**Aula VII - DevAppII**

**Introdução ao Flutter**

**O que é?**

**Android Studio & VsCode**

**Como configurar o Flutter no AS**

**Como configurar o Flutter no VsCode**

**Estrutura de pastas e arquivos de um projeto Flutter**

**Material Design & Cupertino**

**O que são widgets?**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

***Fontes:***

<https://flutter.dev/>

<https://www.devmedia.com.br/o-que-e-flutter/40324>

<https://material.io/design>

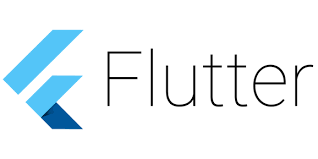
<https://www.treinaweb.com.br/blog/flutter-o-que-sao-widgets-e-qual-sua-importancia>

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Introdução ao Flutter**

**O que é?**

Flutter é o kit de ferramentas de UI do Google para a construção de aplicativos bonitos e nativamente compilados para dispositivos móveis , web , desktop e incorporados a partir de uma única base de código. (fonte: <https://flutter.dev/>)



O Flutter, um framework desenvolvido pelo Google na linguagem Dart, permite o desenvolvimento de aplicações nativas tanto para Android quanto para iOS a partir da composição de Widgets. Podemos utilizar blocos prontos desenvolvidos pela equipe do Google, como botões, cards, menus e muitos outros, para agilizar o desenvolvimento, mas também podemos personalizar e/ou criar Widgets livremente.

O Flutter utiliza uma abordagem até então única para lidar com os componentes nativos de cada plataforma: ao invés de ‘traduzir’ o código para o elemento correspondente (por exemplo, a criação de um botão na tela), o framework implementa os componentes através de um motor de renderização próprio, eliminando assim a necessidade de conversão dos comandos para Android e iOS.

**Sobre o Flutter podemos dizer que:**

* É multiplataforma
* Cria aplicações com ótima performance
* É um dos principais frameworks para desenvolvimento mobile

**Android Studio & VsCode**

|  |  |
| --- | --- |
|  |  |

Podemos usar duas IDEs para desenvolver usando as ferramentas do Flutter. A escolha da IDE é importante pois temos que manipular projetos com dependências e rotinas específicas para o run dele!

Como estamos trabalhando com uma linguagem que permite a criação de aplicações web híbridas podemos fazer a compilação (saída) tanto no navegador quanto no modo mobile.

**Algumas leituras complementares interessantes para que você se inteire entre as diferenças de app nativo, híbrido e web app.**

[Qual é a diferença entre o aplicativo e o web app?](https://www.madeinweb.com.br/blog/qual-e-a-diferenca-entre-o-aplicativo-e-o-web-app/)

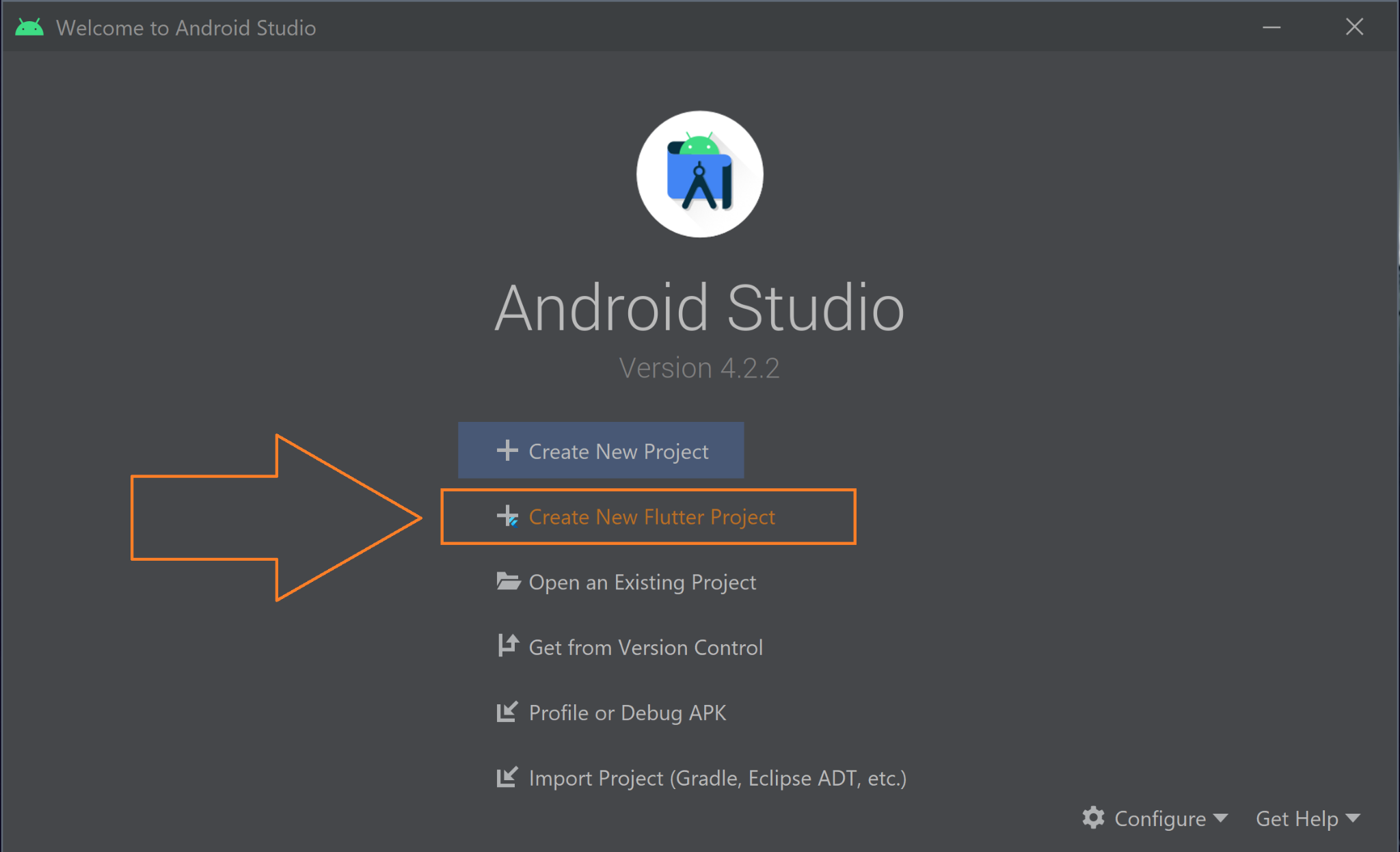
[Entenda o que é aplicativo híbrido e como ele funciona - RankMyAPP](https://www.rankmyapp.com/pt-br/tecnologia/entenda-o-que-e-aplicativo-hibrido-e-como-ele-funciona/#:~:text=Entendendo%20o%20aplicativo%20h%C3%ADbrido,no%20Android%20quanto%20no%20iOS).

