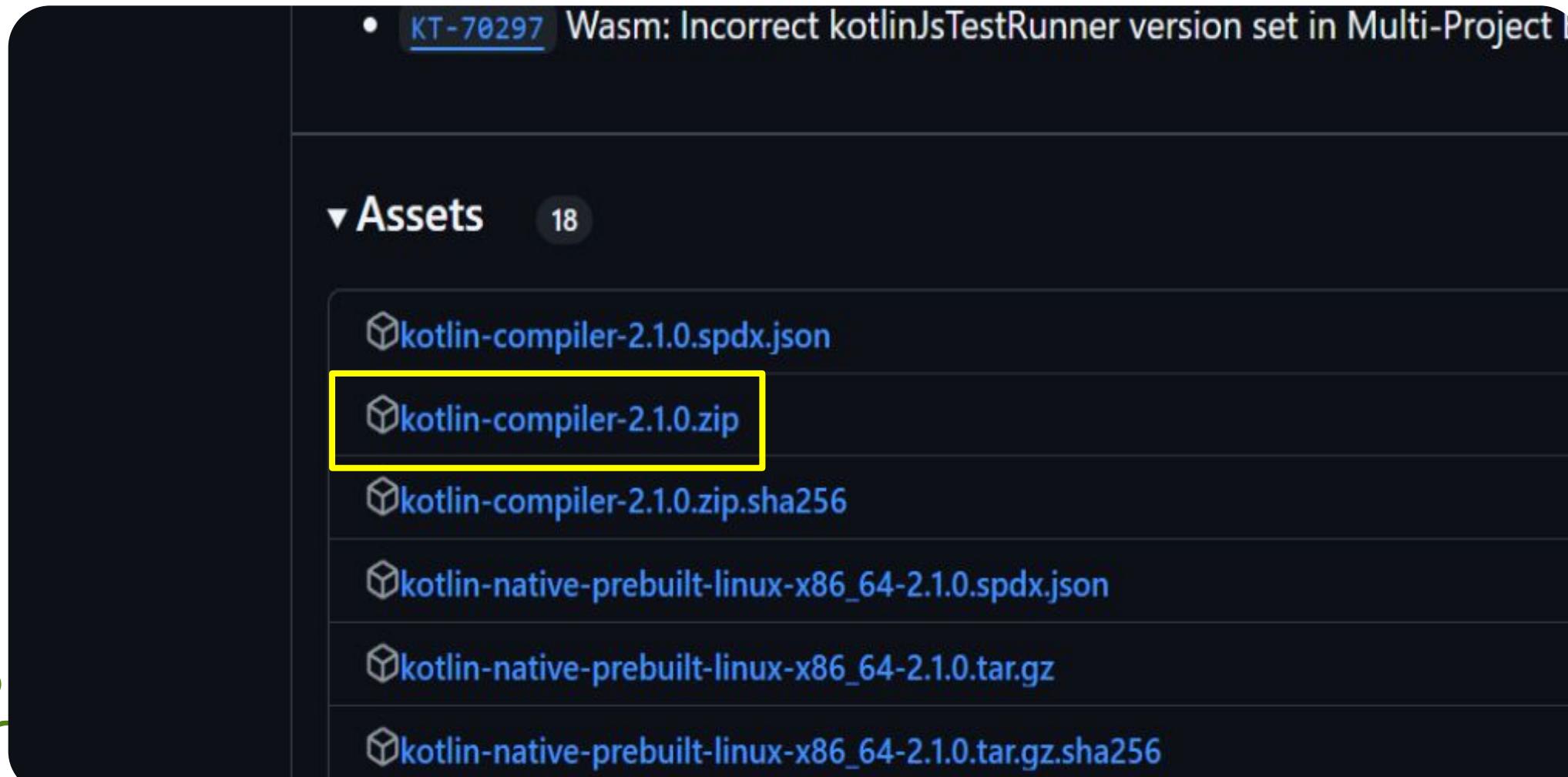
Link: <https://kotlinlang.org/docs/command-line.html>



AMBIENTE DE DESENVOLVIMENTO

Instalação

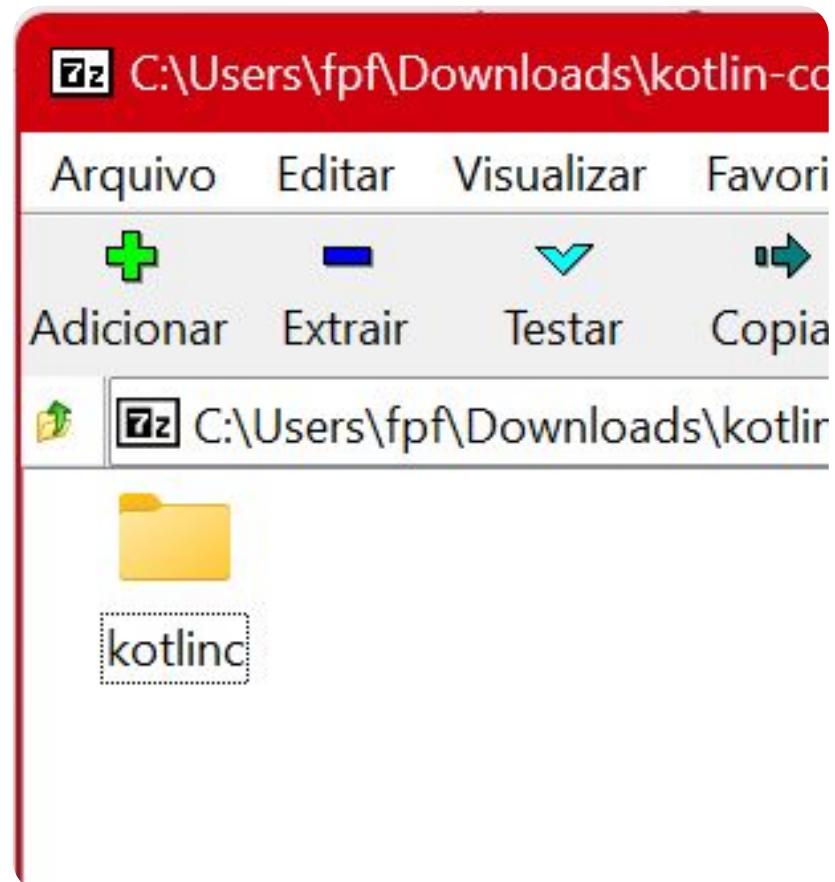
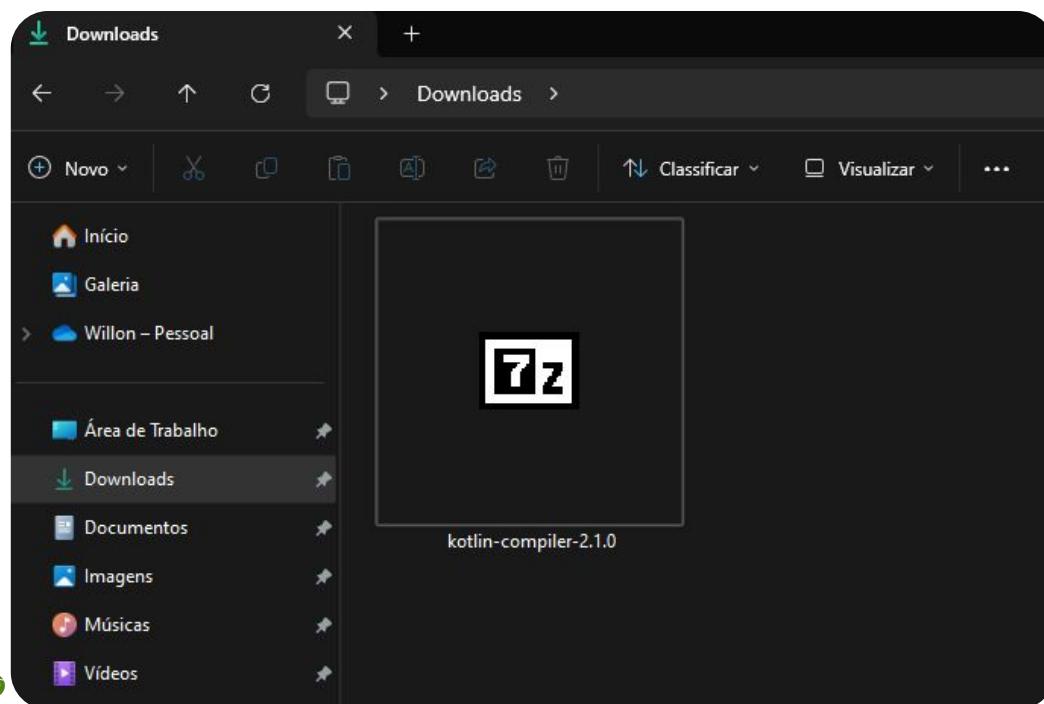
Baixar arquivo .zip com os binários do compilador Kotlin



