



Desenvolvimento  
Mobile 1  
Aula 02

Prof. Me Daniel Vieira

**SENAI**

# Agenda

- 1- Framework
- 2- A Linguagem Dart
- 3 - Tipos de variáveis
- 4 - Exemplo de código com a linguagem Dart
- 5- Criando projeto no VS CODE

# Framework

- Framework é um conjunto estruturado de ferramentas, bibliotecas, padrões e componentes que oferece uma base para o desenvolvimento de aplicações. Ele serve como um "esqueleto" ou infraestrutura que os desenvolvedores podem usar e estender para construir software de forma mais rápida e consistente.

Web



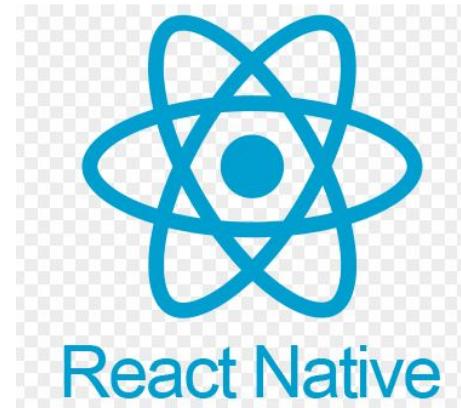
# Framework

- Framework é um conjunto estruturado de ferramentas, bibliotecas, padrões e componentes que oferece uma base para o desenvolvimento de aplicações. Ele serve como um "esqueleto" ou infraestrutura que os desenvolvedores podem usar e estender para construir software de forma mais rápida e consistente.

Mobile



Flutter



Swift

# Diferença entre biblioteca e framework

- **Biblioteca:** Conjunto de funções que você chama conforme a necessidade, controlando o fluxo do programa.  
Exemplo: NumPy em Python.
- **Framework:** Define o fluxo do programa, controlando a execução de certas partes e chamando o código do desenvolvedor.  
Exemplo: Django, que já oferece estrutura para o desenvolvimento web.

# Vantagens e desvantagens do uso de framework

## Vantagens:

- **Produtividade:** Acelera o desenvolvimento ao reutilizar soluções prontas.
- **Organização:** Segue padrões pré definidos que facilitam a manutenção do código.
- **Comunidade:** Frameworks populares contam com grande suporte e documentação.

## Desvantagens:

- **Curva de aprendizado:** Pode ser difícil aprender frameworks mais complexos.
- **Rigidez:** Impõe certas regras, o que pode limitar a flexibilidade do desenvolvedor em casos específicos.

# Vantagens e desvantagens do uso de framework

Um framework é, portanto, uma ferramenta essencial para o desenvolvimento moderno, facilitando o trabalho do desenvolvedor ao mesmo tempo que garante eficiência e consistência.

# Linguagem Dart

- Dart é uma linguagem de programação apresentada pelo Google em 2011 com o objetivo de substituir o JavaScript como principal linguagem utilizada nos navegadores.
- Sintaxe C-Like.
- Paradigma de orientação a objetos.
- Todos os objetos herdam da classe object
- Fortemente tipada, mas não é necessário colocar um tipo, pois o Dart consegue inferir os tipos.
- No Java, C# utilizamos a palavra reservada private, enquanto no Dart, basta colocar um underline(\_) no início do nome de um atributo, método ou classe para torná-lo privado.
- Dart pode ser compilada em ahead-of time(AOT) e just-in-time(JIT)
- Compilação em tempo real e retorno em tempo real da alteração (Hot reload).

# Tipos de Dados

## 1. Primitivos (básicos)

Estes são os tipos de dados fundamentais disponíveis na maioria das linguagens:

- **Numéricos:**

**Inteiros (int):** Números inteiros positivos ou negativos sem ponto decimal.

Exemplo: -5, 0, 42.

**Ponto flutuante (float ou double):** Números com casas decimais.

Exemplo: 3.14, -0.001.

**Decimal (ou precisão alta):** Números com alta precisão para cálculos financeiros.

Exemplo: 1.0001.

- **Caractere (char):**

Representa um único caractere Unicode.

Exemplo: 'a', '1', '?'.

- **String:**

Conjunto de caracteres formando textos.

# Tipos de Dados

Exemplo: "Olá, mundo!".

- **Booleano (bool):**

Representa valores lógicos: True ou False.

Exemplo: True.

## 2. Estruturados

São usados para agrupar vários valores:

- **Listas (arrays ou vetores):**

Coleções ordenadas de elementos, geralmente do mesmo tipo.

Exemplo: [1, 2, 3] ou ["a", "b", "c"].

- **Tuplas:**

Coleções ordenadas e imutáveis de elementos.

- Exemplo: (1, "a", True).

# Tipos de Dados

- **Conjuntos (sets):**

Coleções não ordenadas de elementos únicos.

Exemplo: {1, 2, 3}.

- **Dicionários (ou mapas):**

Coleções de pares chave-valor.

Exemplo: {"nome": "Daniel", "idade": 30}.

# Tipos de Dados

## 4. Abstratos

Definidos pelo programador e baseados em tipos primitivos ou estruturados:

- **Classes:** Criadas com a programação orientada a objetos.
- **Interfaces:** Definem o comportamento esperado de uma classe.
- **Enums:** Conjuntos fixos de constantes.

Exemplo: enum Dia { Segunda, Terça, Quarta }.

# Tipos de Dados

## 5. Específicos de cada linguagem

Algumas linguagens possuem tipos de dados específicos:

- **Python:**

dict, list, tuple, set.

- **JavaScript:**

undefined, null, symbol.

- **SQL:**

VARCHAR, INTEGER, DATE.

## Conversões entre tipos (casting)

- **Implícito:** A linguagem converte automaticamente (exemplo: int para float).
- **Explícito:** O programador força a conversão (exemplo: float(5) em Python).

## Boas práticas

- Escolha tipos que otimizem o uso de memória e desempenho.
- Utilize nomes descritivos para variáveis de tipos complexos.
- Certifique-se de que os tipos sejam compatíveis ao realizar operações.

# Tipos de variáveis

Variável - Área de memória associada a um nome que pode armazenar valores de um determinado tipo.

- var nome = "Daniel Vieira";
- String email = "[danielvieira2006@gmail.com](mailto:danielvieira2006@gmail.com)";
- int numero = 50;
- double preco = 19.99;
- bool acesso = true; // false;

Declaração

```
var nome(mutável)
const pi  (imutável)
```

Memória do computador	
X	
nome	Daniel Vieira
pi	3,14

# Tipos de variáveis Dart

```
import "dart:io";
void main()
{
    // Variável que armazena números inteiros
    int idade = 28;
    print("Idade: $idade");
    // Variável que armazena números decimais
    double raio = 10.25;
    print("Raio: $raio");
    // Variável que armazena caracteres e textos
    String nome = "Daniel";
    print("Olá $nome, seja bem vindo");
    //Variavel que armazena verdadeiro ou false
    bool ligado = true;
```

# Tipos de variáveis Dart

```
// Variável que guarda uma lista genérica  
List numerosgenericos = [10,"Daniel", true,20];  
print(numerosgenericos);
```

```
// Variável que guarda uma lista de numeros inteiros  
List<int> numerosinteiros = [10,20,30,40];  
print(numerosinteiros);
```

```
//Variável que guarda um dicionário com chave e valor em formato texto  
Map<String, String> nome_sobrenome = {"Daniel": "Vieira", "Senai": "Roberto Mange"};
```

```
// Variável sem tipo pré definido, seu tipo é igual ao tipo do primeiro valor que recebe  
var sobrenome = "Vieira";  
print(sobrenome);
```

# Exemplo código Dart

Importa a biblioteca dart:io que permite que o usuário digite valores utilizando o teclado	import "dart:io";
Função principal do código	void main()
Variável que armazena números inteiros	{ int idade = 28;
Exibe no terminal o valor da variável ligado	print("Idade: \$idade");
Variável que armazena números decimais	double raio = 10.25;
Exibe no terminal o valor da variável raio	print("Raio: \$raio");
Variável que armazena caracteres e textos	String nome = "Daniel";
Exibe no terminal o valor da variável nome	print("Olá \$nome, seja bem vindo");

## Exemplo código Dart

# Operadores de comparação

Operadores de comparação	Função	Exemplo
<code>==</code>	Comparação entre dois valores	<code>print(5 == 5); // true</code> <code>print(5 == 3); // false</code>
<code>!=</code>	Diferença entre dois valores	<code>print(5 != 3); // true</code> <code>print(5 != 5); // false</code>
<code>&gt;</code>	Maior que Verifica se o valor à esquerda é maior que o valor à direita.	<code>print(10 &gt; 5); // true</code> <code>print(5 &gt; 10); // false</code>
<code>&lt;</code>	Menor que Verifica se o valor à esquerda é menor que o valor à direita.	<code>print(5 &lt; 10); // true</code> <code>print(10 &lt; 5); // false</code>

# Operadores lógicos

Operadores lógicos	Função	Exemplo
&& (E)	Retorna true se ambas as condições forem verdadeiras.	<code>print(5 == 5); // true print(5 == 3); // false</code>
(OU)	Retorna true se pelo menos uma das condições for verdadeira.	<code>var notaProva = 4; var notaTrabalho = 8; print(notaProva &gt;= 6    notaTrabalho &gt;= 6); // true, pois a segunda condição é verdadeira</code>
! (NÃO/negação)	Inverte o valor lógico.	<code>bool aprovado = true; print(!aprovado); // false, pois o operador NOT inverte o valor</code>

# Exemplo de código solicitando dados ao usuário

Importa a biblioteca dart:io que permite que o usuário digite valores utilizando o teclado	import 'dart:io';
Função principal do código	void main() {
Exibe mensagem no terminal solicitando que o usuário digite seu nome	print("Digite seu nome:");
A variável nome armazena o que o usuário digitar. O comando stdin.readLine Sync() captura o que o usuário digita ! indica que não está vazio e realiza a conversão para String	String nome = stdin.readLineSync()!;

# Exemplo de código solicitando dados ao usuário

Exibe mensagem no terminal solicitando que o usuário digite sua idade.	<pre>print("Digite sua idade:");</pre>
O comando stdin.readLineSync() captura o que o usuário digita ! indica que não está vazio e realiza a conversão para String	<pre>String idadeString = stdin.readLineSync()!;</pre>
int.parse realiza a conversão de um dado string para inteiro.	<pre>int idade = int.parse(idadeString);</pre>
Exibe as informações no terminal "\$nome" o símbolo \$ permite acessar o valor armazenado na variável.	<pre>print("Seu nome é: \$nome"); print("Sua idade é: \$idade"); }</pre>

# Código sendo executado no VS CODE

The screenshot shows the VS Code interface with the following details:

- Code Editor:** Displays a Dart file named `ex2.dart` containing code to read user input for name and age, and print them back.
- Terminal:** Shows the output of running the code. It includes the command `dart ex2.dart`, user input for name and age, and the resulting output showing the name and age printed back.
- Bottom Bar:** Includes tabs for PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL, and PORTS, along with a terminal icon and other UI elements.

```
codigos_dart > ex2.dart > main
1 import 'dart:io';
2
3 Run | Debug
4 void main() {
5     // Solicitando ao usuário que digite seu nome
6     String nome = stdin.readLineSync()!;
7     // Solicitando ao usuário que digite sua idade
8     print("Digite sua idade:");
9     String idadeString = stdin.readLineSync()!;
10    int idade = int.parse(idadeString);
11
12    // Exibindo as informações digitadas pelo usuário
13    print("Seu nome é: $nome");
14    print("Sua idade é: $idade");
15 }
16
```

PROBLEMS 2 OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
engdanielvieira@MacBook-Air-de-Daniel codigos_dart % ls
ex1.dart
engdanielvieira@MacBook-Air-de-Daniel codigos_dart % dart ex1.dart
Idade: 28
Olá: 10.25
Olá Daniel, seja bem vindo
engdanielvieira@MacBook-Air-de-Daniel codigos_dart % dart ex2.dart
Digite seu nome:
Daniel
Digite sua idade:
29
Seu nome é: Daniel
Sua idade é: 29
engdanielvieira@MacBook-Air-de-Daniel codigos_dart % 
```

# Principais comandos da linguagem Dart

Importa a biblioteca dart:io que permite que o usuário digite valores utilizando o teclado	import 'dart:io';
void -> Vazio, não retorna nada; main é a função principal do código, nela todas as inicializações de bibliotecas, variáveis são executadas () → recebe argumentos de fora;	void main() {}
função que imprime informações no console;	print();
Necessário para terminar uma linha de código.	;
O \$ é uma interpolação de string que permite acessar o valor da variável	Para acessar informações de uma variável utilizamos o \$variável Necessário estar entre “ \$variavel”
int.parse realiza a conversão de um dado string para int float.parse realiza a conversão de um dado string para float	int.parse(readLineSync()!); float.parse(readLineSync()!);

# Principais comandos da linguagem Dart

Importa a biblioteca dart:io que permite que o usuário digite valores utilizando o teclado	import 'dart:io';
void -> Vazio, não retorna nada; main é a função principal do código, nela todas as inicializações de bibliotecas, variáveis são executadas () → recebe argumentos de fora;	void main() {}
função que imprime informações no console;	print();
Necessário para terminar uma linha de código.	;
O \$ é uma interpolação de string que permite acessar o valor da variável	Para acessar informações de uma variável utilizamos o \$variável Necessário estar entre “ \$variavel”
int.parse realiza a conversão de um dado string para int float.parse realiza a conversão de um dado string para float	int.parse(readLineSync()!); float.parse(readLineSync()!);

# Aplicativos Mobile - Flutter - Desenvolvimento

The screenshot shows a Flutter development setup. On the left, the code editor displays `main.dart` with the following content:

```
main.dart x
projeto-flutter > projeto3 > lib > main.dart > MyApp > build
1 import 'package:flutter/material.dart';
2
3 void main() {
4   runApp(MyApp());
5 }
6
7 class MyApp extends StatelessWidget {
8   @override
9   Widget build(BuildContext context) {
10     return MaterialApp(
11       title: 'Counter App',
12       theme: ThemeData(
13         appBarTheme: AppBarTheme(
14           backgroundColor: Colors.red,
15         ), // AppBarTheme
16       ), // ThemeData
17       home: CounterScreen(),
18     );
19   }
}
```

On the right, a mobile phone icon represents the running application. The app's title bar says "Primeiro aplicativo em Flutter". The screen shows a counter value of 7. Below the counter are three buttons: "Incrementar" (green), "Reset" (blue), and "Decrementar" (red).

# Aplicativos Mobile - Flutter - Recursos necessários

Git <https://git-scm.com/download/win>

SDK Flutter <https://docs.flutter.dev/get-started/install>

Android Studio SDK + Emulador

[https://developer.android.com/studio?gclid=Cj0KCQjw2eilBhCCARIsAG0Pf8vTiRnMeJg5uKGukaJuvs-Y54bJas-86pWq6tzA8zHcevK57S8Mx0aAI7eEALw\\_wcB&qclsrc=aw.ds](https://developer.android.com/studio?gclid=Cj0KCQjw2eilBhCCARIsAG0Pf8vTiRnMeJg5uKGukaJuvs-Y54bJas-86pWq6tzA8zHcevK57S8Mx0aAI7eEALw_wcB&qclsrc=aw.ds)

VSCode <https://code.visualstudio.com/download>

Extensões Flutter

Dart : <https://dart.dev/get-dart>

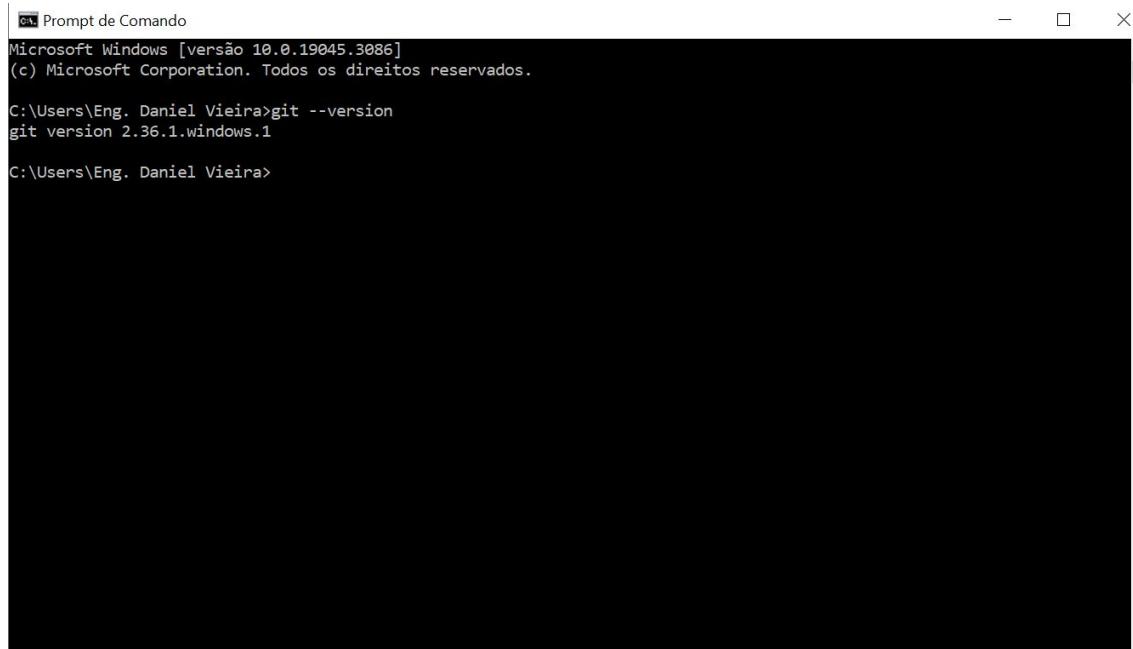
# Git instalação

1º Passo : Verificar se o Git está instalado no computador

Menu Iniciar -> Digite cmd

Vai abrir prompt de comando

Digitar o comando git --version



```
Prompt de Comando
Microsoft Windows [versão 10.0.19045.3086]
(c) Microsoft Corporation. Todos os direitos reservados.