[Entenda a diferença de App Nativo, Web App e App Híbrido](https://www.rmcbrothers.com.br/entenda-a-diferenca-de-app-nativo-web-app-e-app-hibrido/#fwdrapPlayer0?catid=0&trackid=3)

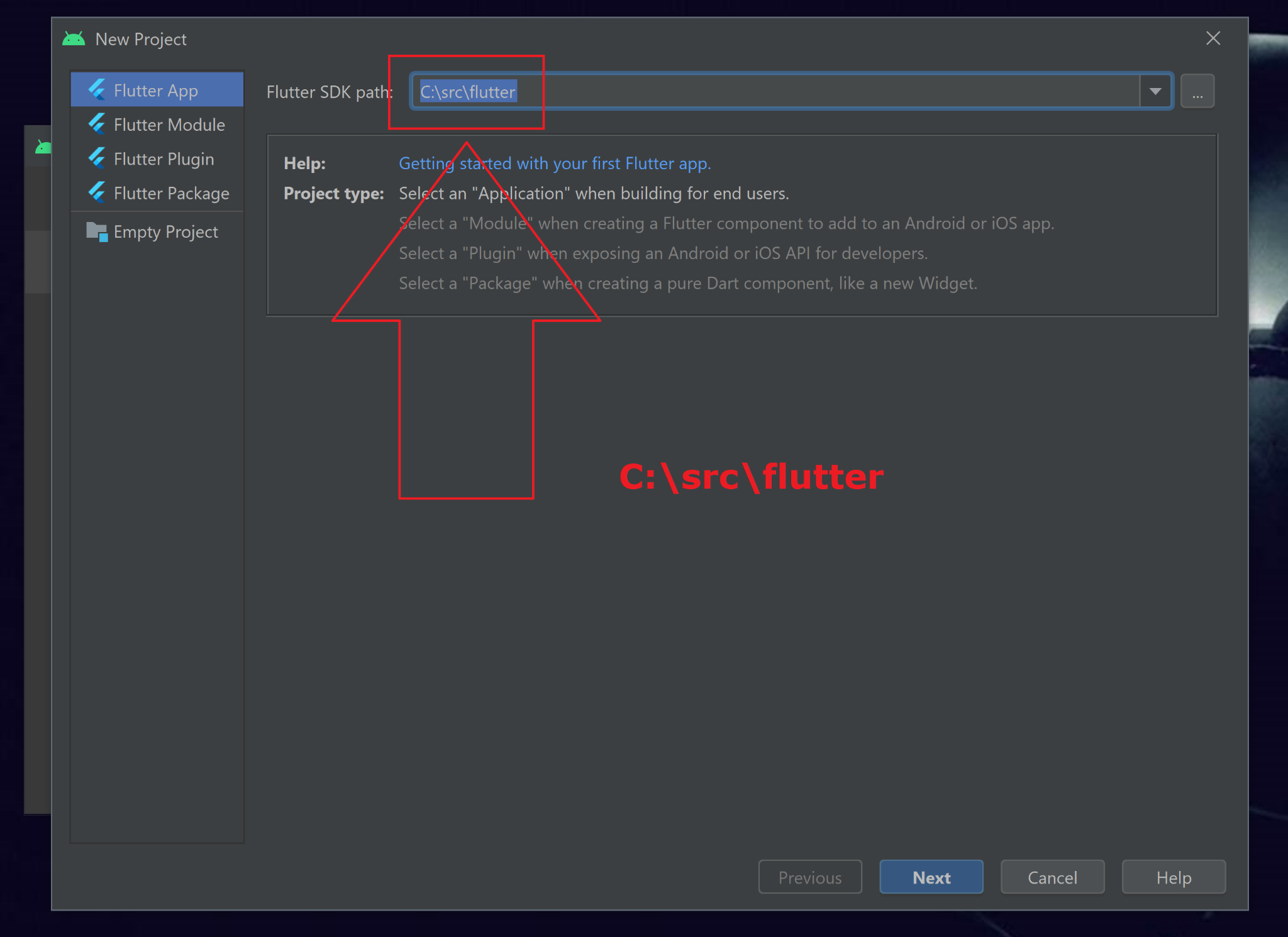
[Aplicativos nativos: o que são e como fazer](https://mundodevops.com/blog/aplicativos-nativos/#:~:text=O%20que%20s%C3%A3o%20aplicativos%20nativos,do%20dispositivo%20m%C3%B3vel%20em%20quest%C3%A3o).