AMBIENTE DE DESENVOLVIMENTO

Instalação

Baixar arquivo .zip com os binários do compilador Kotlin



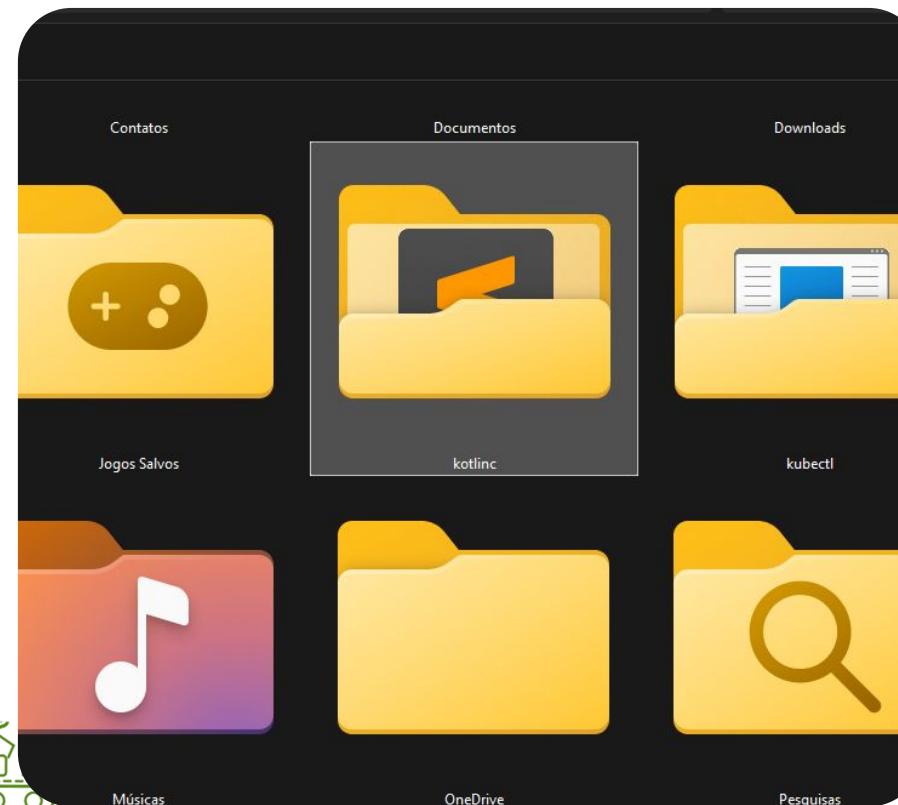
AMBIENTE DE DESENVOLVIMENTO

Instalação

Copiar conteúdo do arquivo .zip para a pasta raiz do S.O

Windows -> Pasta do usuário

Linux -> Pasta *home*

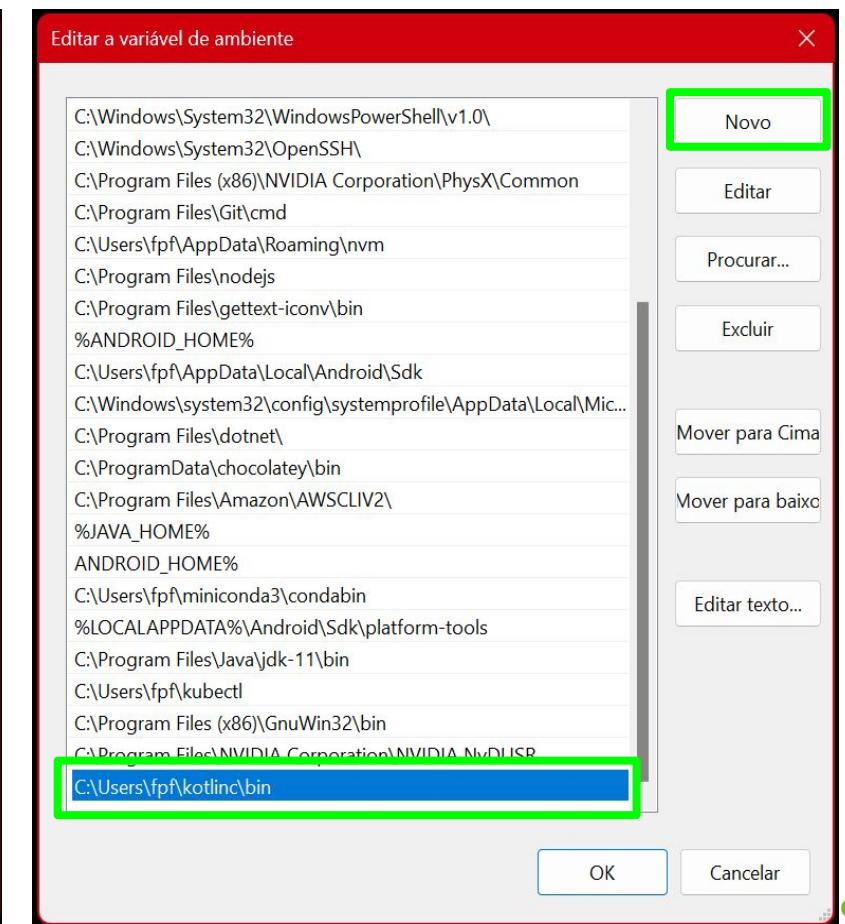
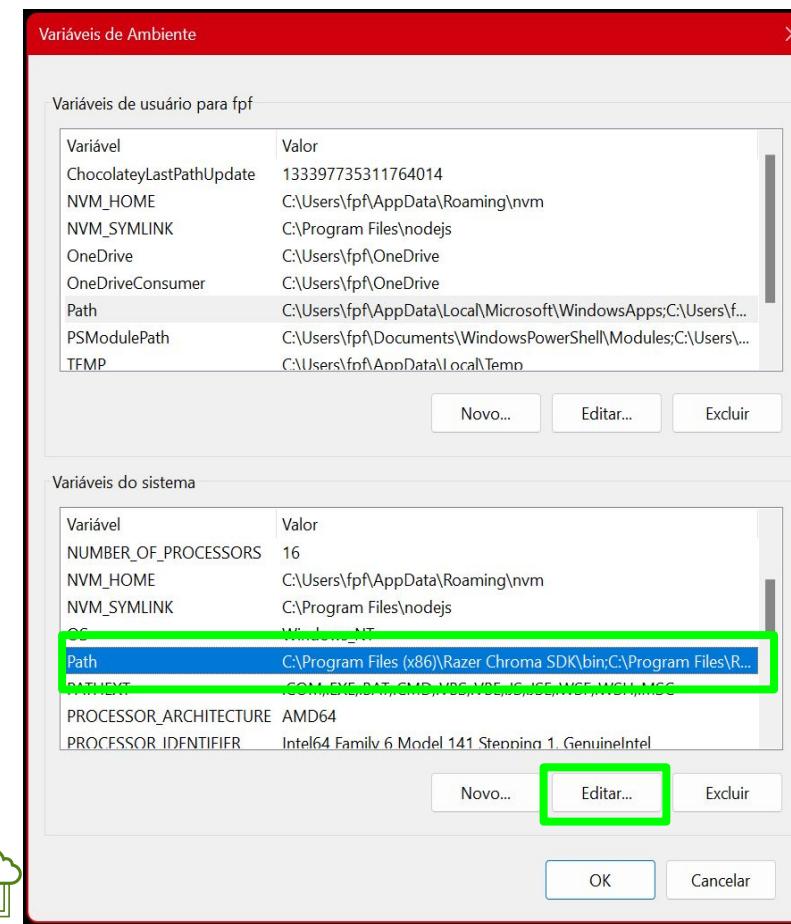
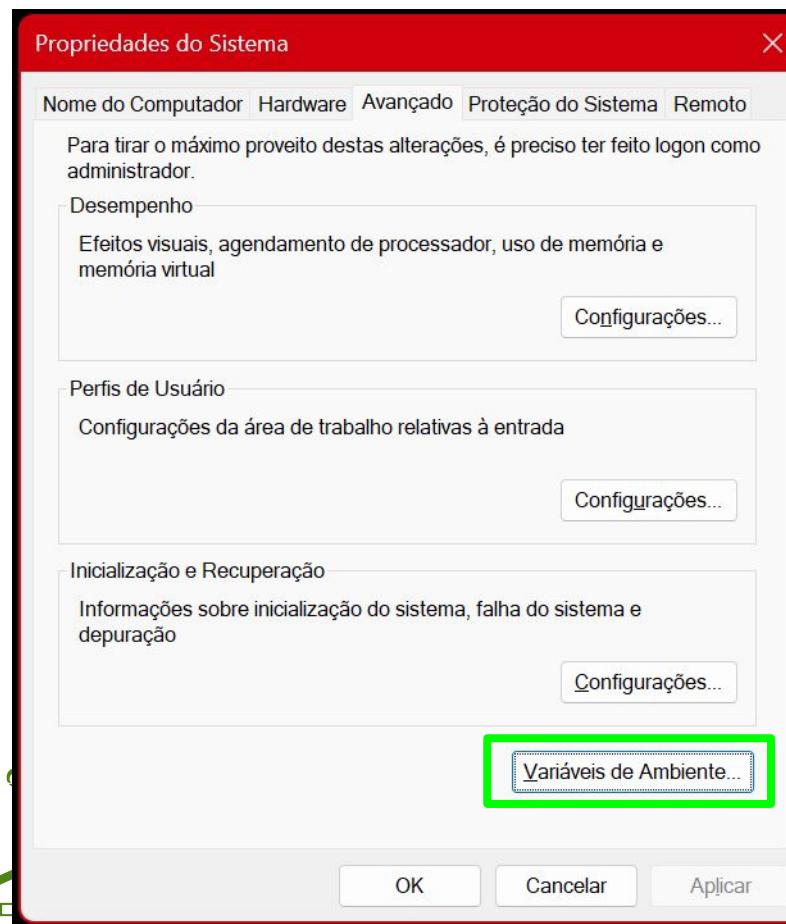