C:\Users\Eng. Daniel Vieira>git --version
git version 2.36.1.windows.1

C:\Users\Eng. Daniel Vieira>
```

# Git instalação

1º Passo : Verificar se o Git está instalado no computador

2º Passo entrar no site Git -> Fazer o download e executar a instalação

The screenshot shows the official Git website ([git-scm.com/](https://git-scm.com/)). The main navigation menu includes 'About', 'Documentation', 'Downloads' (which is highlighted in red), and 'Community'. A sidebar on the left provides information about the 'Pro Git book' by Scott Chacon and Ben Straub, available online for free. The central content area is titled 'Download for Windows' and features a large button with the text 'Click here to download the latest (2.41.0) 64-bit version of Git for Windows.' Below this, there are links for 'Other Git for Windows downloads', including 'Standalone Installer', '32-bit Git for Windows Setup.', '64-bit Git for Windows Setup.', 'Portable ("thumbdrive edition")', '32-bit Git for Windows Portable.', and '64-bit Git for Windows Portable.'. There is also a section for 'Using winget tool' with a command-line example: `winget install --id Git.Git -e --source winget`. At the bottom, it says 'Now What?' followed by the text 'Now that you have downloaded Git, it's time to start using it.'

# Instalação do Flutter

1º Passo : Entrar no site flutter e selecionar de acordo com o sistema operacional: Windows, Linux, Mac

The screenshot shows the Flutter website's 'Install' page. At the top, there is a navigation bar with links for Multi-Platform, Development, Ecosystem, Showcase, Docs, and a search bar. To the right of the search bar is a 'Get started' button. Below the navigation bar, a blue banner says 'Read the announcement!' and has a 'Set up an editor' link with icons for GitHub and VS Code.

The main content area is titled 'Install' and shows the path 'Get started > Install'. It asks the user to 'Select the operating system on which you are installing Flutter:' and provides four options: Windows, macOS, Linux, and ChromeOS, each with its respective logo. A yellow callout box at the bottom left contains the text: 'Important: If you're in China, read [Using Flutter in China](#)'. To the right of this box is another 'Set up an editor' link. On the left side of the page, there is a sidebar with a 'Get started' section containing numbered steps (1. Install, 2. Set up an editor, etc.) and a 'From another platform?' section with a 'Dart language overview' link. Other sections in the sidebar include 'Stay up to date', 'Samples & tutorials', 'User interface', 'Navigation & routing', 'Data & backend', 'Accessibility & localization', 'Platform integration', and 'Packages & plugins'.

# Instalação do Flutter

## 2º Passo : Baixar a versão do Flutter

The screenshot shows the official Flutter website's 'Get started' page for Windows. The left sidebar has sections like 'Get started' (with '1. Install' selected), 'Stay up to date', 'Samples & tutorials', and various developer tools. The main content area discusses minimum requirements (Windows 10, 1.64 GB disk space, Git for Windows 2.x), provides a link to the 'Flutter SDK' (downloadable as 'flutter\_windows\_3.10.6-stable.zip'), and includes a warning about installing to paths with special characters. The right sidebar contains links for 'Contents', 'System requirements', 'Get the Flutter SDK', 'Update your path', 'Run flutter doctor', 'Android setup', 'Install Android Studio', 'Set up your Android device', 'Set up the Android emulator', 'Agree to Android Licenses', 'Windows setup', 'Additional Windows requirements', and 'Next step'.

To install and run Flutter, your development environment must meet these minimum requirements:

- **Operating Systems:** Windows 10 or later (64-bit), x86-64 based.
- **Disk Space:** 1.64 GB (does not include disk space for IDE/tools).
- **Tools:** Flutter depends on these tools being available in your environment.
  - [Windows PowerShell 5.0](#) or newer (this is pre-installed with Windows 10)
  - [Git for Windows 2.x](#), with the [Use Git from the Windows Command Prompt](#) option.

If Git for Windows is already installed, make sure you can run `git` commands from the command prompt or PowerShell.

## Get the Flutter SDK

**Important:** If you're in China, read [Using Flutter in China](#).

1. Download the following installation bundle to get the latest stable release of the Flutter SDK:

[flutter\\_windows\\_3.10.6-stable.zip](#)

For other release channels, and older builds, check out the [SDK archive](#).

2. Extract the zip file and place the contained `flutter` in the desired installation location for the Flutter SDK (for example, `C:\src\flutter`).

**Warning:** Do not install Flutter to a path that contains special characters or spaces.

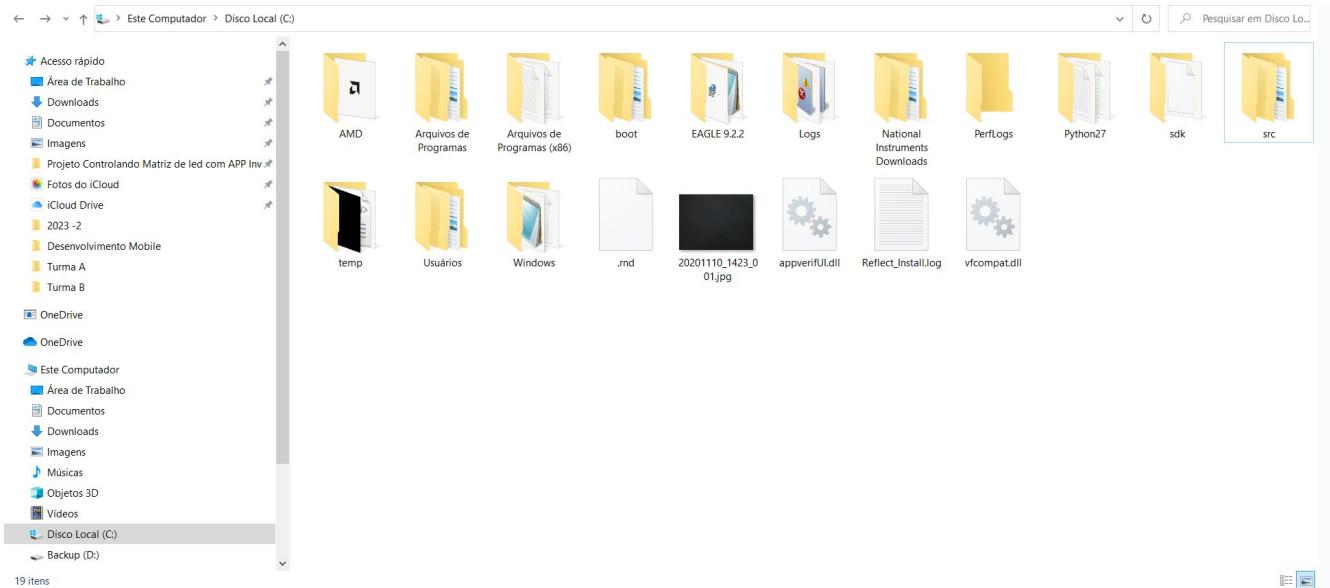
# Instalação do Flutter

3º Passo : Após fazer o download, descompactá-lo.

4º Passo Criar uma pasta no disco C chamada src.

Não colocar a pasta flutter no diretório raiz C, pois pedirá permissão de administrador.

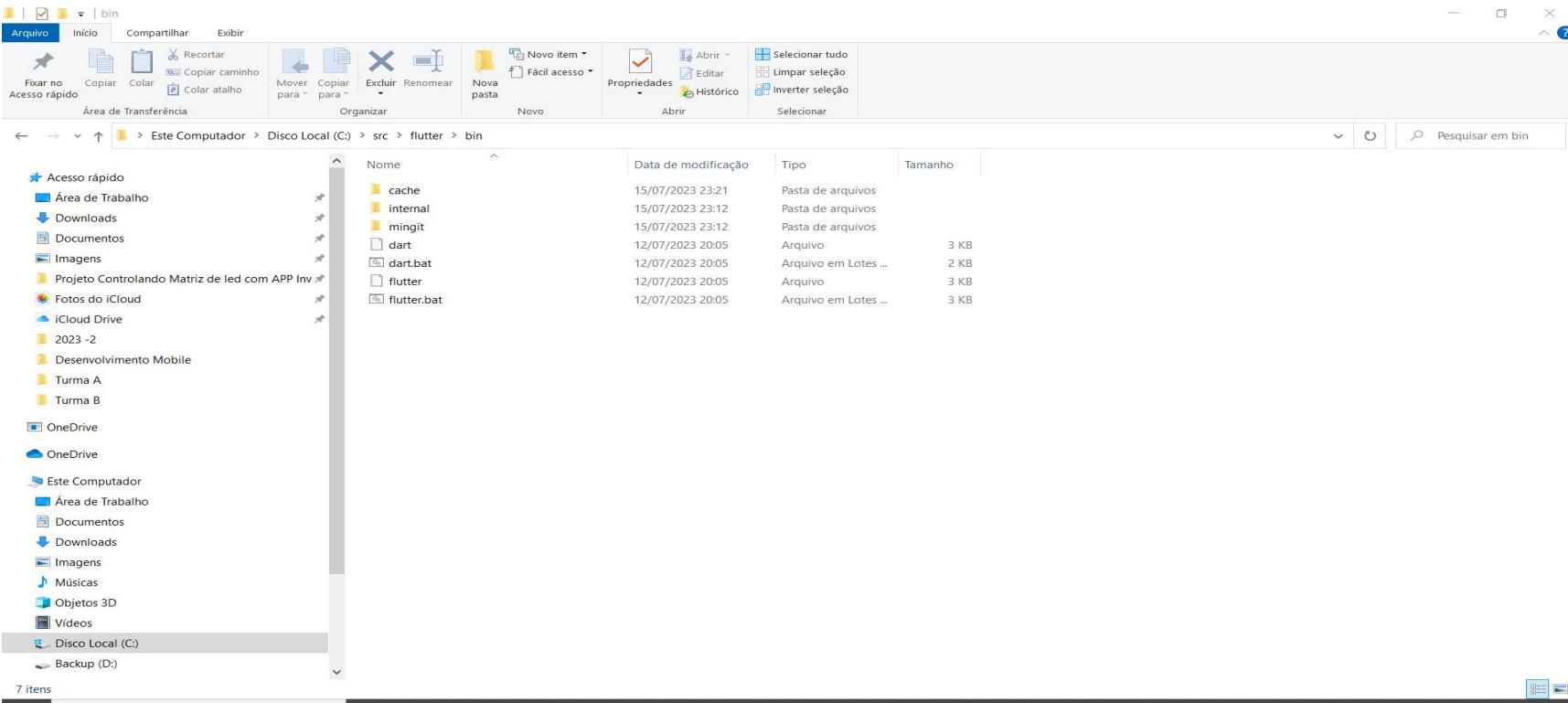
5º Passo: Copiar a pasta flutter para a pasta src



# Instalação do Flutter

6º Passo : Entrar na pasta src ->flutter->bin e copiar esse caminho, pois iremos configurar as variáveis de ambiente

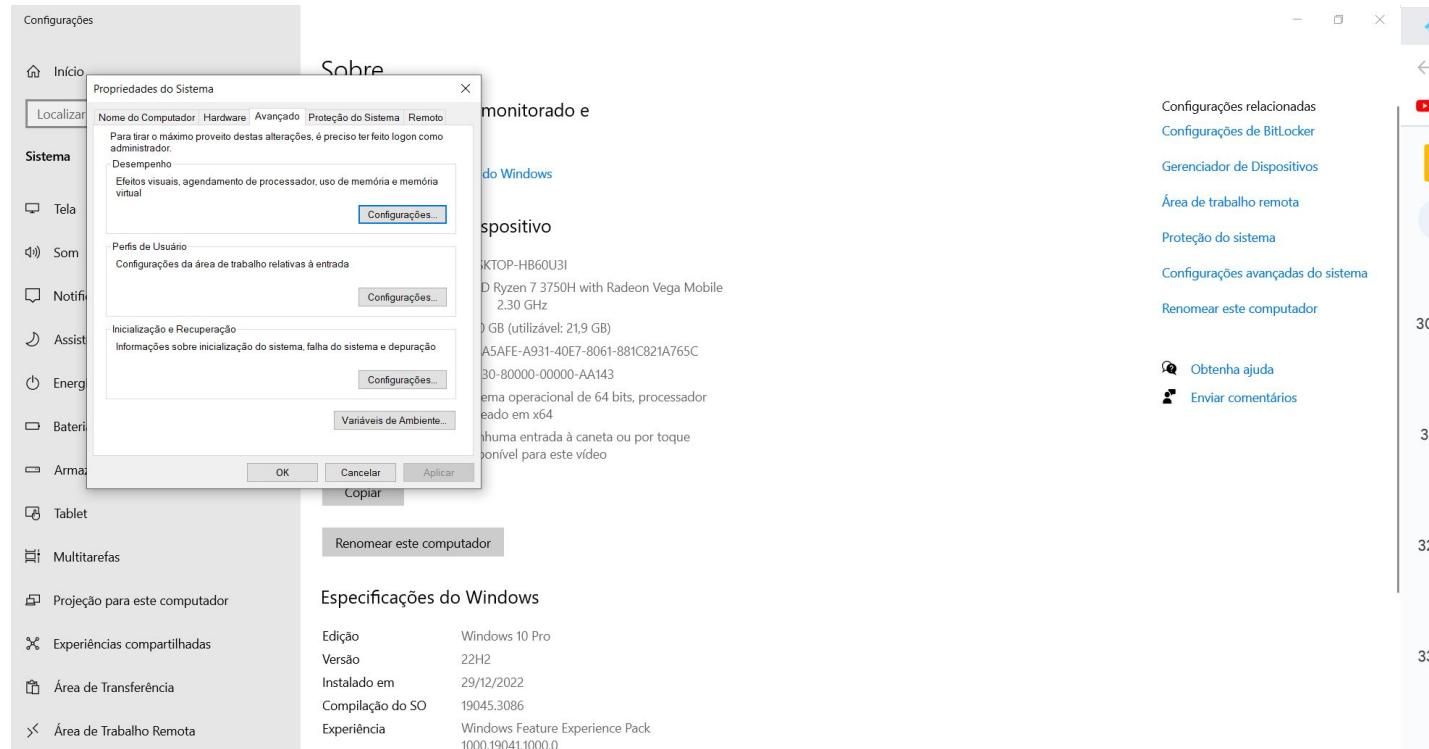
C:\src\flutter\bin



# Instalação do Flutter

## 7º Passo : Configurar variáveis de ambiente

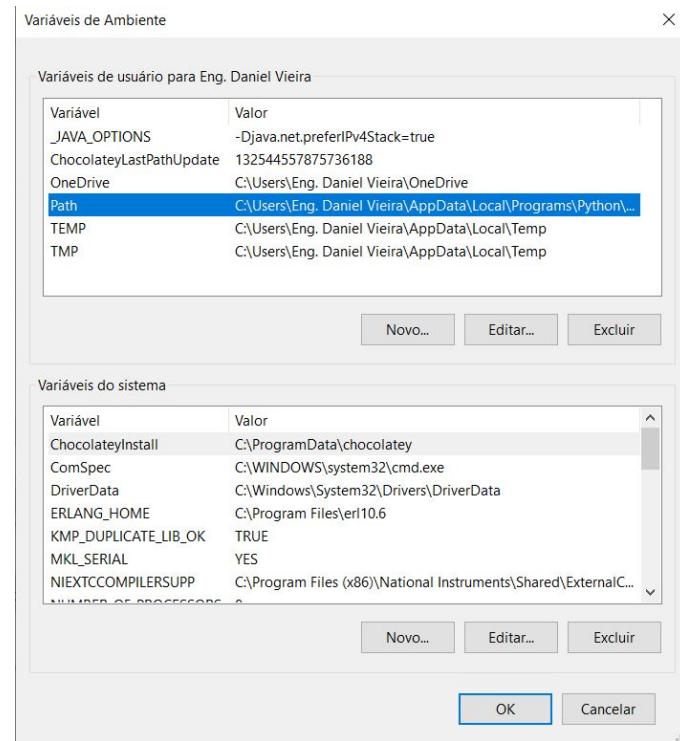
Meu Computador -> Propriedades -> Configurações avançadas do sistema



# Instalação do Flutter

## 8º Passo : Configurar variáveis de ambiente

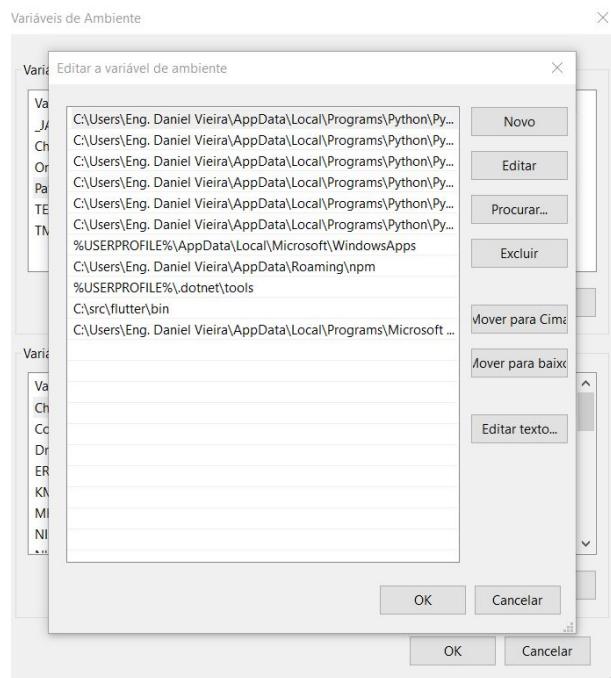
Clicar em variáveis de ambiente -> Procurar pelo campo Path e clicar em editar



# Instalação do Flutter

## 9º Passo : Configurar variáveis de ambiente

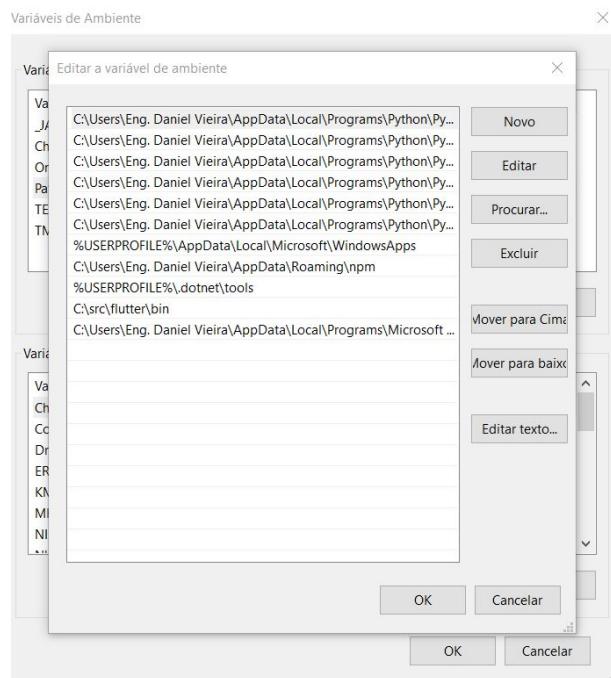
Procurar pelo campo Path e clicar em editar-> Novo e colar o caminho copiado anteriormente  
C:\src\flutter\bin



# Instalação do Flutter

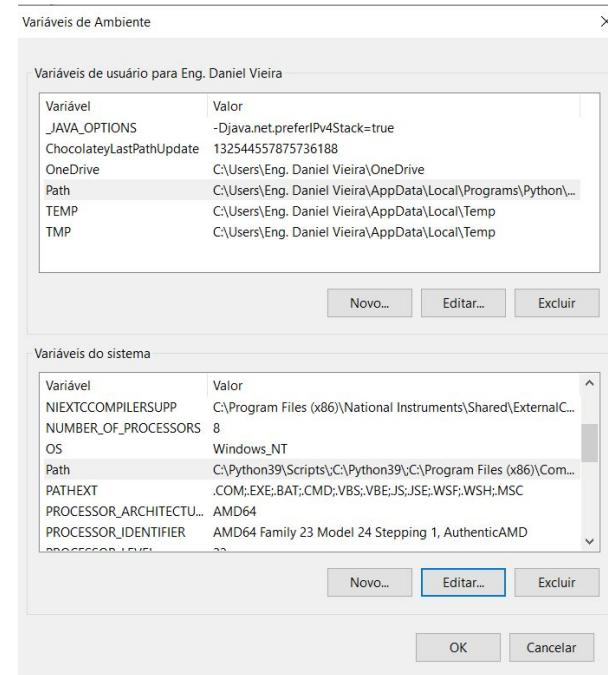
## 9º Passo : Configurar variáveis de ambiente

Procurar pelo campo Path e clicar em editar-> Novo e colar o caminho copiado anteriormente  
C:\src\flutter\bin



# Instalação do Flutter

Aqui na escola temos que colocar como variável do sistema, só do usuário dá conflito.  
Procurar pelo campo Path e clicar em editar-> Novo e colar o caminho copiado anteriormente  
C:\src\flutter\bin - Clicar ok e fecha as janelas

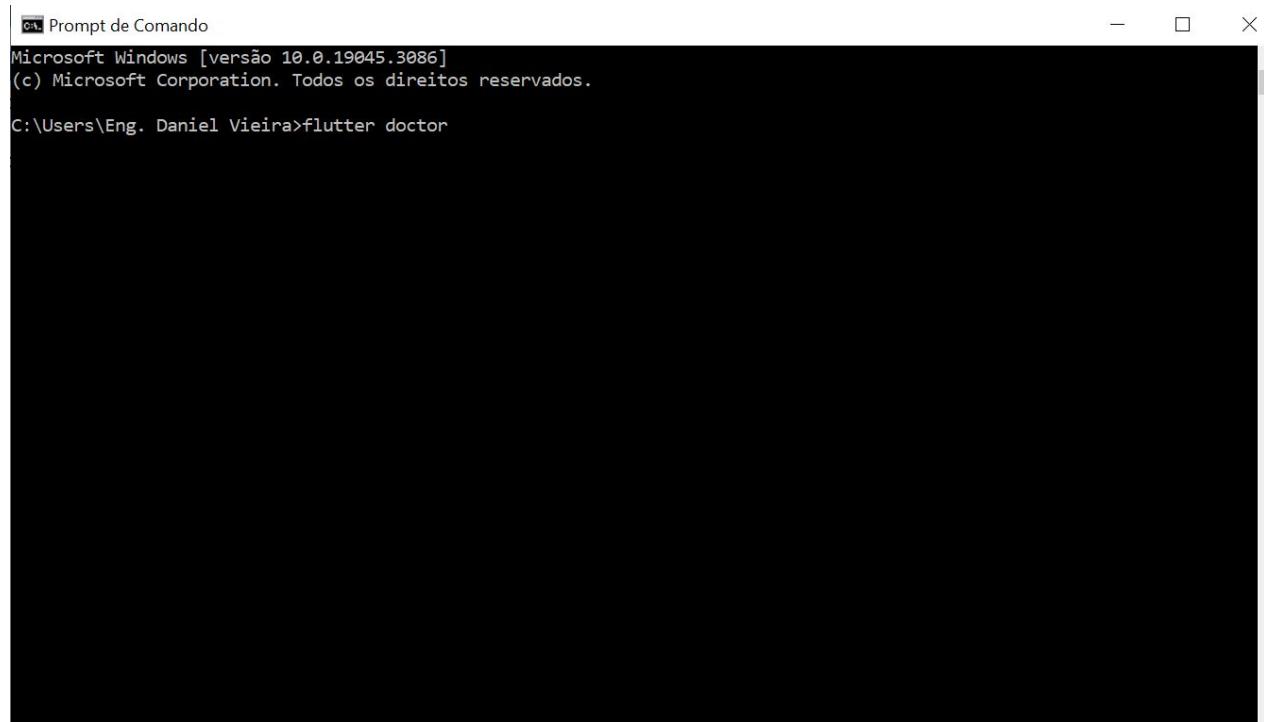


# Instalação do Flutter

10º Passo

Menu iniciar -> cmd

11º Passo digitar flutter doctor para verificar os componentes instalados

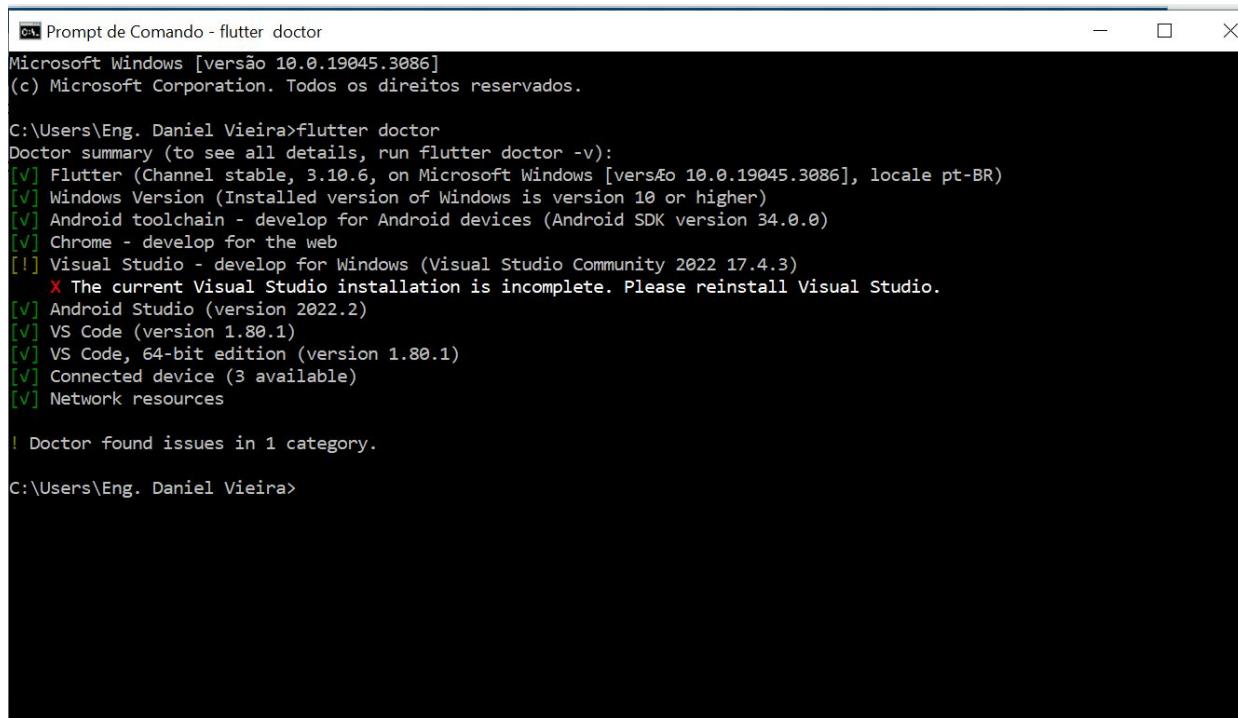


```
Prompt de Comando
Microsoft Windows [versão 10.0.19045.3086]
(c) Microsoft Corporation. Todos os direitos reservados.

C:\Users\Eng. Daniel Vieira>flutter doctor
```

# Instalação do Flutter

12º Se as configurações das variáveis de ambiente foram realizadas com sucesso, aparecerá a tela abaixo



```
Prompt de Comando - flutter doctor
Microsoft Windows [versão 10.0.19045.3086]
(c) Microsoft Corporation. Todos os direitos reservados.