**Vamos começar pelo Android Studio!**

Depois da instalação e configuração do AS + Dart + Flutter, como faremos a criação de um novo projeto baseado em Flutter?

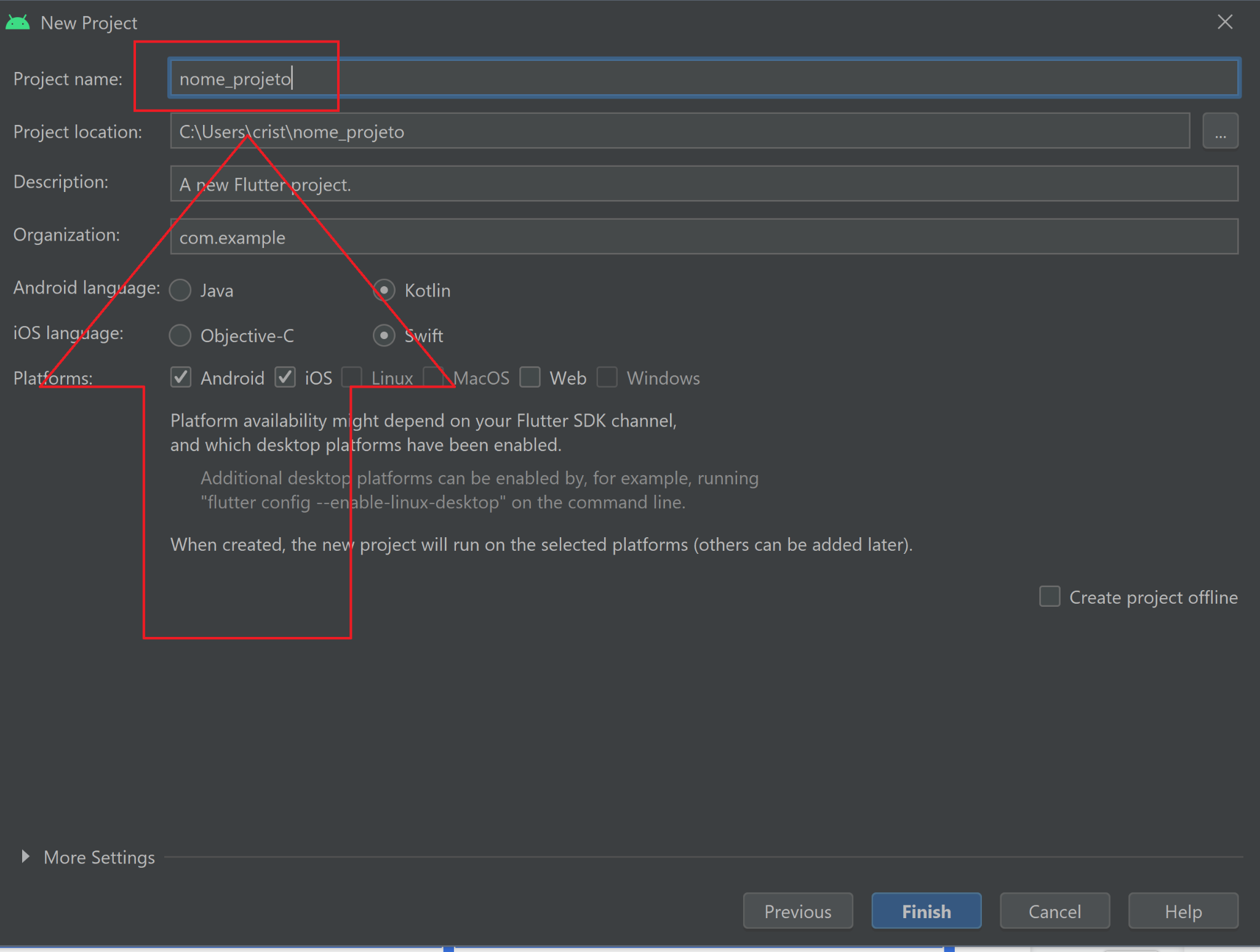
**Passo 1:** 

**Passo 2:**

Escolhemos a opção Flutter App e inserimos o path do SDK que já instalamos!