AMBIENTE DE DESENVOLVIMENTO

Instalação

Windows: Adicionar o caminho do compilador nas variáveis de ambiente

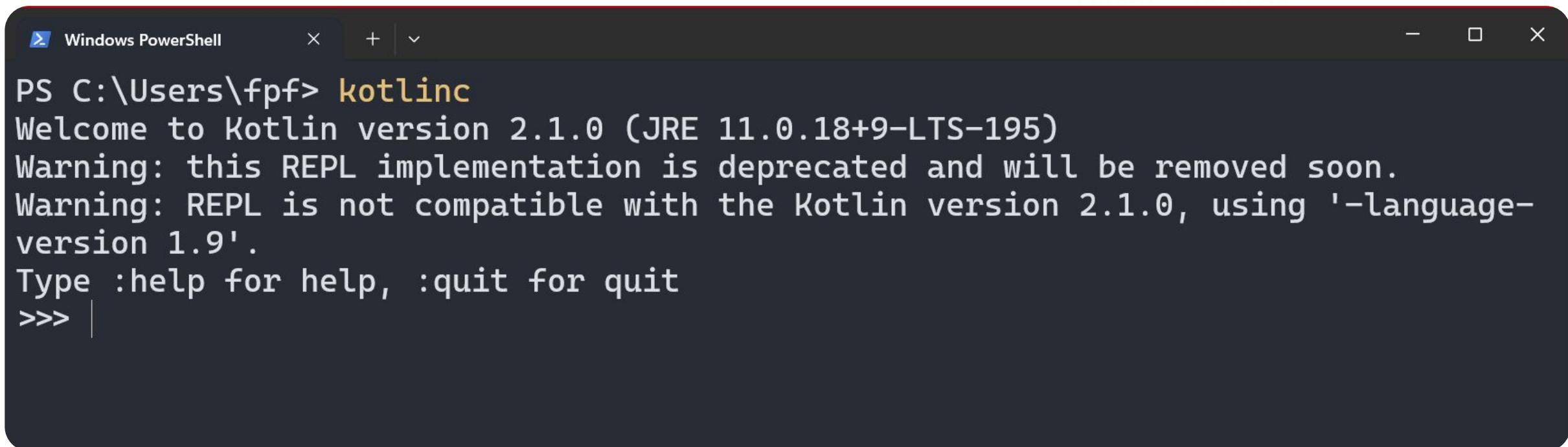
Linux: adicionar ao *path* do sistema



AMBIENTE DE DESENVOLVIMENTO

Instalação Windows

Abra o terminal e verifique se a instalação foi bem sucedida



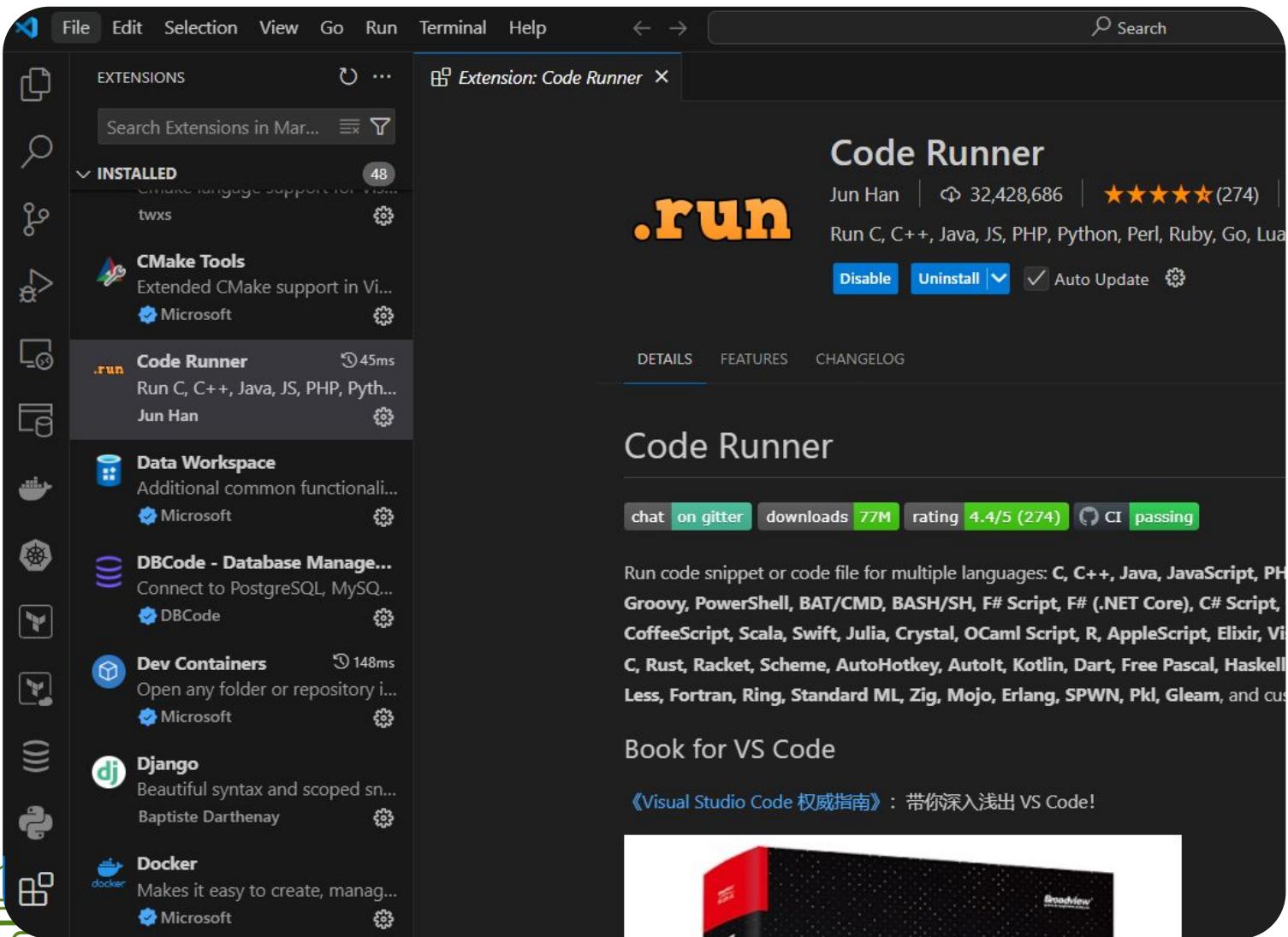
A screenshot of a Windows PowerShell window titled "Windows PowerShell". The window shows the following text output:

```
PS C:\Users\fpf> kotlinc
Welcome to Kotlin version 2.1.0 (JRE 11.0.18+9-LTS-195)
Warning: this REPL implementation is deprecated and will be removed soon.