C:\Users\Eng. Daniel Vieira>flutter doctor
Doctor summary (to see all details, run flutter doctor -v):
[✓] Flutter (Channel stable, 3.10.6, on Microsoft Windows [versão 10.0.19045.3086], locale pt-BR)
[✓] Windows Version (Installed version of Windows is version 10 or higher)
[✓] Android toolchain - develop for Android devices (Android SDK version 34.0.0)
[✓] Chrome - develop for the web
[!] Visual Studio - develop for Windows (Visual Studio Community 2022 17.4.3)
  X The current Visual Studio installation is incomplete. Please reinstall Visual Studio.
[✓] Android Studio (version 2022.2)
[✓] VS Code (version 1.80.1)
[✓] VS Code, 64-bit edition (version 1.80.1)
[✓] Connected device (3 available)
[✓] Network resources

! Doctor found issues in 1 category.

C:\Users\Eng. Daniel Vieira>
```

# Instalação do SDK Android

## 1º Fazer o download do Android Studio

The screenshot shows the 'Download' section of the Android Studio page. At the top, there's a navigation bar with links for 'Essentials', 'Design & Plan', 'Docs', and 'Google Play'. On the right, there are search, language ('Português'), and account ('Android Studio') buttons. Below the navigation, the 'ANDROID STUDIO' section is visible, featuring tabs for 'Download', 'Android Studio editor', 'Android Gradle Plugin', 'SDK tools', and 'Preview'. A large 'Android Studio' logo with the tagline 'Get the official Integrated Development Environment (IDE) for Android app development.' is prominently displayed. A green button at the bottom left says 'Download Android Studio Flamingo' with a download icon. To the right, the Android Studio IDE interface is shown, displaying Java code for a Composable function named 'TopicSelection'. The code uses 'rememberLazyGridState' and 'topicSelectionTestTag'. The IDE also shows a preview of a mobile application screen with various UI components like cards and buttons. At the bottom, there's an 'App Quality Insights' section showing a list of issues with details like count, last 60 days, users, and device distribution.

developers

Essentials ▾ Design & Plan ▾ Docs Google Play

Search Português - ... Android Studio Fazer login

ANDROID STUDIO

Download Android Studio editor Android Gradle Plugin SDK tools Preview

# Android Studio

Get the official Integrated Development Environment (IDE) for Android app development.

Download Android Studio Flamingo

Read release notes

```
rowinandroid - main.kt
```

```
ForYouScreen.kt
```

```
@Composable
private fun TopicSelection(
    onBoardingUIState: OnboardingUIState,
    onTopicCheckedChanged: (String, Boolean) -> Unit,
    modifier: Modifier = Modifier
) = tree(sectionsName = "TopicSelection") {
    val lazyGridState = rememberLazyGridState()
    val topicSelectionTestTag = "ForYou:topicSelection"
    TrackScrollViewLazyGridState(lazyGridState, sectionsName = topicSelectionTestTag)