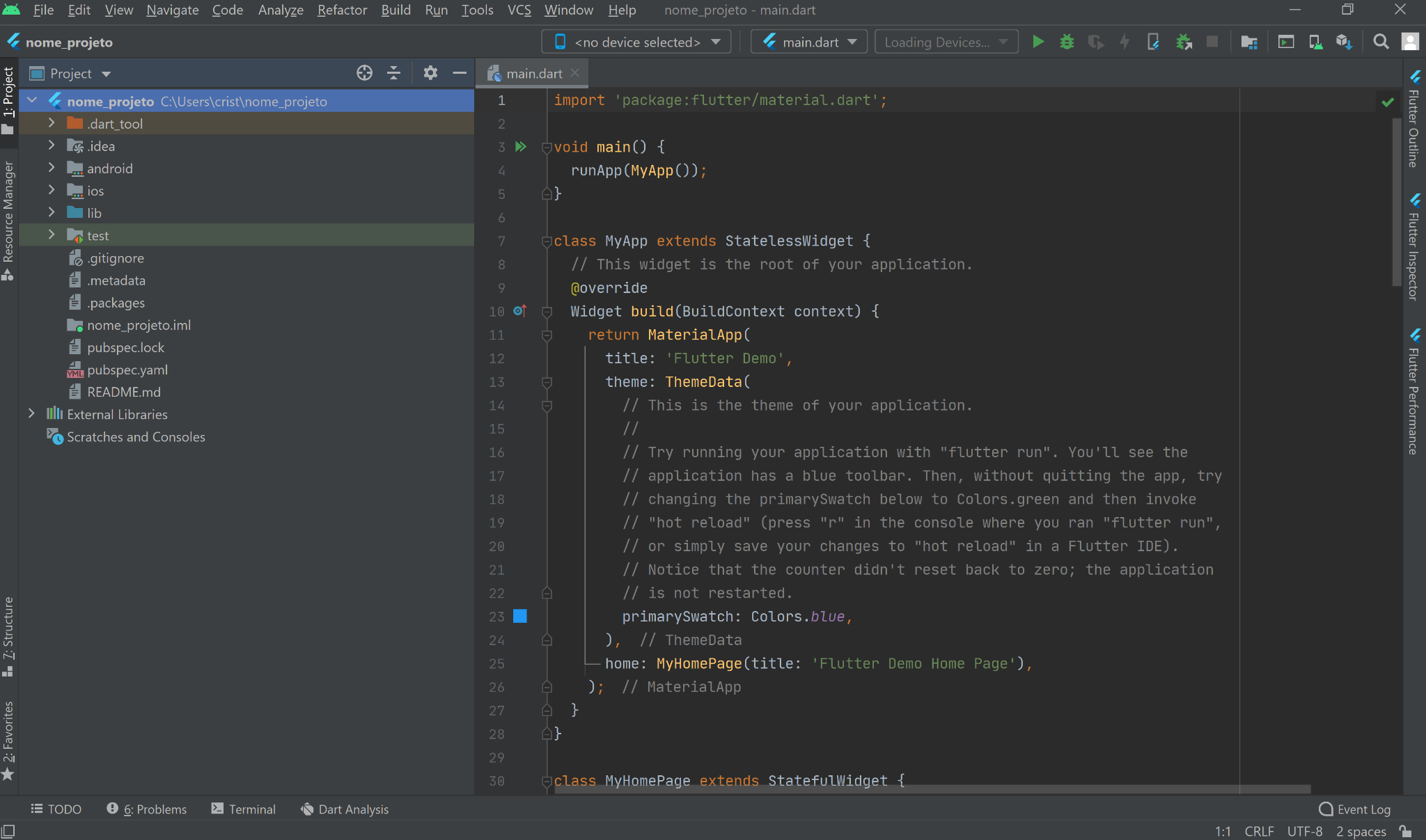
**Passo 3:**

Um detalhe importante aqui: não devemos usar caracteres maiúsculos!