Warning: REPL is not compatible with the Kotlin version 2.1.0, using '-language-version 1.9'.
Type :help for help, :quit for quit
>>> |
```



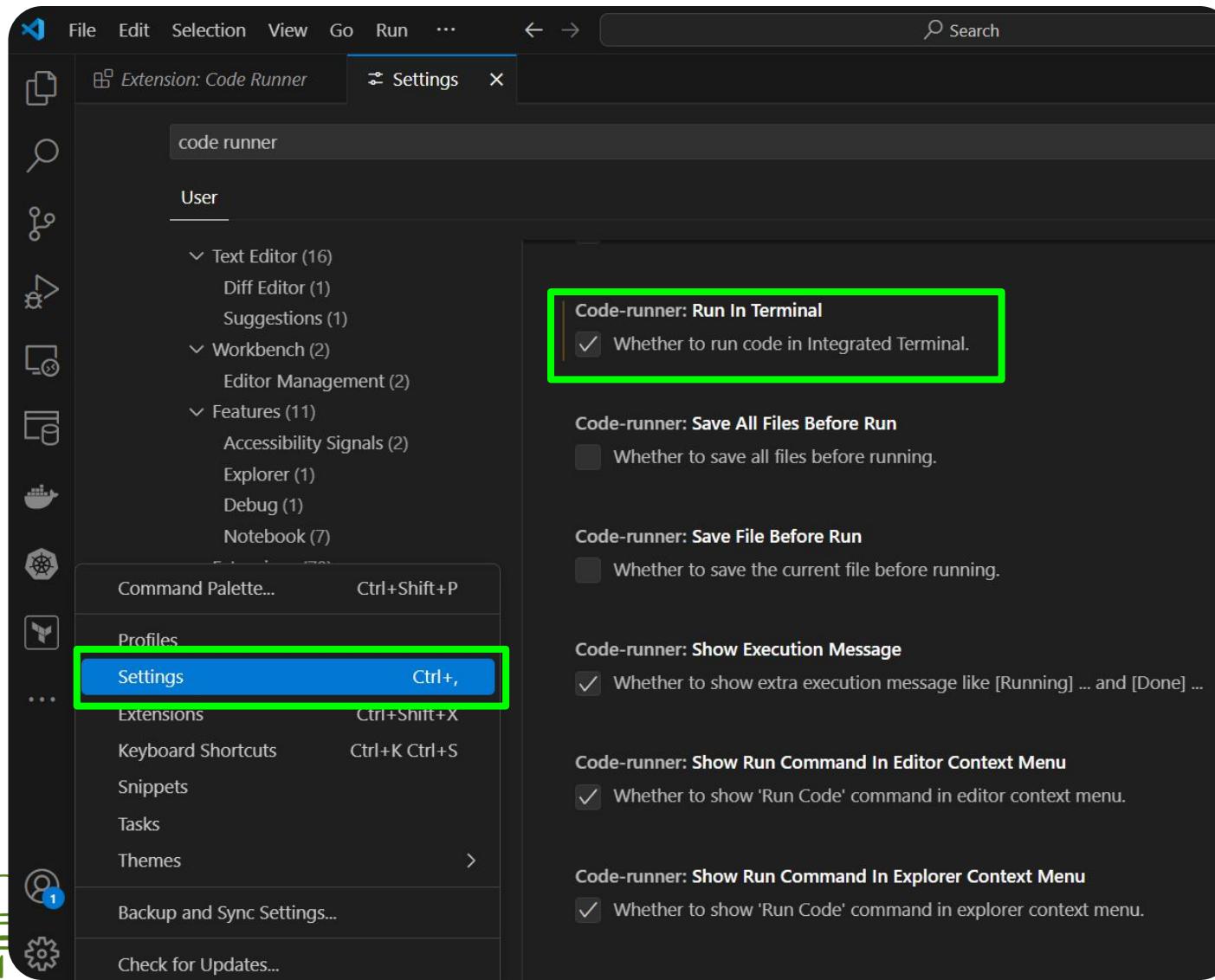
AMBIENTE DE DESENVOLVIMENTO

Visual Studio Code: Plugin Code Runner



AMBIENTE DE DESENVOLVIMENTO

Visual Studio Code: Ativar opção de execução em terminal



AMBIENTE DE DESENVOLVIMENTO

Execução de um código em Kotlin via terminal

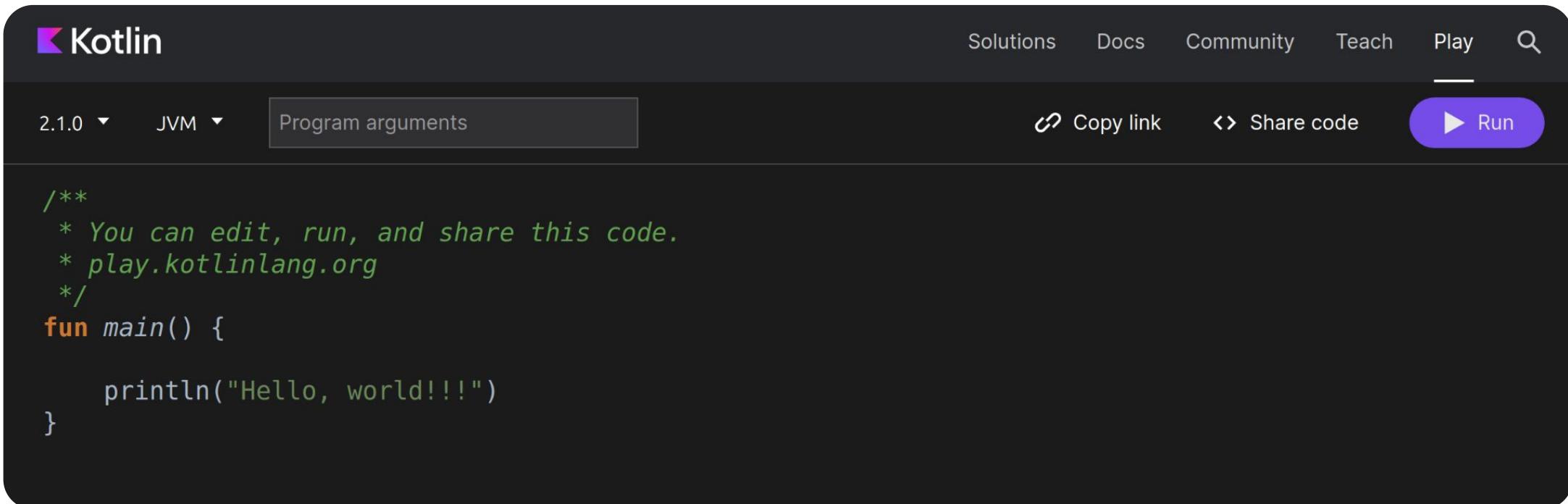


```
1 kotlinc <nome-arquivo>.kt -include-runtime -d <nome-arquivo>.jar  
2 java -jar <nome-arquivo>.jar
```