    LazyHorizontalGrid(
        state = lazyGridState,
        rows = GridCells.Fixed(count = 3),
        horizontalArrangement = Arrangement.spacedBy(12.dp),
        verticalArrangement = Arrangement.spacedBy(12.dp),
        contentPadding = PaddingValues(16.dp),
        modifier = modifier
            .heightIn(max(240.dp, with(LocalDensity.current) { 240.sp.toDp() }))
            .fillMaxWidth()
            .testTag(topicSelectionTestTag)
    )
}
```

App Quality Insights Google Firebase Crashlytics

Issue	Last 60 days	Events	Users	Details
!hasExtras()	24	7	Google 29% Other 8%	
!app:nowInAndroid/u0a	12	4	Other 8%	
!led to system_server and system app only, unless they are annotated with @Readable.	10	3	Android Versions	
!long:IndoorOrUnknownException - Index 4, size: 4	7	1	Android (12) 58% Android (13) 25% Android (9) 8% Other 8%	
!java.SQLiteDatabaseException - Database or disk is full (code 13 SQLITE_FULL)	1	1		
!long:IndoorOrUnknownException - consensus	1	1		

# Instalação do SDK Android

## 1º Fazer o download do Android Studio

The screenshot shows the official Android Studio download page. At the top, there's a navigation bar with links for 'developers' (with a logo), 'Essentials', 'Design & Plan', 'Docs' (which is underlined in green), and 'Google Play'. To the right of the navigation is a search bar with the placeholder 'Search', a language dropdown set to 'Português - ...', and links for 'Android Studio' and 'Fazer login'. Below the navigation, the page title 'ANDROID STUDIO' is displayed. Underneath it, there are tabs for 'Download' (which is selected and highlighted in blue), 'Android Studio editor', 'Android Gradle Plugin', 'SDK tools', and 'Preview'. The main content area features a large 'Android Studio' logo and a brief description: 'Get the official Integrated Development Environment (IDE) for Android app development.' Below this is a green button with the text 'Download Android Studio Flamingo' and a downward arrow icon. Further down, there's a link 'Read release notes' with a document icon. The bottom half of the page contains a screenshot of the Android Studio IDE interface, showing code in the editor and a preview of an app running on a 'Google Pixel 7 Pro' device. At the very bottom, there's a 'App Quality Insights' section with a chart showing issues over the last 60 days across various devices and operating systems.

developers

Essentials ▾ Design & Plan ▾ Docs Google Play