**Passo 4:**

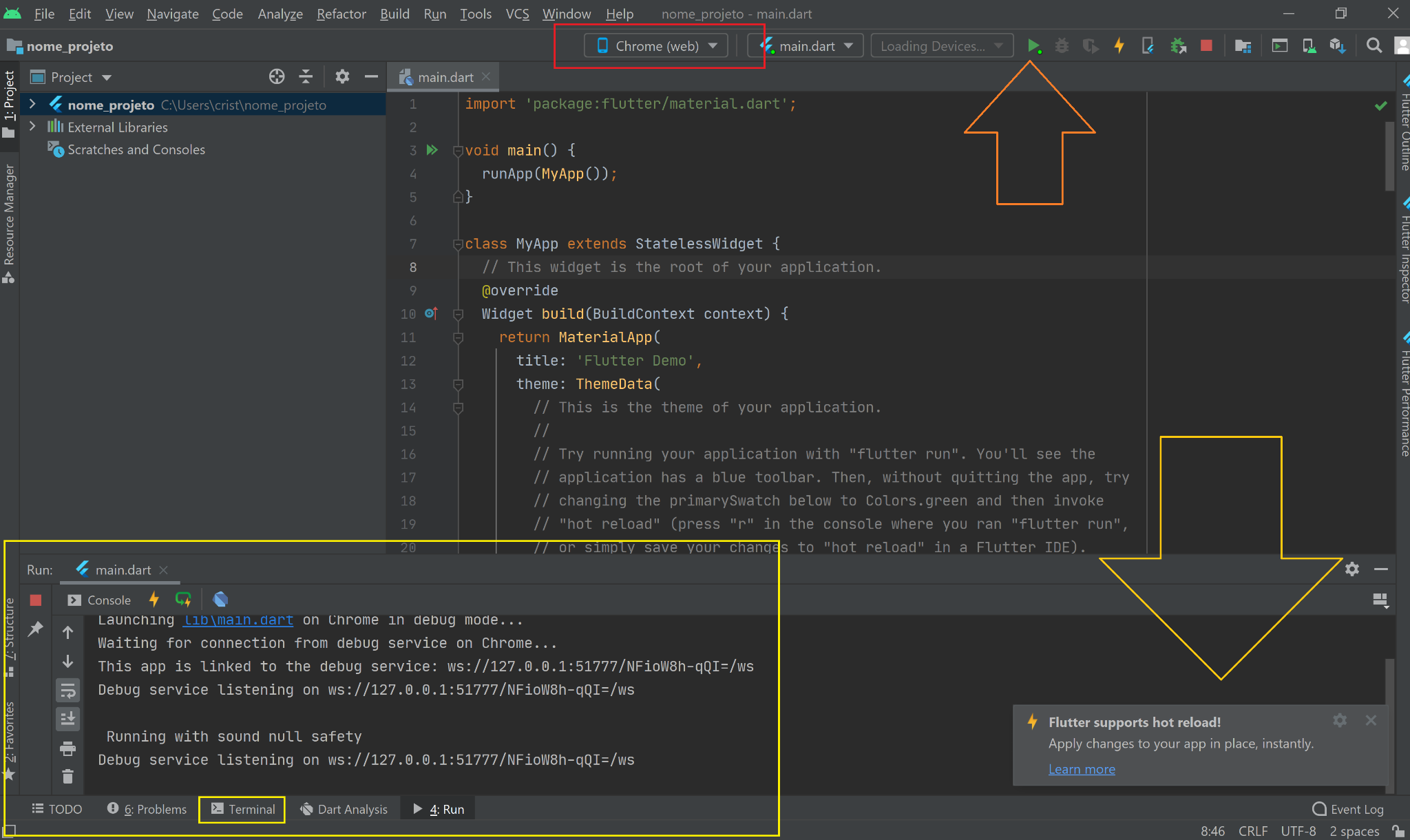
Projeto pronto para poder ser codado. O arquivo que faremos as alterações será (por enquanto) o main.dart.