AMBIENTE DE DESENVOLVIMENTO

Kotlin Playground

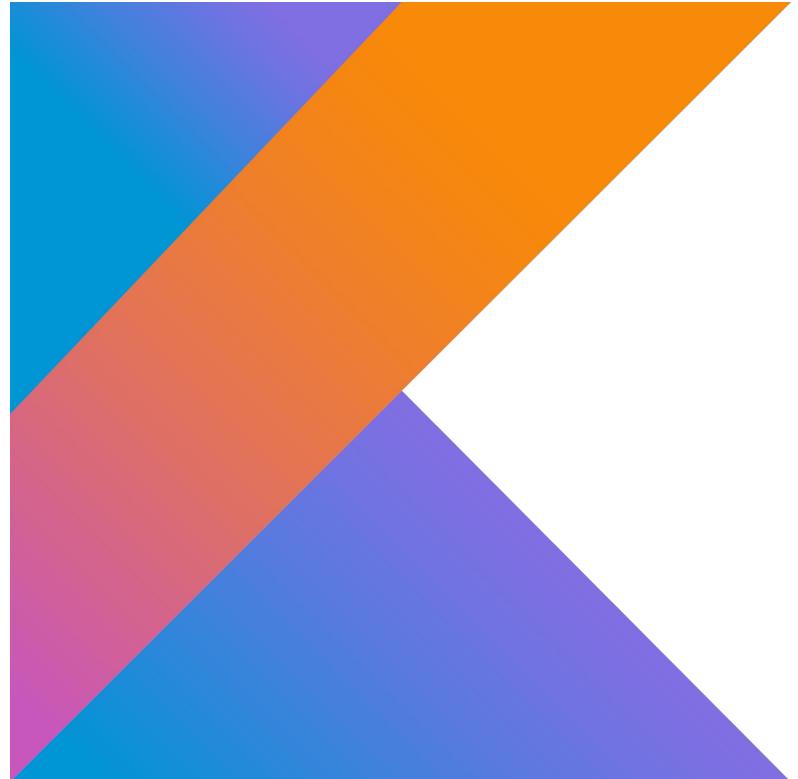


The screenshot shows the Kotlin Playground interface. At the top, there's a navigation bar with the Kotlin logo, version 2.1.0, JVM dropdown, a "Program arguments" input field, and links for Solutions, Docs, Community, Teach, Play, and a search icon. Below the navigation bar is a toolbar with "Copy link", "Share code", and a prominent purple "Run" button. The main area contains the following Kotlin code:

```
/**  
 * You can edit, run, and share this code.  
 * play.kotlinlang.org  
 */  
fun main() {  
    println("Hello, world!!!")  
}
```



Linguagem Kotlin

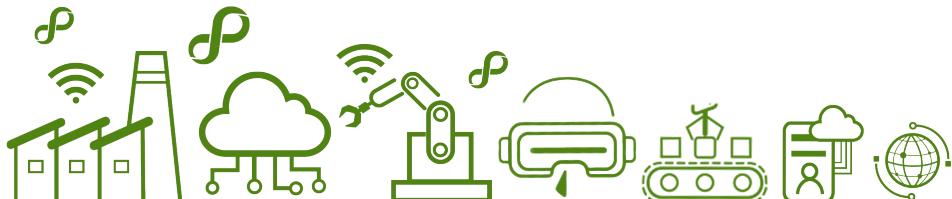


LINGUAGEM KOTLIN



Declaração de Variáveis

```
1 fun main() {  
2  
3     // var: VARIÁVEL MUTÁVEL  
4     // val: VARIÁVEL IMUTÁVEL  
5  
6     var idade: Int = 29  
7     val nome: String = "Will"  
8     var estado: String  
9     var cidade: String? = null  
10    var pais = "Brazil"  
11    var continente = "Teste"  
12  
13 }
```



LINGUAGEM KOTLIN



Declaração de Variáveis

```
1 fun main() {  
2  
3     // var: VARIÁVEL MUTÁVEL  
4     // val: VARIÁVEL IMUTÁVEL  
5  
6     var idade: Int = 29  
7     val nome: String = "Will"  
8     var estado: String  
9     var cidade: String? = null  
10    var pais = "Brazil"  
11    var continente = "Teste"  
12  
13 }
```



Tipos de dados - Numéricos



```
val idade: Int = 25
val ano: Short = 1984
val distancia: Long = 150000L // 'L' indica o tipo Long
val altura: Float = 1.75f // 'L' indica que o tipo Float
val peso: Double = 68.5
```



Tipos de dados - Caracteres



```
val inicial: Char = 'W'
```

```
val nome: String = "Cassian Andor"
```



Tipos de dados - Boolean e Null



```
val estaAtivo: Boolean = true
```

```
var endereco: String? = null
```



Tipos de dados - Any, Unit, Nothing

Any: supertipo de todos os tipos não nulos. “Qualquer coisa”;

Unit: indica que uma função não retorna um valor significativo. Similar ao *void* em Java);

Nothing: indica que uma função não retorna normalmente. Lança exceção.



Tipos de dados - Any, Unit, Nothing

```
fun olaMundo( ): Unit {  
    println("Olá, mundo!")  
}  
  
fun imprimir(qualquerCoisa: Any): Unit {  
    println(qualquerCoisa)  
}  
  
fun lancaExcecao( ): Nothing {  
    throw Exception("Ocorreu um erro!")  
}
```



Tipos de dados - Data Classes

Classe “container” para armazenar dados. Gera métodos como *equals()*, *hashCode()* e *toString()*.



```
data class Pessoa (val nome: String, val idade: Int)  
val will = Pessoa("Will", 30)
```

Operadores

SÍMBOLO	OPERAÇÃO
+	SOMA
-	SUBTRAÇÃO
*	MULTIPLICAÇÃO
/	DIVISÃO
%	RESTO
=	ATRIBUIÇÃO



Operadores

SÍMBOLO	OPERAÇÃO
+ =	ATRIBUIÇÃO DE ADIÇÃO
- =	ATRIBUIÇÃO DE SUBTRAÇÃO
* =	ATRIBUIÇÃO DE MULTIPLICAÇÃO
/ =	ATRIBUIÇÃO DE DIVISÃO
% =	ATRIBUIÇÃO DE RESTO



Operadores

SÍMBOLO	OPERAÇÃO
++	INCREMENTO
--	DECREMENTO



Operadores

SÍMBOLO	OPERAÇÃO
&&	AND (E)
 	OR (OU)
!	NOT(NEGAÇÃO)



Operadores

SÍMBOLO	OPERAÇÃO
<code>==</code>	IGUALDADE
<code>!=</code>	DIFERENTE
<code>></code>	MAIOR
<code>≥</code>	MAIOR OU IGUAL
<code><</code>	MENOR
<code>≤</code>	MENOR OU IGUAL



LINGUAGEM KOTLIN

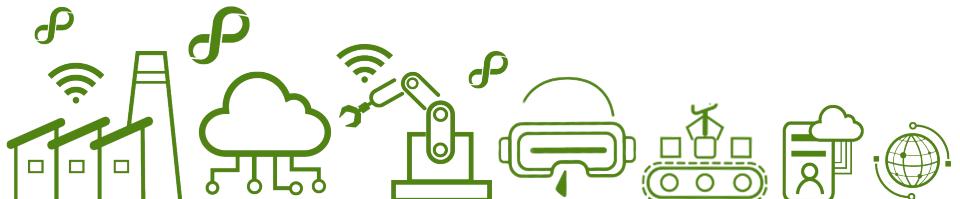
TEMPLATE STRING

```
fun main() {  
    val nome = "Willon"  
    println("Olá, ${nome}!!")  
}
```

Olá, Willon!!

```
fun main() {  
    val nome = "Willon"  
    println("Olá, $nome")  
}
```

Olá, Willon





LINGUAGEM KOTLIN

ENTRADA DE VALOR

main.kt > ...