Search Português - ... Android Studio Fazer login

ANDROID STUDIO

Download Android Studio editor Android Gradle Plugin SDK tools Preview

# Android Studio

Get the official Integrated Development Environment (IDE) for Android app development.

Download Android Studio Flamingo

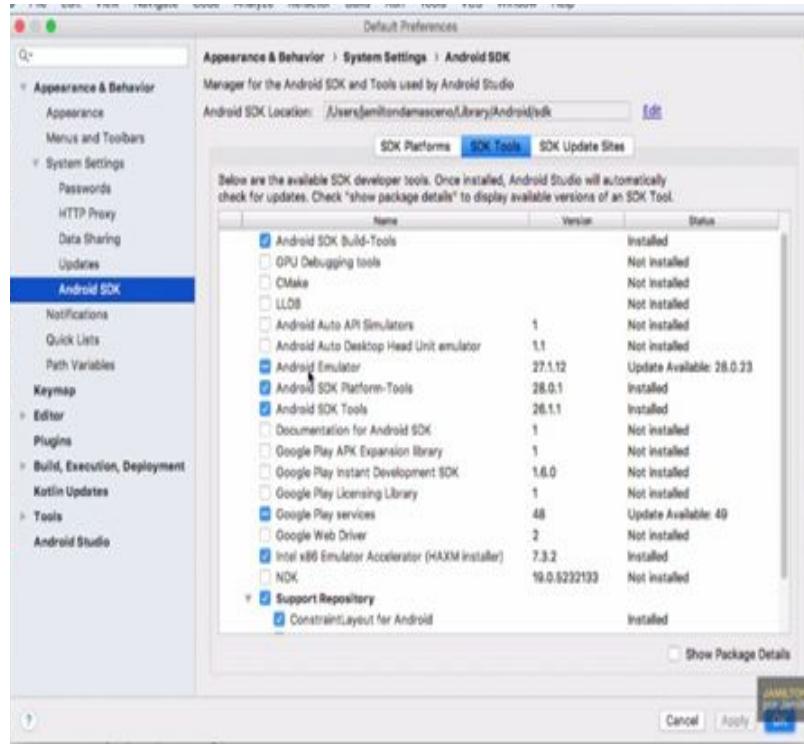
Read release notes

App Quality Insights Google Last 60 days All versions All signal states All devices All operating systems

Issue	Events	Users	Details
! [has extras]	24	7	Google 29% Other 8%
! [has extras]	12	4	Other 8%
! [has extras]	10	4	Android Versions 3%
! [has extras]	7	3	Android (12) 58% Android (13) 25% Android (9) 8% Other 8%
! [has extras]	1	1	Android (12) 58% Android (13) 25% Android (9) 8% Other 8%
! [has extras]	1	1	Android (12) 58% Android (13) 25% Android (9) 8% Other 8%
! [has extras]	1	1	Android (12) 58% Android (13) 25% Android (9) 8% Other 8%

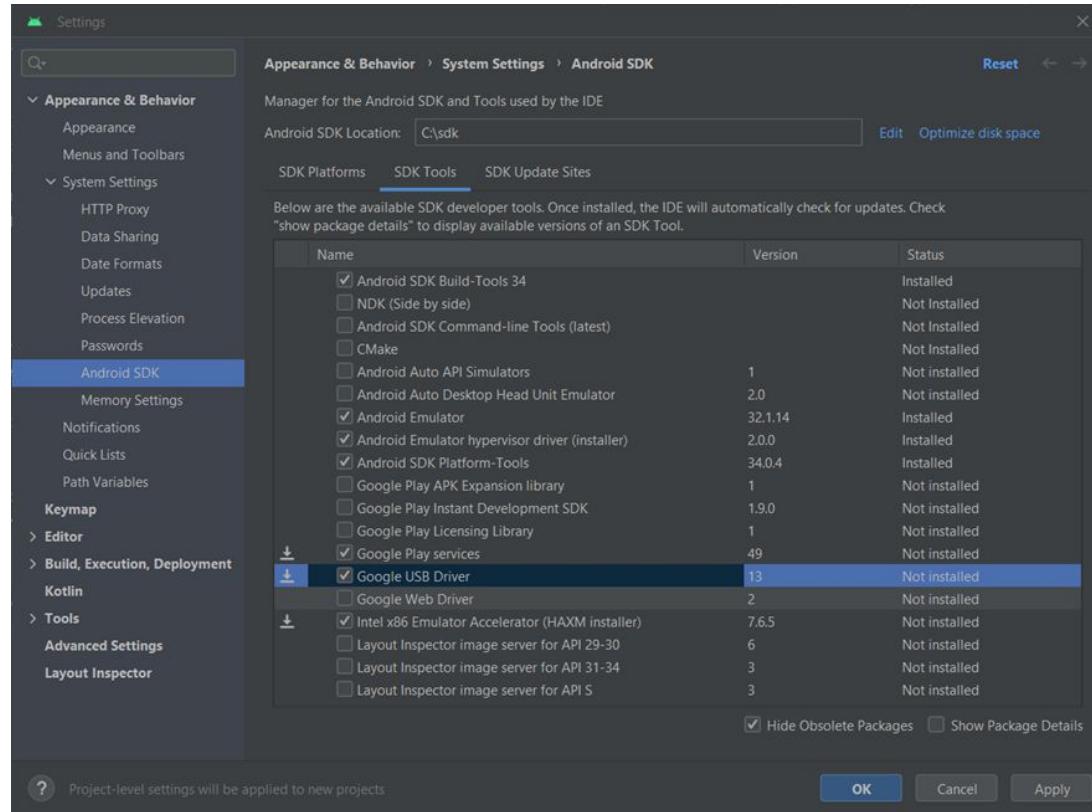
# Instalação do SDK Android

## 2º Instalar o Android Studio



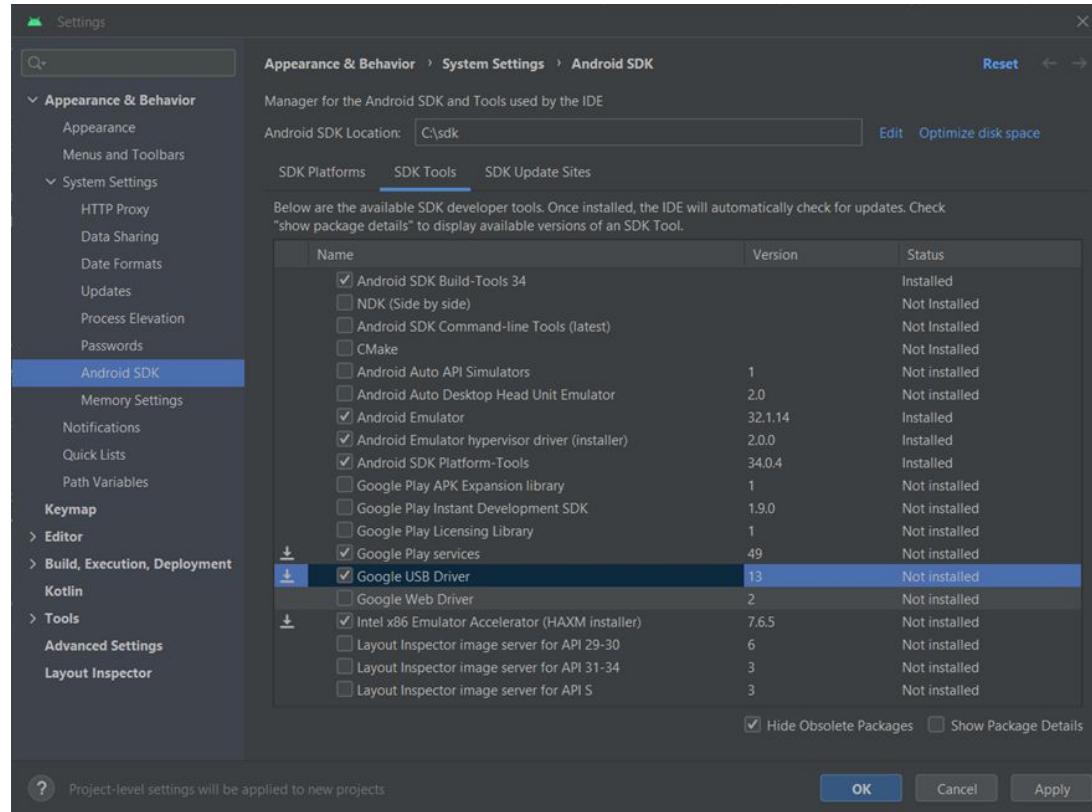
# Instalação do SDK Android

## 2º Instalar o SDK



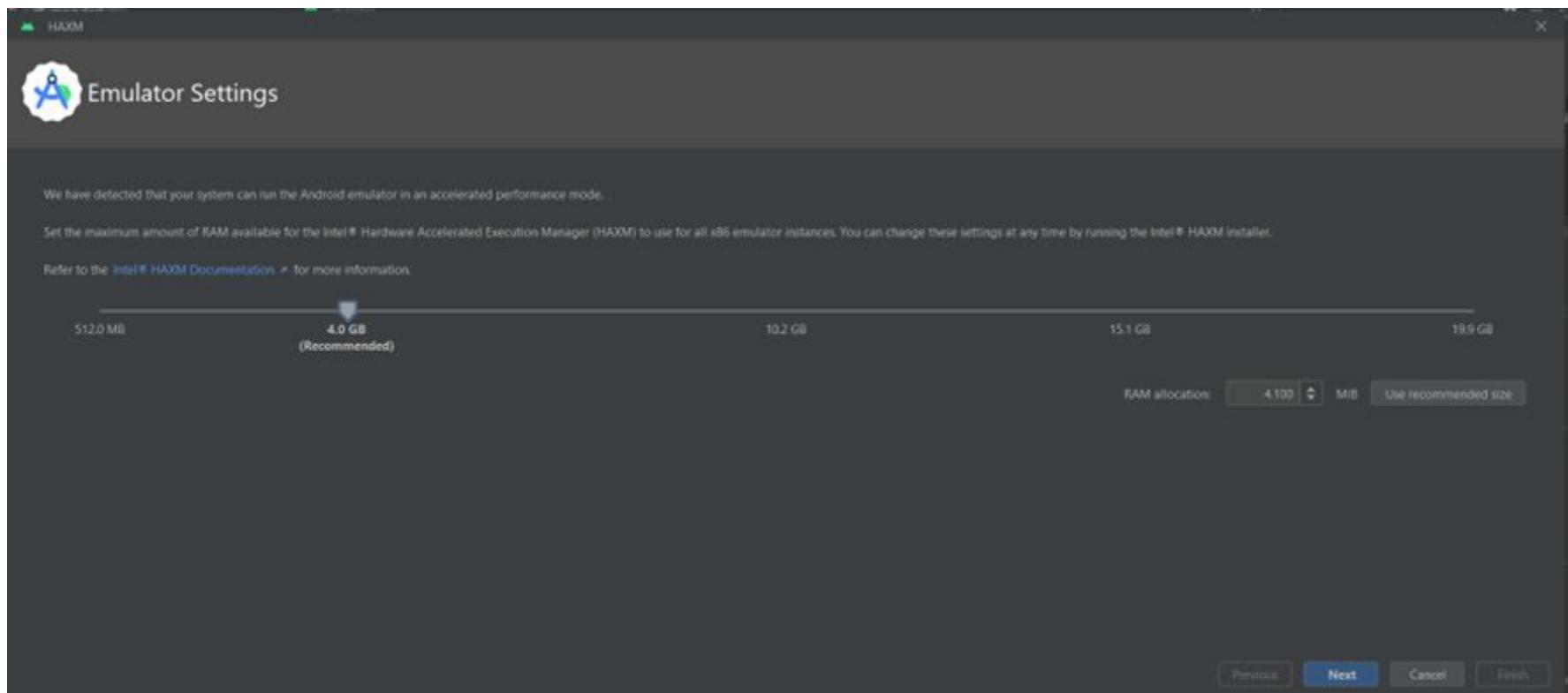
# Instalação do SDK Android

## 2º Instalar o SDK



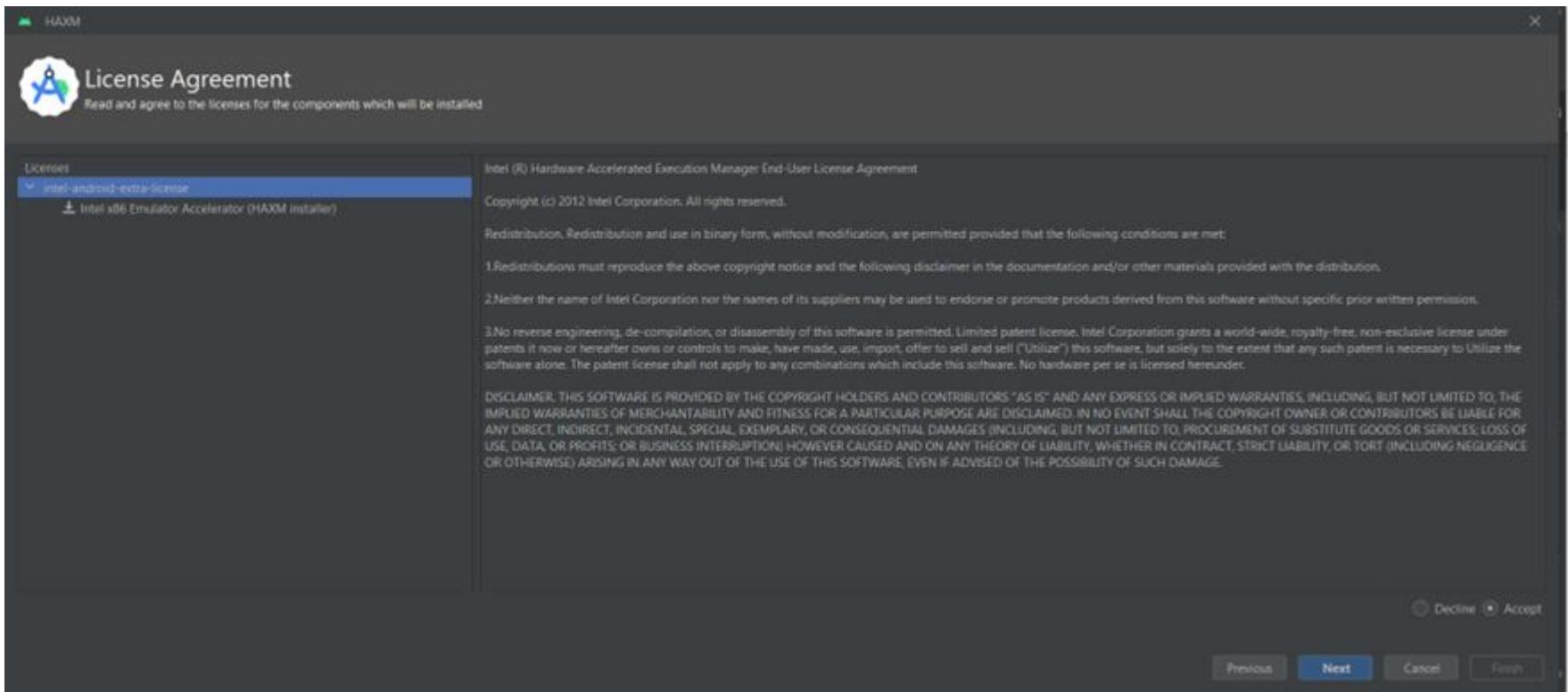
# Instalação do SDK Android

## 3º Instalar o Emulador



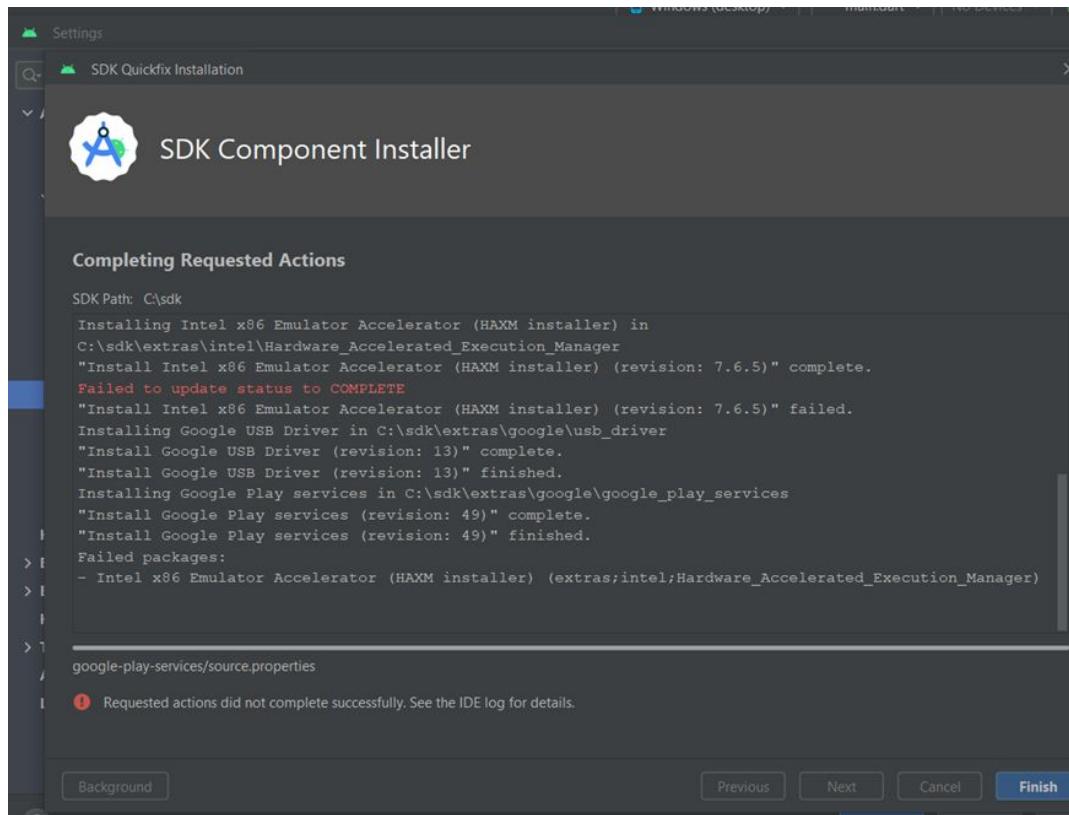
# Instalação do SDK Android

## 4º Concordar com a licença



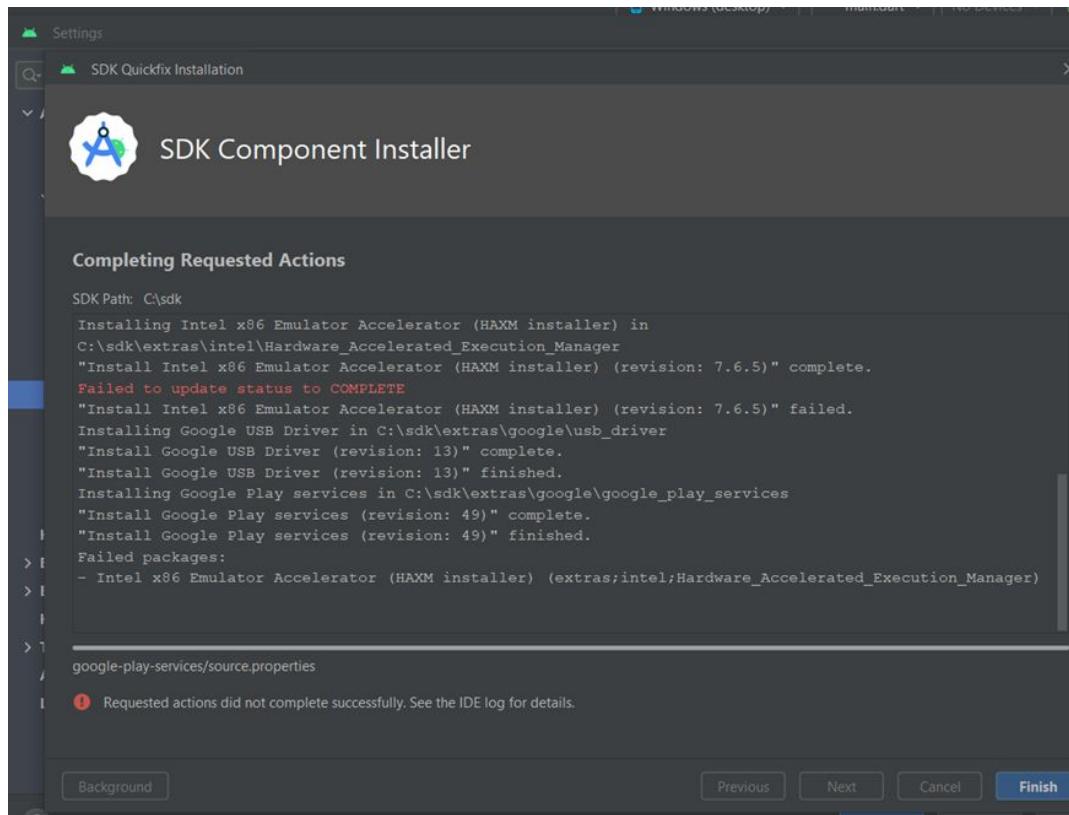
# Instalação do SDK Android

## 5º Instalação do Emulador



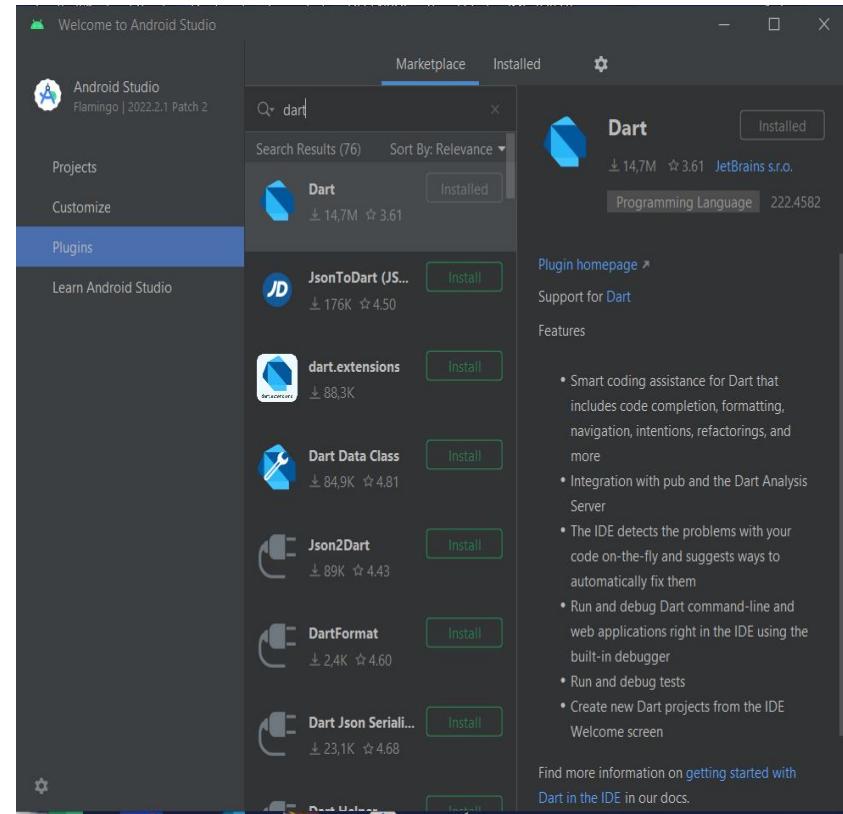
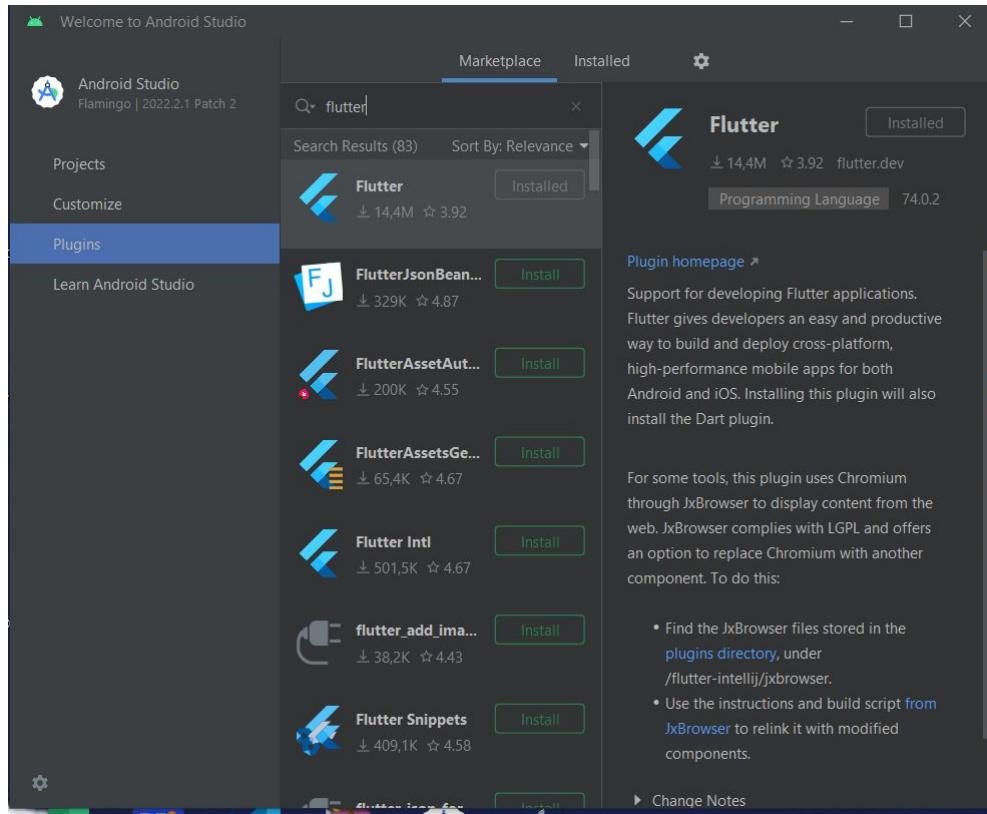
# Instalação do SDK Android

## 5º Instalação do Emulador



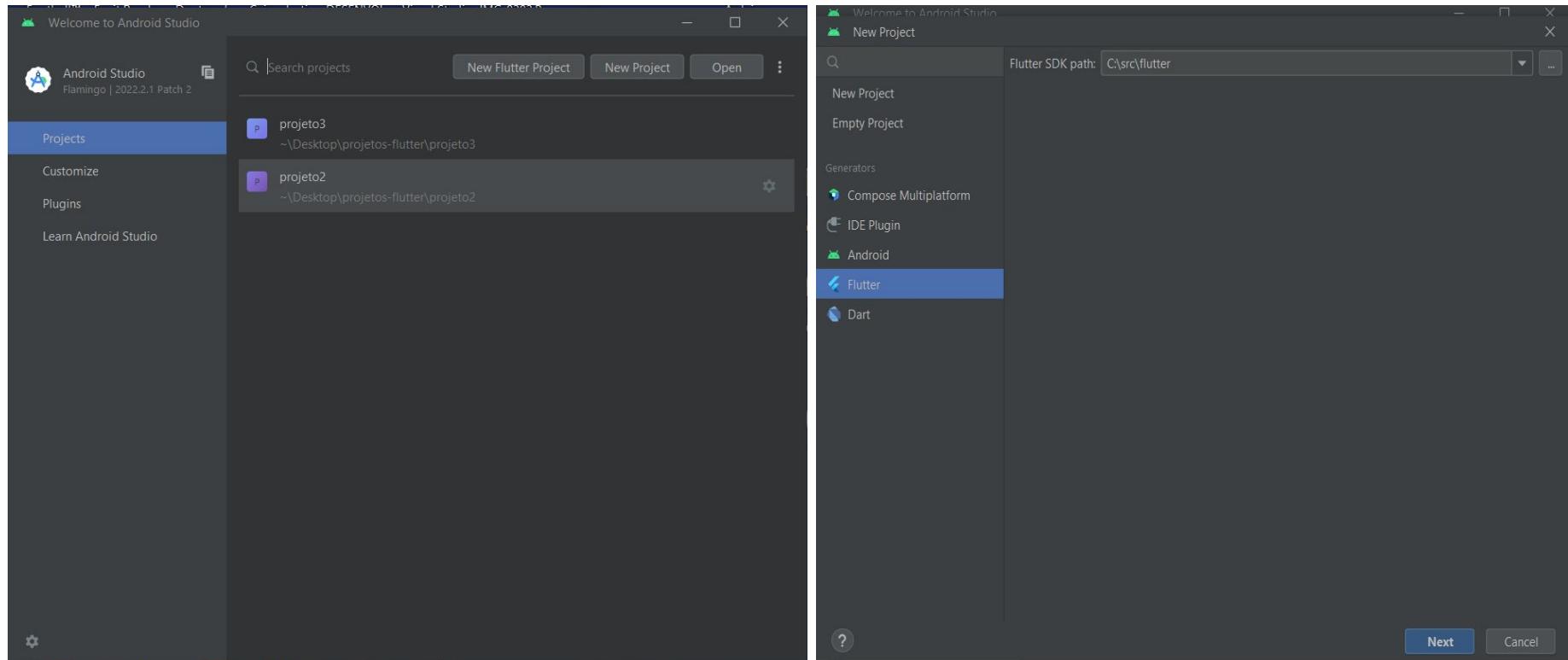
# Instalação do SDK Android

## 6º Instalação dos plugins Flutter e Dart



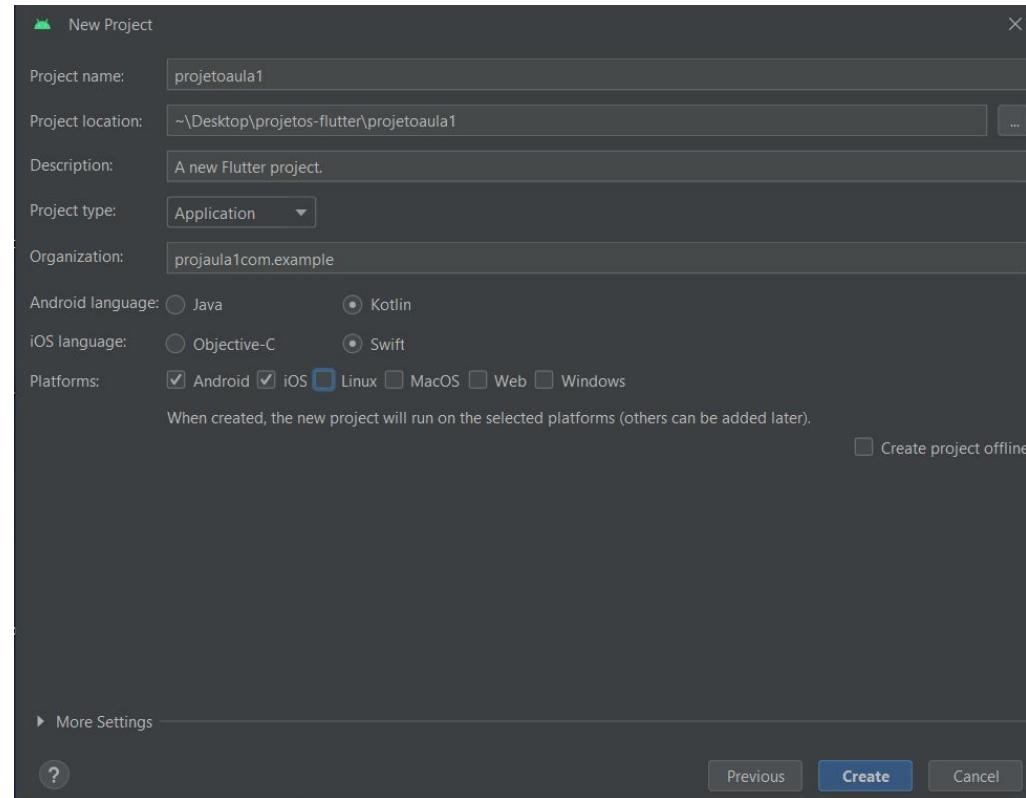
# Instalação do SDK Android

7º Criando um projeto Flutter no Android Studio  
Clicar em New Flutter Project



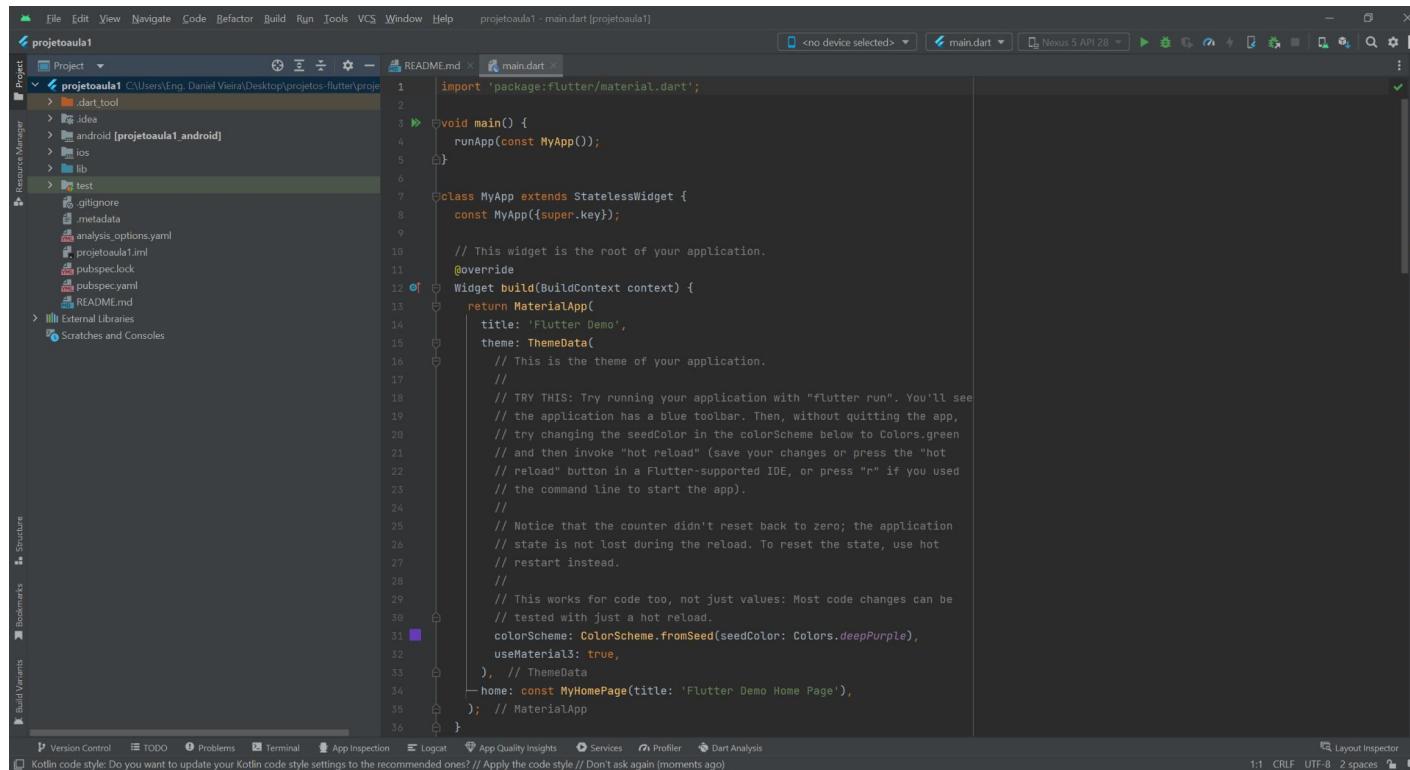
# Instalação do SDK Android

## 7º Criando um projeto Flutter no Android Studio Clicar em New Flutter Project



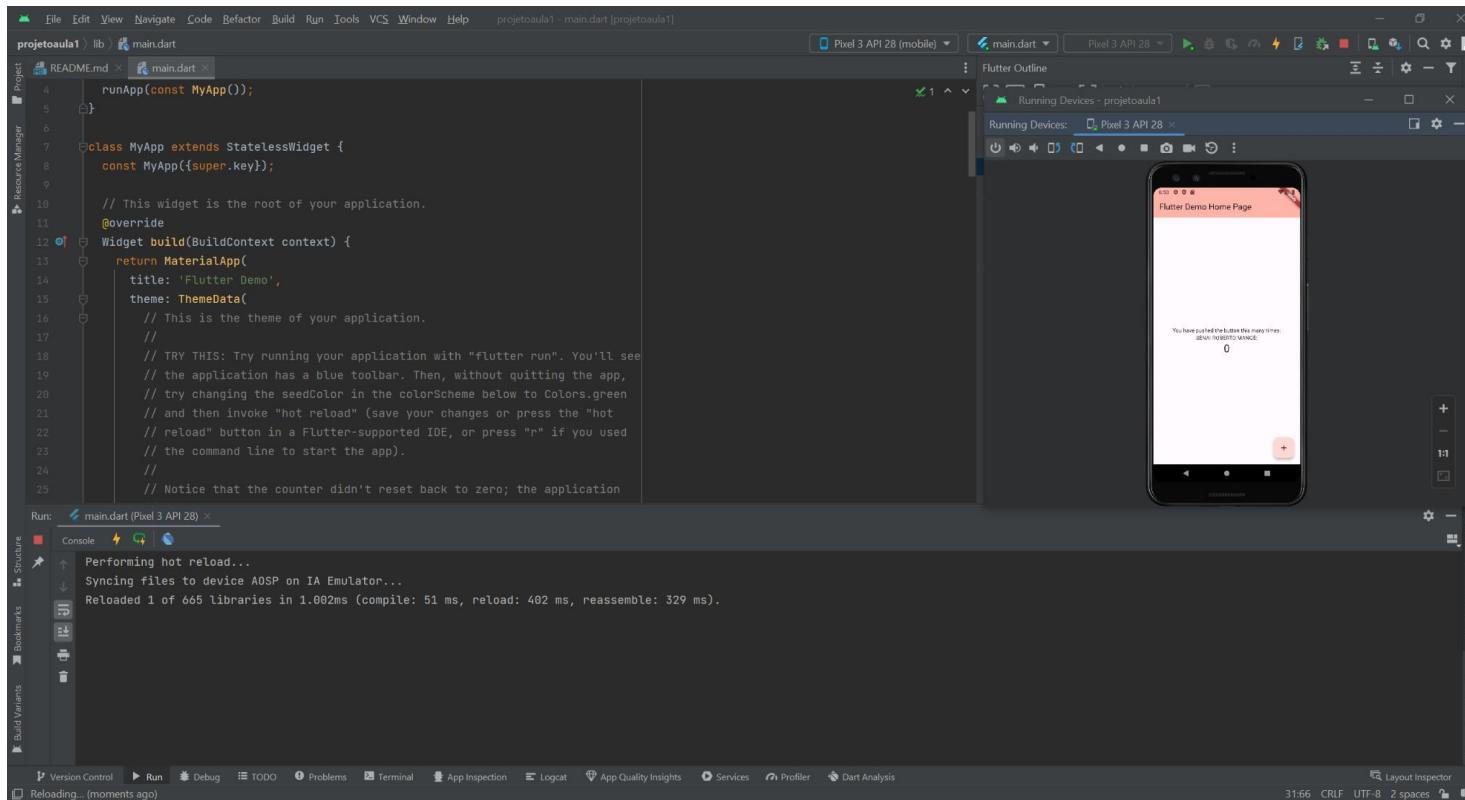
# Instalação do SDK Android

## 7º Criando um projeto Flutter no Android Studio Clicar em New Flutter Project



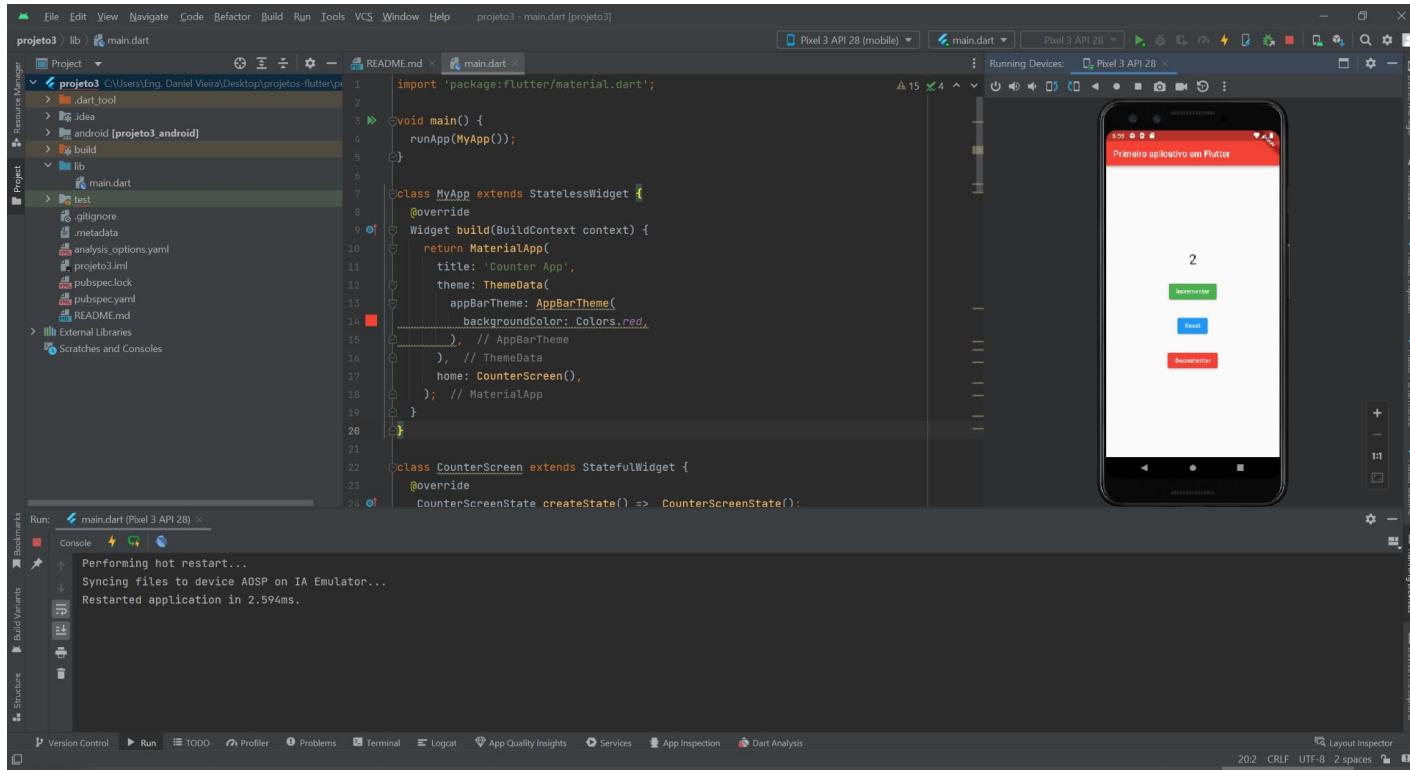
# Instalação do SDK Android

## 7º Criando um projeto Flutter no Android Studio Clicar em New Flutter Project



# Instalação do SDK Android

## 7º Criando um projeto Flutter no Android Studio Clicar em New Flutter Project



# Código Flutter