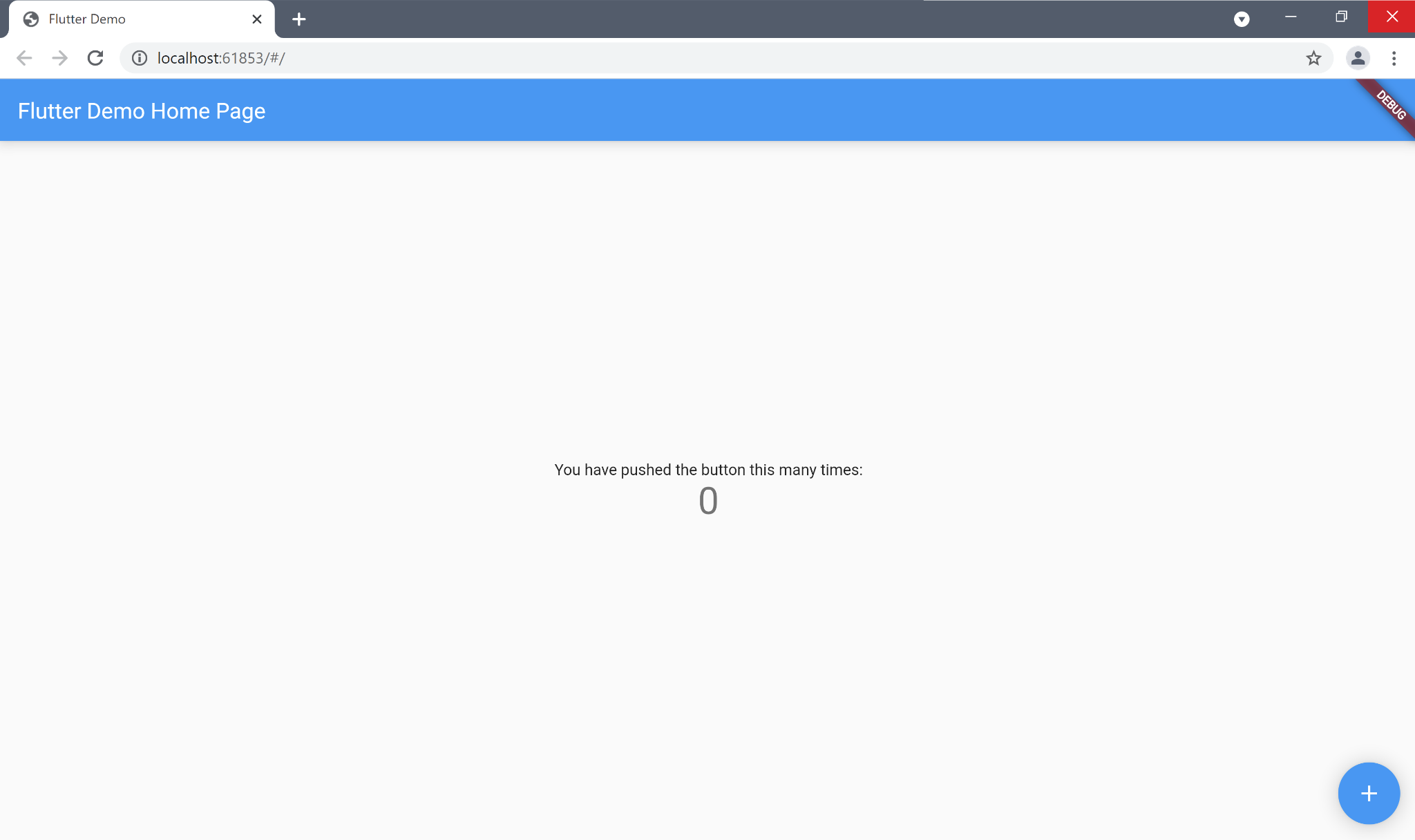
**Para que possamos executar a saída deste projeto:**

**Podemos abrir o projeto, digitar:**

* flutter create
* flutter run
* Ou ainda seguir a execução pela seta de acordo com a imagem abaixo. O Flutter cria um servidor local para exibir as alterações feitas no código e, observem que não há necessidade de atualizações - elas são dinâmicas.



Veremos ….