1

Run | Debug

2 fun main() {

3 val nome = readLine()

4 println(nome)

5 }

PROBLEMS

OUTPUT

DEBUG CONSOLE

TERMINAL

PORTS

```
cd "/home/will/Desktop/kotlin/" && kotlinc main.kt -include-runtime
```

```
will@will-Alienware-m15-R6: /home/will/Desktop/kotlin
```

```
● → cd "/home/will/Desktop/kotlin/" && kotlinc main.kt -include-
```

jar

Willon
Willon



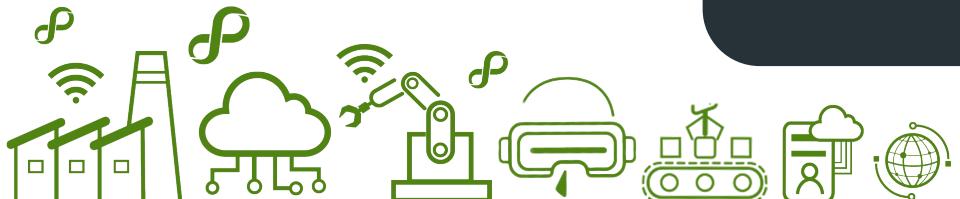
Estruturas de repetição - For

```
for (i in 1..10) {  
    println(i)  
}
```



Estruturas de repetição - While

```
var i = 0
while (i < 10) {
    println(i)
    i++
}
```



Estruturas de repetição - Do While

```
i = 0
do {
    println(i)
    i++
} while (i < 10) // Flag
```



Estruturas de decisão - If/Else



```
idade: Int = 30
```

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

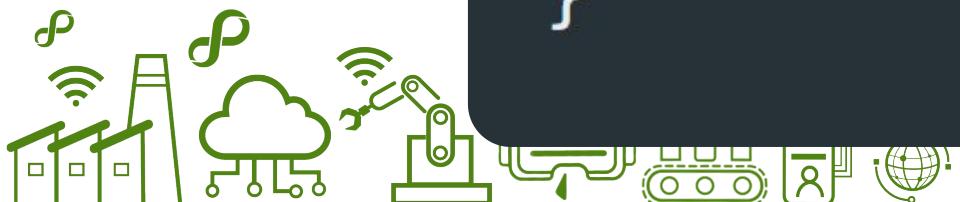


Estruturas de decisão - When



```
val diaSemana = 1

when (diaSemana) {
    1 -> println("É domingo!")
    2 -> println("É Segunda-feira!")
    else -> println("É outro dia!")
}
```



Funções Lambda



```
val saudacao: (String) -> String = { nome -> "Olá, $nome!"}

println(saudacao("Will"))
```



EXERCÍCIOS



do one pushup for every
error in your code

6 months later



"Omg"



Tipos de dados - Estruturados



```
val numeros: Array<Int> = arrayOf(1, 2, 3, 4, 5)
```



LINGUAGEM KOTLIN

Tipos de dados -

Estruturados - arrayOf



```
fun main() {
    val numeros = arrayOf(1, 2, 3, 4, 5)

    for (numero in numeros) {
        println(numero)
    }

    numeros.forEach { numero ->
        println(numero)
    }

    numeros.forEachIndexed { index, numero ->
        println("Índice $index: $numero")
    }

    for (i in numeros.indices) {
        println("Índice $i: ${numeros[i]}")
    }

    for ((index, numero) in numeros.withIndex()) {
        println("Índice $index: $numero")
    }
}
```



Tipos de dados - Estruturados - arrayOf

```
fun main( ) {  
    val numeros = arrayOf(1, 2, 3, 4, 5)  
    val novoArray = numeros + 6  
  
    for (numero in novoArray) {  
        println(numero)  
    }  
}
```



Coleções - listOf



```
1 fun main() {  
2     val numeros = listOf(42, 29, 31, 4, 7, 12) // IMUTÁVEL  
3     val listaVazia = listOf<Int>()  
4  
5     println(numeros.size) // TAMANHO DA LISTA  
6     println(listaVazia.isEmpty()) // LISTA VAZIA?  
7     println(numeros.isNotEmpty()) // LISTA NÃO VAZIA?  
8  
9     println(numeros[0]) // ACESSO POR ÍNDICE  
10    println(numeros[1]) // ACESSO POR ÍNDICE  
11  
12    println(numeros.first()) // PRIMEIRO ELEMENTO  
13    println(numeros.last()) // ÚLTIMO ELEMENTO  
14  
15    println(numeros.getOrNull(4)) // ELEMENTO ENCONTRADO OU NULL  
16    println(numeros.getOrNull(10)) // ELEMENTO ENCONTRADO OU NULL  
17 }
```



LINGUAGEM KOTLIN

Coleções - listOf

```
1 fun main() {  
2     val numeros = listOf(42, 29, 31, 4, 7, 12)  
3  
4     val pares = numeros.filter { it % 2 == 0 } // FILTRAGEM  
5     println(pares)  
6  
7     val dobrados = numeros.map { it * 2 } // MAPEAMENTO  
8     println(dobrados)  
9  
10    val maiorQueTres = numeros.find { it > 3 } // PRIMEIRO ELEMENTO DA CONDIÇÃO  
11    println(maiorQueTres)  
12  
13    val ordenadosAsc = numeros.sorted() // ORDENAÇÃO (ASC)  
14    println(ordenadosAsc)  
15  
16    val ordenadosDesc = numeros.sortedDescending() // ORDENAÇÃO (DESC)  
17    println(ordenadosDesc)  
18  
19    println(3 in numeros) // EXISTÊNCIA DE VALOR NA LISTA  
20    println(numeros.contains(6)) // EXISTÊNCIA DE VALOR NA LISTA  
21  
22    numeros.forEach { it → println(it)} // FOR EACH PARA INTERAÇÃO COM ÍNDICE|  
23  
24    for (index in 0..numeros.size - 1) { // FOR PARA INTERAÇÃO COM ÍNDICE  
25        println(numeros[index])  
26    }  
27  
28    for (element in numeros) { // FOR PARA INTERAÇÃO COM O ELEMENTO  
29        println(element)  
30    }  
31 }
```





LINGUAGEM KOTLIN

Coleções

mutableListOf

```
1 fun main() {  
2     val numeros = mutableListOf(42, 29, 31, 4, 7, 12) // MUTÁVEL  
3     val listaVazia = mutableListOf<Int>()  
4  
5     println(numeros.size) // TAMANHO DA LISTA  
6     println(listaVazia.isEmpty()) // LISTA VAZIA?  
7     println(numeros.isNotEmpty()) // LISTA NÃO VAZIA?  
8  
9     println(numeros[0]) // ACESSO POR ÍNDICE  
10    println(numeros[1]) // ACESSO POR ÍNDICE  
11  
12    println(numeros.first()) // PRIMEIRO ELEMENTO  
13    println(numeros.last()) // ÚLTIMO ELEMENTO  
14  
15    println(numeros.getOrNull(4)) // ELEMENTO ENCONTRADO OU NULL  
16    println(numeros.getOrNull(10)) // ELEMENTO ENCONTRADO OU NULL  
17  
18 }
```