```
import 'package:flutter/material.dart';

void main() {
  runApp(MyApp());
}

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Counter App',
      theme: ThemeData(
        appBarTheme: AppBarTheme(
          backgroundColor: Colors.green,
        ),
        ),
      home: CounterScreen(),
    );
  }
}
```

# Código Flutter

```
class CounterScreen extends StatefulWidget {  
  @override  
  _CounterScreenState createState() => _CounterScreenState();  
}  
}
```

# Código Flutter

```
void _resetCounter() {  
    setState(() {  
        _counter = 0;  
    });  
}  
void _decrementCounter() {  
    setState(() {  
        _counter = _counter -1;  
    });  
}
```

# Código Flutter

```
@override
Widget build(BuildContext context) {
  return Scaffold(
    appBar: AppBar(
      title: Text('Primeiro aplicativo em Flutter'),
    ),
    body: Center(
      child: Column(
        mainAxisAlignment: MainAxisAlignment.center,
        children: [
          Text(
            '_$_counter',
            style: TextStyle(fontSize: 30),
          ),
          SizedBox(height: 30),
          ElevatedButton(
            onPressed: _incrementCounter,
            child: Text('Incrementar'),
            style: ButtonStyle(
              backgroundColor: MaterialStateProperty.all<Color>(Colors.green),
            ),
          ),
        ],
      ),
    ),
  );
}
```

# Código Flutter

```
@override
Widget build(BuildContext context) {
  return Scaffold(
    appBar: AppBar(
      title: Text('Primeiro aplicativo em Flutter'),
    ),
    body: Center(
      child: Column(
        mainAxisAlignment: MainAxisAlignment.center,
        children: [
          Text(
            '_$_counter',
            style: TextStyle(fontSize: 30),
          ),
          SizedBox(height: 30),
          ElevatedButton(
            onPressed: _incrementCounter,
            child: Text('Incrementar'),
            style: ButtonStyle(
              backgroundColor: MaterialStateProperty.all<Color>(Colors.green),
            ),
          ),
        ],
      ),
    ),
  );
}
```

# Código Flutter

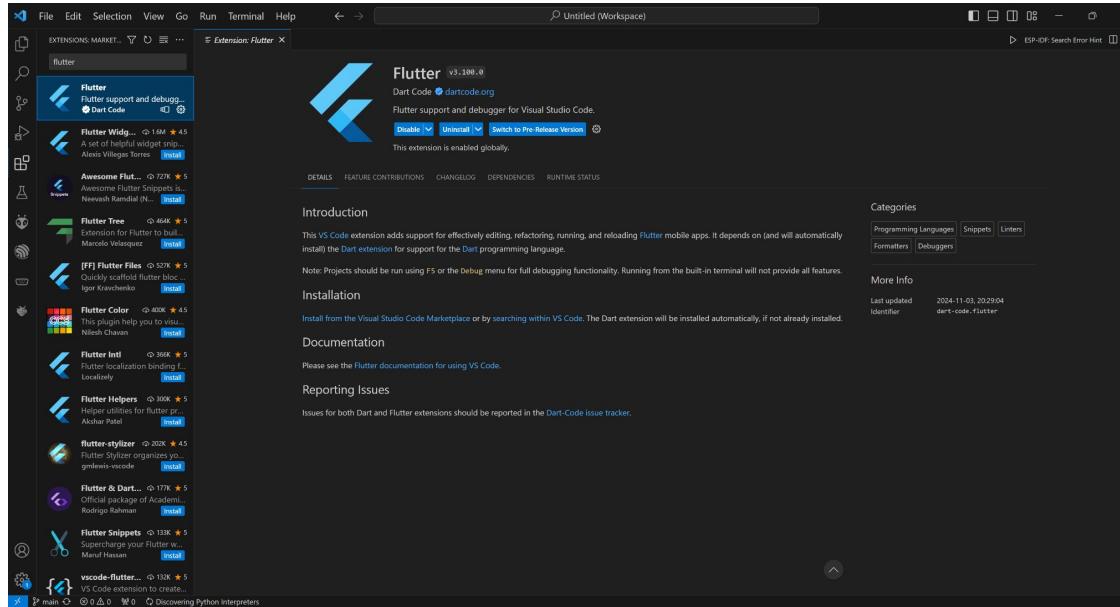
```
SizedBox(height: 30),  
    ElevatedButton(  
        onPressed: _resetCounter,  
        child: Text('Reset'),  
        style: ButtonStyle(  
            backgroundColor: MaterialStateProperty.all<Color>(Colors.blue),  
        ),  
    ),
```

# Código Flutter