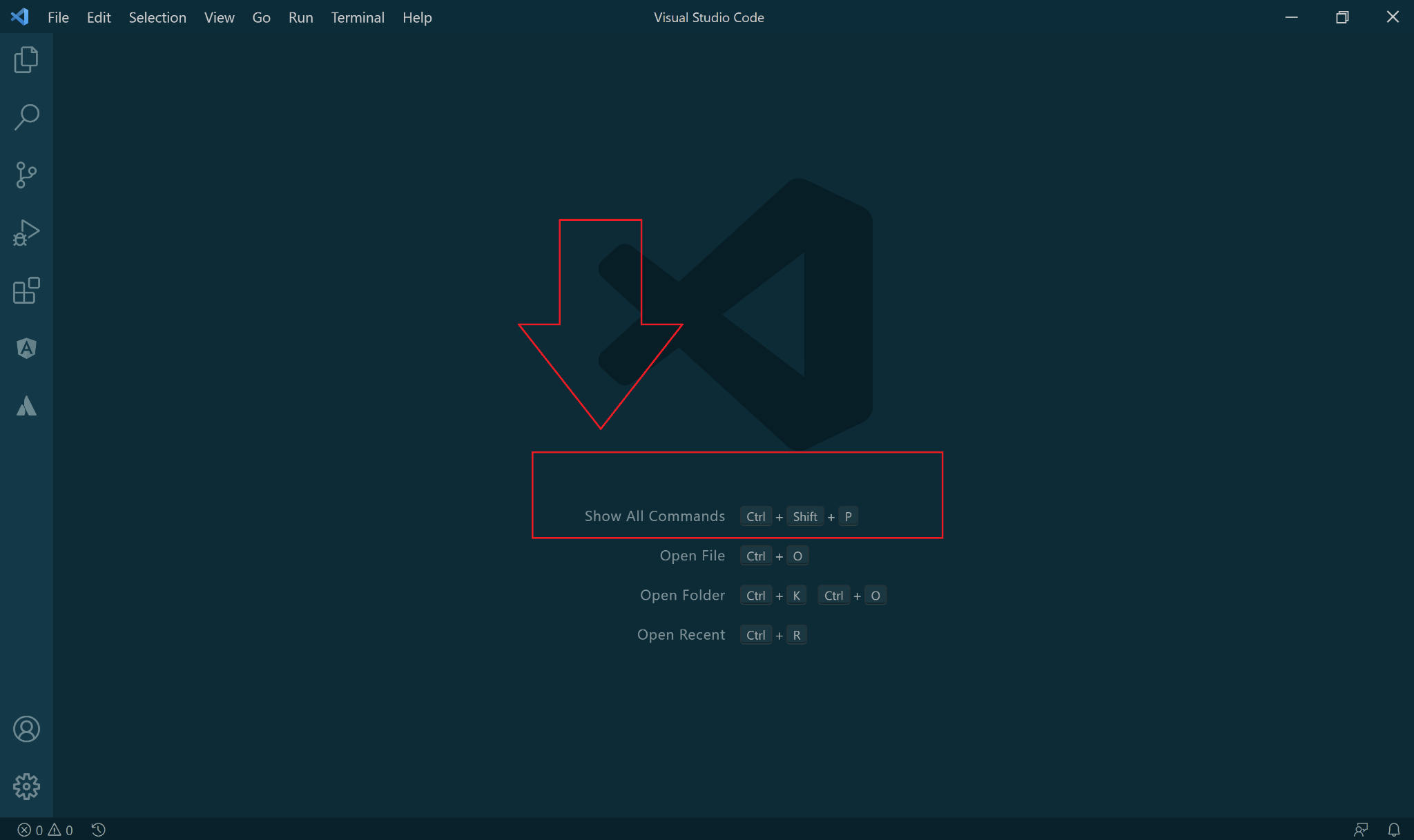
**No VsCode**

Vamos abrir o VsCode depois e instalar as extensões do Dart e Flutter.

|  |  |
| --- | --- |
|  |  |
| Instalação da extensão Dart | Instalação da extensão Flutter |