LINGUAGEM KOTLIN

Coleções

mutableListOf

```
1 fun main() {  
2     val numeros = mutableListOf(42, 29, 31, 4, 7, 12)  
3  
4     val pares = numeros.filter { it % 2 == 0 } // FILTRAGEM  
5     println(pares)  
6  
7     val dobrados = numeros.map { it * 2 } // MAPEAMENTO  
8     println(dobrados)  
9  
10    val maiorQueTres = numeros.find { it > 3 } // PRIMEIRO ELEMENTO DA CONDIÇÃO  
11    println(maiorQueTres)  
12  
13    val ordenadosAsc = numeros.sorted() // ORDENAÇÃO (ASC)  
14    println(ordenadosAsc)  
15  
16    val ordenadosDesc = numeros.sortedDescending() // ORDENAÇÃO (DESC)  
17    println(ordenadosDesc)  
18  
19    println(3 in numeros) // EXISTÊNCIA DE VALOR NA LISTA  
20    println(numeros.contains(6)) // EXISTÊNCIA DE VALOR NA LISTA  
21  
22    numeros.forEach { it -> println(it)} // FOR EACH PARA INTERAÇÃO COM ÍNDICE  
23  
24  
25    for (index in 0..numeros.size - 1) { // FOR PARA INTERAÇÃO COM ÍNDICE  
26        println(numeros[index])  
27    }  
28  
29    for (element in numeros) { // FOR PARA INTERAÇÃO COM O ELEMENTO  
30        println(element)  
31    }  
32 }
```



LINGUAGEM KOTLIN

Coleções

mutableListOf

```
1 fun main() {  
2     val numeros = mutableListOf(42, 29, 31, 4, 7, 12)  
3     val outrosNumeros = mutableListOf(14, 15, 16)  
4     val numerosParaRemover = listOf(29, 31, 12)  
5  
6     numeros.add(99) // ADICIONAR ELEMENTO AO FINAL DA LISTA  
7     println(numeros)  
8  
9     numeros.add(3, 97) // ADICIONAR ELEMENTO EM POSIÇÃO ESPECÍFICA (pos, elem)  
10    println(numeros)  
11  
12    numeros.addAll(outrosNumeros) // ADICIONAR ELEMENTOS DE OUTRA LISTA  
13    println(numeros)  
14  
15    val valor1 = numeros.remove(10) // REMOÇÃO DA PRIMEIRA OCORRÊNCIA DO ELEMENTO  
16    val valor2 = numeros.remove(4) // REMOÇÃO DA PRIMEIRA OCORRÊNCIA DO ELEMENTO  
17    println(numeros)  
18    println(valor1)  
19    println(valor2)  
20  
21    numeros.removeAt(0) // REMOÇÃO DE UMA POSIÇÃO ESPECÍFICA  
22    println(numeros)  
23  
24    numeros.removeAll(numerosParaRemover) // REMOÇÃO DE ELEMENTOS DA COLEÇÃO FORNECIDA  
25    println(numeros)  
26  
27    numeros.clear() // REMOVER TODOS OS ELEMENTOS  
28    println(numeros)  
29 }  
30 }
```



Coleções - mutableListOf



```
1 fun main() {  
2  
3     val numeros = mutableListOf(42, 29, 31, 4, 7, 12)  
4     println(numeros)  
5  
6     numeros[0] = 2 // ALTERAÇÃO DE ELEMENTO  
7     println(numeros)  
8  
9     numeros.set(1, 200) // ALTERAÇÃO DE ELEMENTO EM POSIÇÃO ESPECÍFICA  
10    println(numeros)  
11  
12    println(numeros.indexOf(300)) // RETORNA POSIÇÃO DO ELEMENTO FORNECIDO  
13  
14 }
```



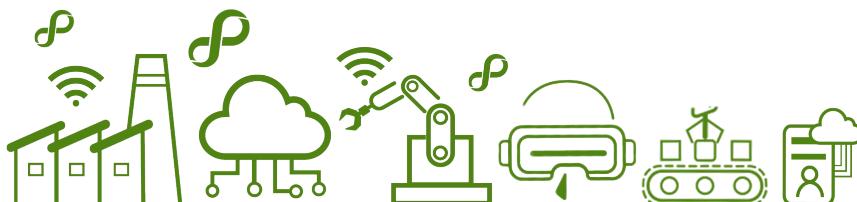
LINGUAGEM KOTLIN

Coleções -

mapOf

mutableMapOf

```
1 fun main() {  
2  
3     // IMUTÁVEL  
4     val mapaImutavel = mapOf("x" to 10, "y" to 20, "z" to 30)  
5     println(mapaImutavel)  
6  
7     // MUTÁVEL  
8     val mapaMutavel = mutableMapOf("nome" to "Will", "idade" to 29, "linguagem" to "Python")  
9     println(mapaMutavel)  
10  
11    val mapaVazio = mutableMapOf<String, Int>()  
12    println(mapaVazio)  
13  
14    // ADICIONAR CHAVE E VALOR  
15    mapaMutavel["cidade"] = "Manaus"  
16    println(mapaMutavel)  
17  
18    // ADICIONAR VÁRIAS CHAVES E VALORES  
19    mapaMutavel.putAll(mapOf("sobrenome" to "Ferreira", "casa" to 461))  
20    println(mapaMutavel)  
21  
22    mapaMutavel.remove("casa") // REMOVER CHAVE  
23    println(mapaMutavel)  
24  
25    mapaMutavel.put("casa", 99) // ADICIONAR CHAVE E VALOR  
26    mapaMutavel.put("sobrenome", "da Silva") // ATUALIZAR CHAVE E VALOR  
27    println(mapaMutavel)  
28  
29 }  
30 }
```



Coleções -

mapOf

mutableMapOf

```
1 fun main() {  
2  
3     // MUTÁVEL  
4     val mapaMutavel = mutableMapOf("nome" to "Will", "idade" to 29, "linguagem" to "Python")  
5     println(mapaMutavel)  
6  
7     println(mapaMutavel.getOrElse("casa", 42)) // ACESSAR CHAVE COM VALOR PADRÃO, CASO NÃO EXISTA  
8     println(mapaMutavel.getOrElse("nome", "LILLY")) // ACESSAR CHAVE COM VALOR PADRÃO, CASO NÃO EXISTA  
9  
10    println(mapaMutavel.size) // TAMANHO  
11    println(mapaMutavel.isEmpty()) // ESTÁ VAZIO?  
12  
13    println(mapaMutavel.containsKey("x")) // EXISTÊNCIA DE CHAVE  
14    println(mapaMutavel.containsValue(20)) // EXISTÊNCIA DE VALOR  
15  
16    for ((chave, valor) in mapaMutavel) { // ITERAÇÃO COM CHAVE E VALOR  
17        println("$chave → $valor")  
18    }  
19  
20    for (chave in mapaMutavel.keys) { // ITERAÇÃO APENAS COM CHAVE  
21        println(chave)  
22    }  
23  
24    for (valor in mapaMutavel.values) { // ITERAÇÃO APENAS COM VALOR  
25        println(valor)  
26    }  
27  
28    // MAPEAMENTO  
29    val mapaImutavel = mapOf("x" to 10, "y" to 20, "z" to 30)  
30    val mapaDobrado = mapaImutavel.mapValues { it.value * 2 }  
31    println(mapaDobrado)  
32  
33 }  
34 }
```