```
SizedBox(height: 30),  
    ElevatedButton(  
        onPressed: _decrementCounter,  
        child: Text('Decrementar'),  
        style: ButtonStyle(  
            backgroundColor: MaterialStateProperty.all<Color>(Colors.red),  
        ),  
    ),  
,  
],  
,  
,  
);  
}  
}
```

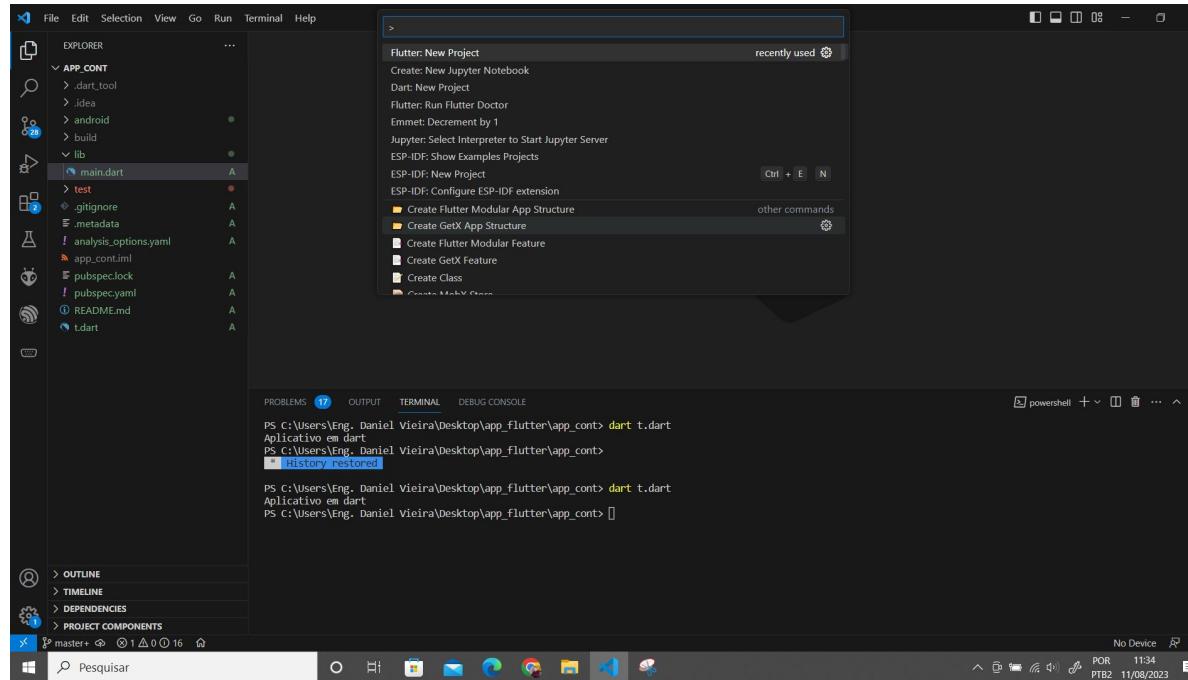
# Criando projeto no VS CODE

## 1º Passo: Instalar a extensão do Flutter no VS CODE



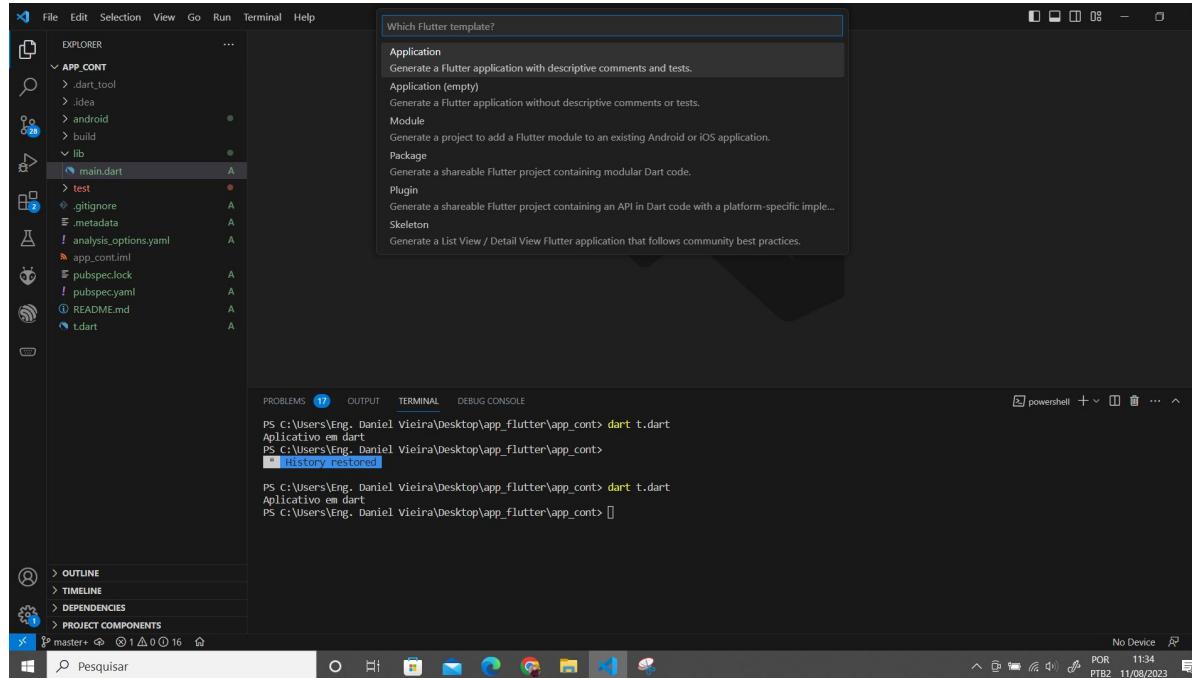
# Criando projeto no VS CODE

## 2º Passo: Apertar F1 e clicar em Flutter New Project



# Criando projeto no VS CODE

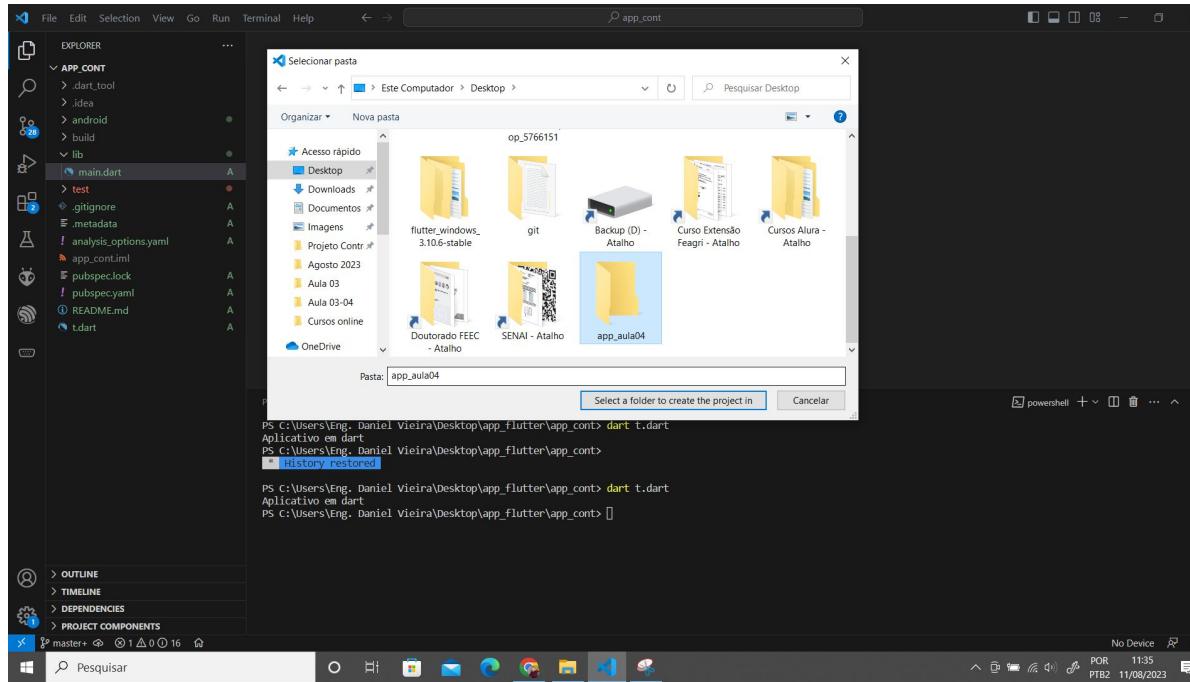
## 3º Passo: Selecionar Application



# Criando projeto no VS CODE

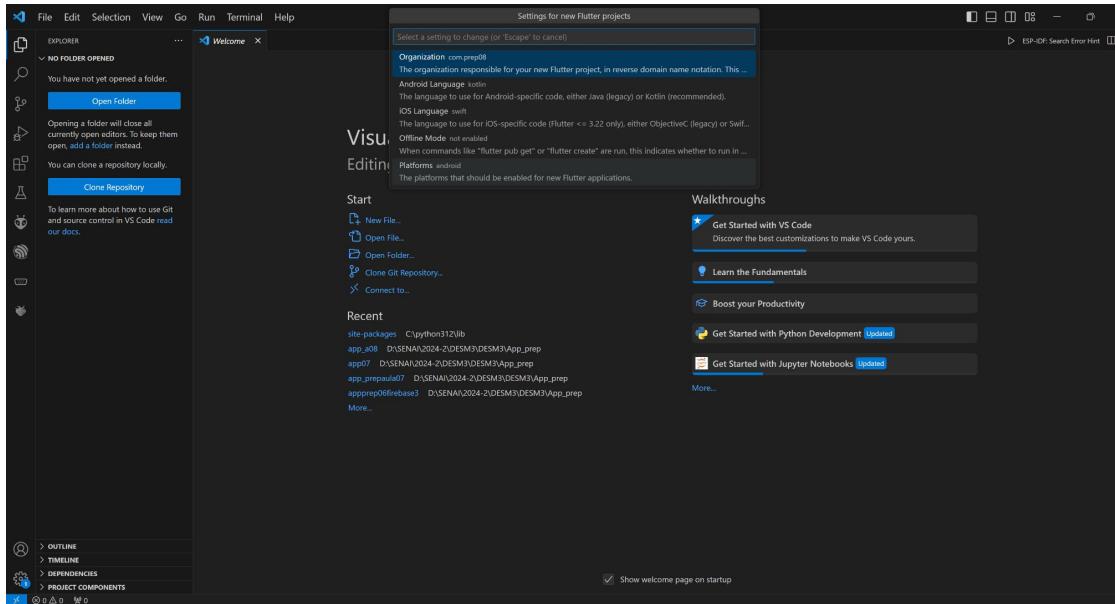
4º Passo: Selecionar a pasta que projeto deve ser salvo

Nao colocar espaço , número e nem letra maiúscula tanto no nome da pasta, quanto no nome do projeto



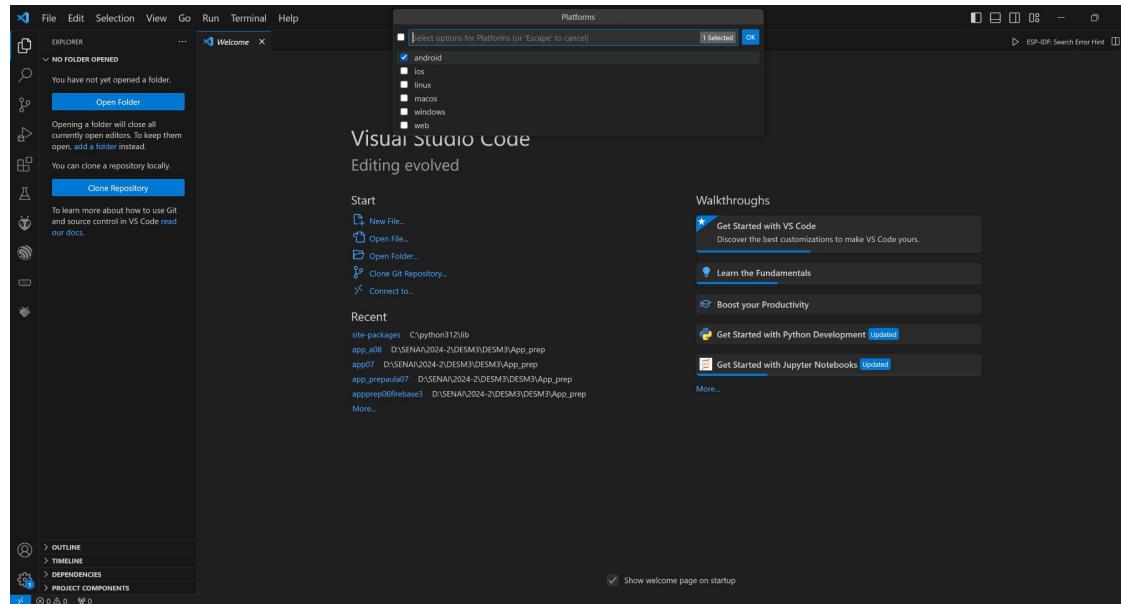
# Criando projeto no VS CODE

## 5º Passo: Selecionar as plataformas do projeto



# Criando projeto no VS CODE

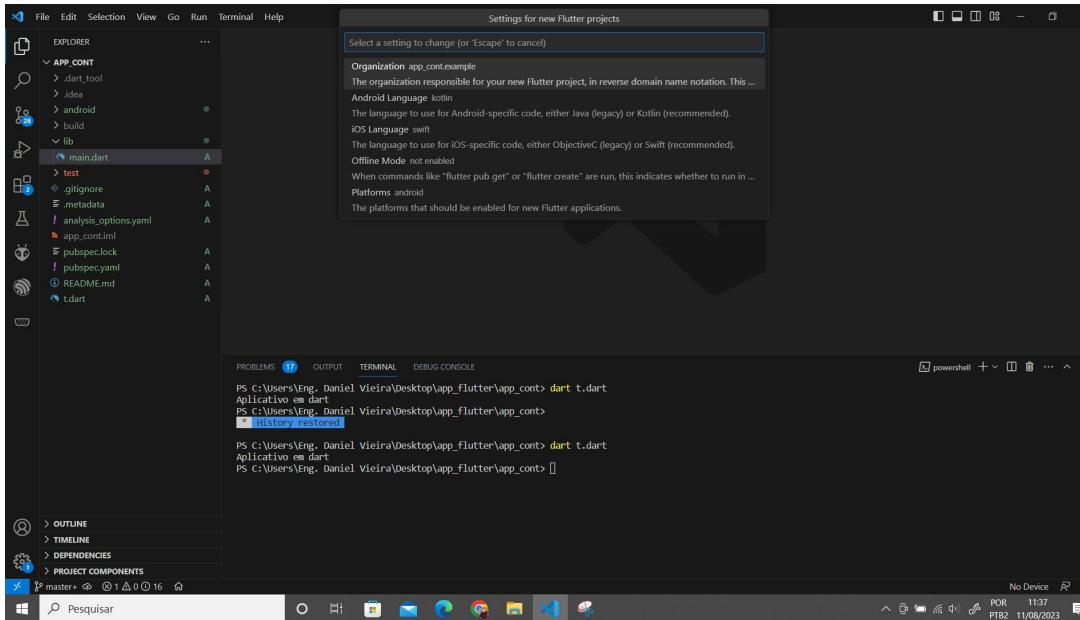
## 6º Passo: Selecionar as plataformas do projeto



# Criando projeto no VS CODE

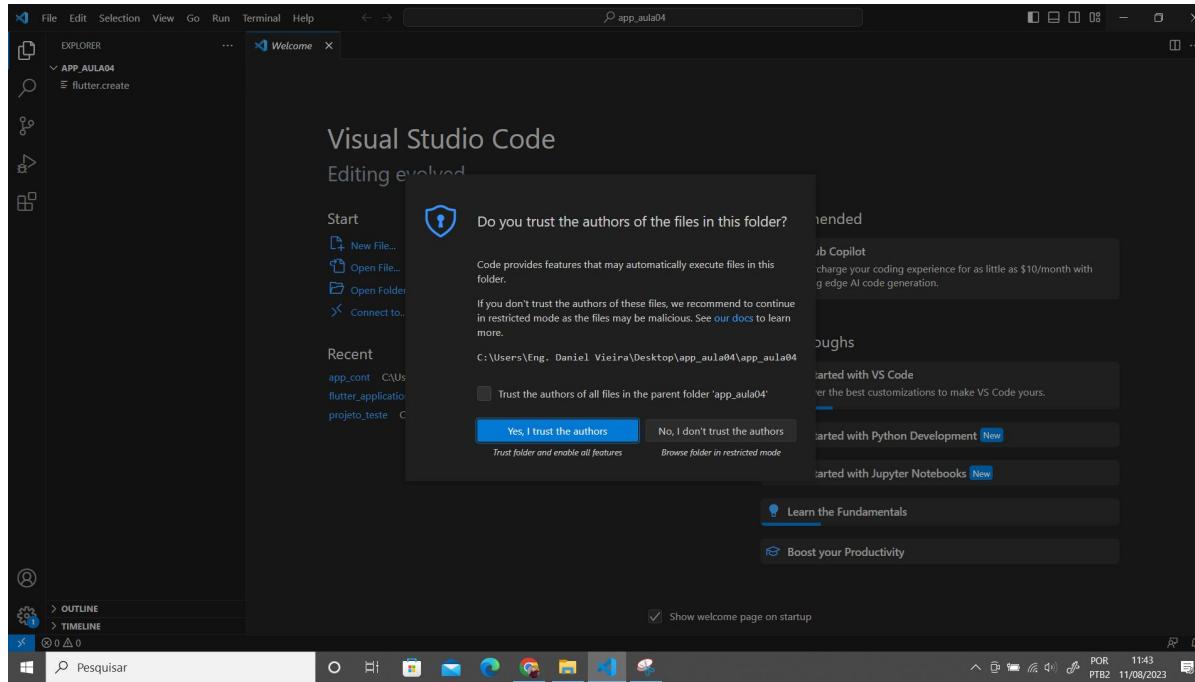
## 7º Passo: Alterar o organization

Organization é o parâmetro que indica o domínio da empresa que desenvolveu o app.  
É um parâmetro importante para publicação do APP em lojas virtuais.



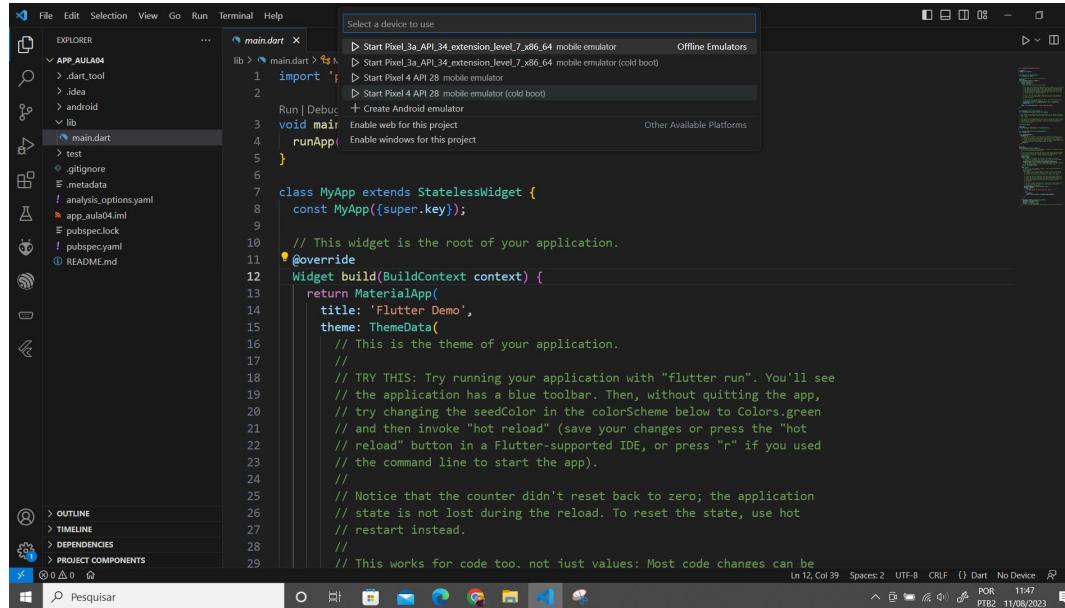
# Criando projeto no VS CODE

## 8º Passo: Finalização do projeto criado



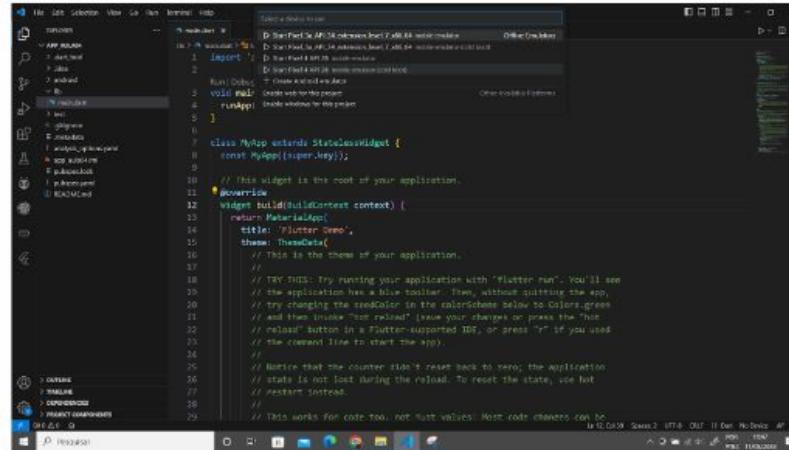
# Criando projeto no VS CODE

## 9º Passo: Escolhendo o emulador para executar o aplicativo



# Criando projeto no VS CODE

## 10º Passo: Emulador aberto - clicar no VS CODE Run Without Debug



A screenshot of the Visual Studio Code interface. The left sidebar shows a file tree with a 'lib' folder containing 'main.dart'. The main code editor area displays the following Dart code:

```
class MyApp extends StatelessWidget {
  MyApp({super.key});

  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Flutter Demo',
      theme: ThemeData(
        // This is the theme of your application.
        //
        // TRY THIS: Try running your application with "Flutter run". You'll see
        // the application has a blue toolbar. Then, without quitting the app,
        // try changing the seedColor in the colorScheme below to Colors.green
        // and then invoke "hot reload" (swipe your finger or press the "hot
        // reload" button in a Flutter-supported IDE, or press "F5" if you used
        // the command line to start the app).
        //
        // Notice that the counter didn't reset back to zero; the application
        // state is not lost during the reload. To reset the state, use hot
        // restart instead.
        //
        // This works for code too; not just values! Most code changes can be
        // detected automatically.
      ),
    );
}
```

The terminal at the bottom shows the command: `flutter build web`.



# Obrigado!

Prof. Me Daniel Vieira

Email: [danielvieira2006@gmail.com](mailto:danielvieira2006@gmail.com)

Linkedin: Daniel Vieira

Instagram: Prof daniel.vieira95