Enums



```
enum class DiaSemana { SEGUNDA, TERÇA, QUARTA, QUINTA, SEXTA, SÁBADO, DOMINGO }

enum class Cor(val rgb: String) {
    VERMELHO("#FF0000"),
    VERDE("#00FF00"),
    AZUL("#0000FF")
}

fun main() {
    val hoje = DiaSemana.SEXTA
    println(hoje)

    val cor = Cor.VERMELHO
    println("Cor: ${cor.name}, Código RGB: ${cor.rgb}")
}
```

EXERCÍCIOS



do one pushup for every
error in your code

6 months later



"Omg"



DESAFIOS I



do one pushup for every
error in your code

6 months later



"Omg"



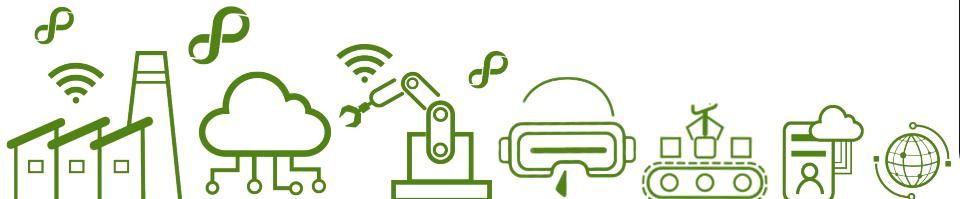


Orientação a objeto -

Classes e Objetos

```
class Carro {  
    var marca: String? = null  
    var modelo: String? = null  
    var ano: Int? = null  
}
```

```
fun main() {  
    val meuCarro = Carro()  
  
    meuCarro.ano = 2025  
    println(meuCarro)  
    println(meuCarro.ano)  
    println(meuCarro.marca)  
}
```



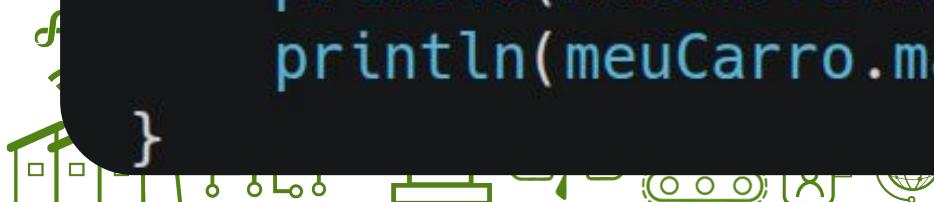
Orientação a objeto - Classes e Objetos



```
// Construtor primário
class Carro(var marca: String, var modelo: String, var ano: Int)

fun main() {
    val meuCarro = Carro("Ford", "Mustang", 2025)

    println(meuCarro.ano)
    println(meuCarro.modelo)
    println(meuCarro.marca)
}
```



Orientação a objeto - Métodos

```
1 class Carro (var marca: String, var modelo: String, var ano: Int) {  
2  
3     fun dirigir() {  
4         println("Wrooom!")  
5     }  
6  
7     fun velocidadeMax(velocMax: Int) {  
8         println("Velocidade máxima é: " + velocMax + "!")  
9     }  
10  
11 }  
12  
13 fun main() {  
14     val carro = Carro("Ford", "Mustang", 1969)|  
15     carro.dirigir()  
16     carro.velocidadeMax(300)  
17 }
```



Orientação a objeto - Objetos de instância única



```
object Logger {  
    fun log(msg: String) = println("LOG: $msg")  
}  
  
fun main() {  
    Logger.log("Sistema iniciado")  
}
```



Orientação a objeto - Modificadores de acesso

Modificador	Dentro da Classe	Dentro do Mesmo Arquivo	Dentro do Mesmo Módulo	Fora do Módulo
private	Sim	Não	Não	Não
protected	Sim	Não	Não	Não
internal	Sim	Sim	Sim	Não
public	Sim	Sim	Sim	Sim



Orientação a objeto - Modificadores de acesso



```
class ContaBancaria(public var nomeCliente: String, private var saldo: Double) {  
  
    fun depositar(valor: Double) { saldo += valor }  
  
    fun obterSaldo() = saldo  
  
    fun obterSaldo2(): Double {  
        return this.saldo  
    }  
  
    fun obterSaldo3(): Double {  
        return saldo  
    }  
}
```



Orientação a objeto - Modificadores de acesso

```
fun main() {  
    val conta = ContaBancaria("Will", 100.0)  
    conta.depositar(50.0)  
    println(conta.obterSaldo())  
    println(conta.obterSaldo2())  
    println(conta.obterSaldo3())  
    println(conta.saldo)  
}
```



Orientação a objeto - Interfaces



```
interface Trabalhavel {  
    fun trabalhar()  
}  
  
class Desenvolvedor : Trabalhavel {  
    override fun trabalhar() = println("Escrevendo código...")  
}  
  
fun main() {  
    val dev = Desenvolvedor()  
    dev.trabalhar()  
}
```



Orientação a objeto -

Herança

```
1 open class ClassePai {  
2     val valorPai = 5  
3  
4     fun funcaoPai() {  
5         println("Função pai!")  
6     }  
7 }  
8  
9 class ClasseFilha: ClassePai() {  
10    fun funcaoFilha() {  
11        println("Função filha!")  
12    }  
13 }  
14  
15 fun main() {  
16     val objeto = ClasseFilha()  
17     objeto.funcaoPai()  
18     objeto.funcaoFilha()  
19 }  
20 }
```



Orientação a objeto - Herança

```
open class Animal(val nome: String) {  
    open fun som() = "Faz um som"  
    final fun greeting() = "Olá!" // Padrão  
}  
  
class Cachorro(nome: String) : Animal(nome) {  
    override fun som() = "Latido"  
  
    // Não é possível sobrescrever método final  
    override fun greeting() = "Oi!"  
}  
  
fun main() {  
    val dog = Cachorro("Rex")  
    println(dog.som())  
    println(dog.greeting())  
}
```





LINGUAGEM KOTLIN

Orientação a objeto -

Sealed classes

“Container” de
subclasses

```
sealed class Resultado {  
    class Sucesso(val dados: String) : Resultado()  
    class Erro(val mensagem: String) : Resultado()  
}  
  
class Warning(val mensagem: String): Resultado()  
  
fun processarResultado(resultado: Resultado) {  
    when (resultado) {  
        is Resultado.Sucesso -> println("Sucesso: ${resultado.dados}")  
        is Resultado.Erro -> println("Erro: ${resultado.mensagem}")  
    }  
}  
  
fun main() {  
    val sucesso: Resultado = Resultado.Sucesso("Operação concluída!")  
    val erro: Resultado = Resultado.Erro("Falha na conexão.")  
    val warning: Warning = Warning("Conexão lenta. Atenção.")  
  
    processarResultado(sucesso)  
    processarResultado(erro)  
}
```



LINGUAGEM KOTLIN



Orientação a objeto -

Sealed classes

“Container” de
subclasses

```
sealed class Resultado {  
    class Sucesso(val dados: String) : Resultado()  
    class Erro(val mensagem: String) : Resultado()  
    class Warning(val mensagem: String): Resultado()  
}  
  
fun processarResultado(resultado: Resultado) {  
    when (resultado) {  
        is Resultado.Sucesso -> println("Sucesso: ${resultado.dados}")  
        is Resultado.Erro -> println("Erro: ${resultado.mensagem}")  
        is Resultado.Warning -> println("Atenção: ${resultado.mensagem}")  
    }  
}  
  
fun main() {  
    val sucesso: Resultado = Resultado.Sucesso("Operação concluída!")  
    val erro: Resultado = Resultado.Erro("Falha na conexão.")  
    val warning: Resultado = Resultado.Warning("Conexão lenta.")  
  
    processarResultado(sucesso)  
    processarResultado(erro)  
    processarResultado(warning)  
}
```



EXERCÍCIOS 2



do one pushup for every
error in your code

6 months later



"Omg"



DESAFIOS II



do one pushup for every
error in your code

6 months later



"Omg"



Obrigado!