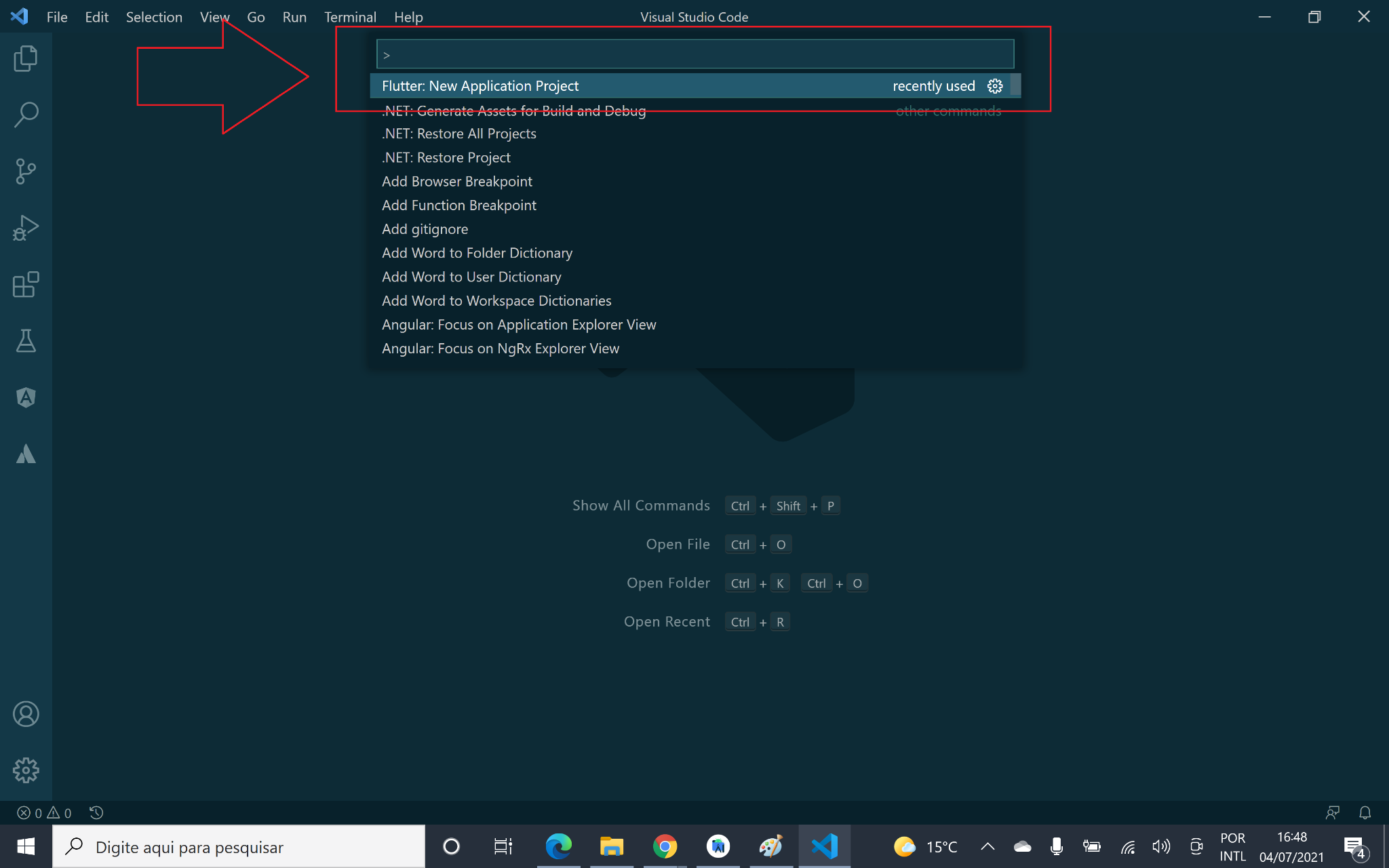
**Passo 1:**

Pressionar as teclas Ctrl + Shift + P



**Passo 2:**

Digitar: Flutter New Application Project



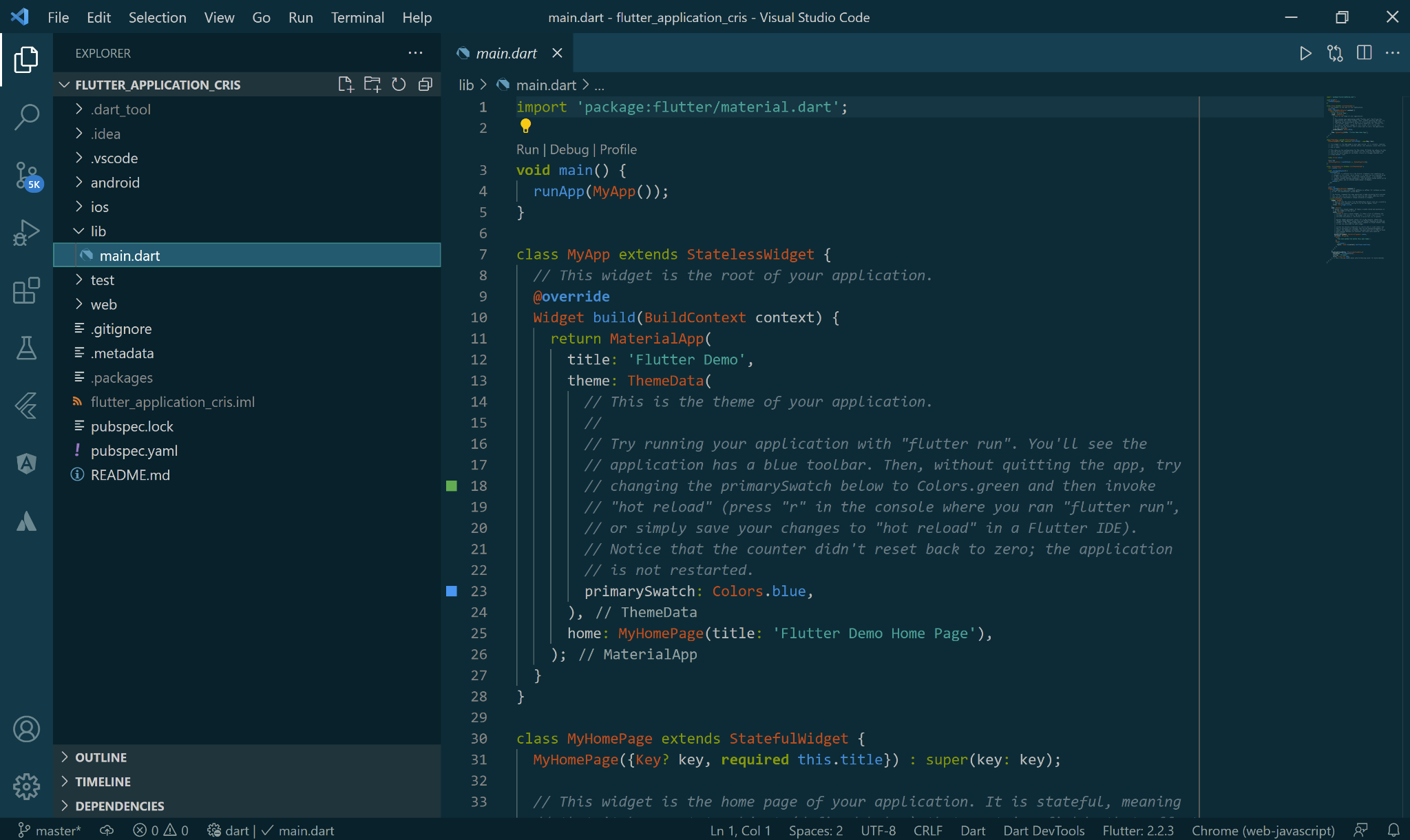
**Passo 3:**

Escolher uma pasta já previamente criada, será onde salvaremos o nosso projeto!



**Passo 4:**

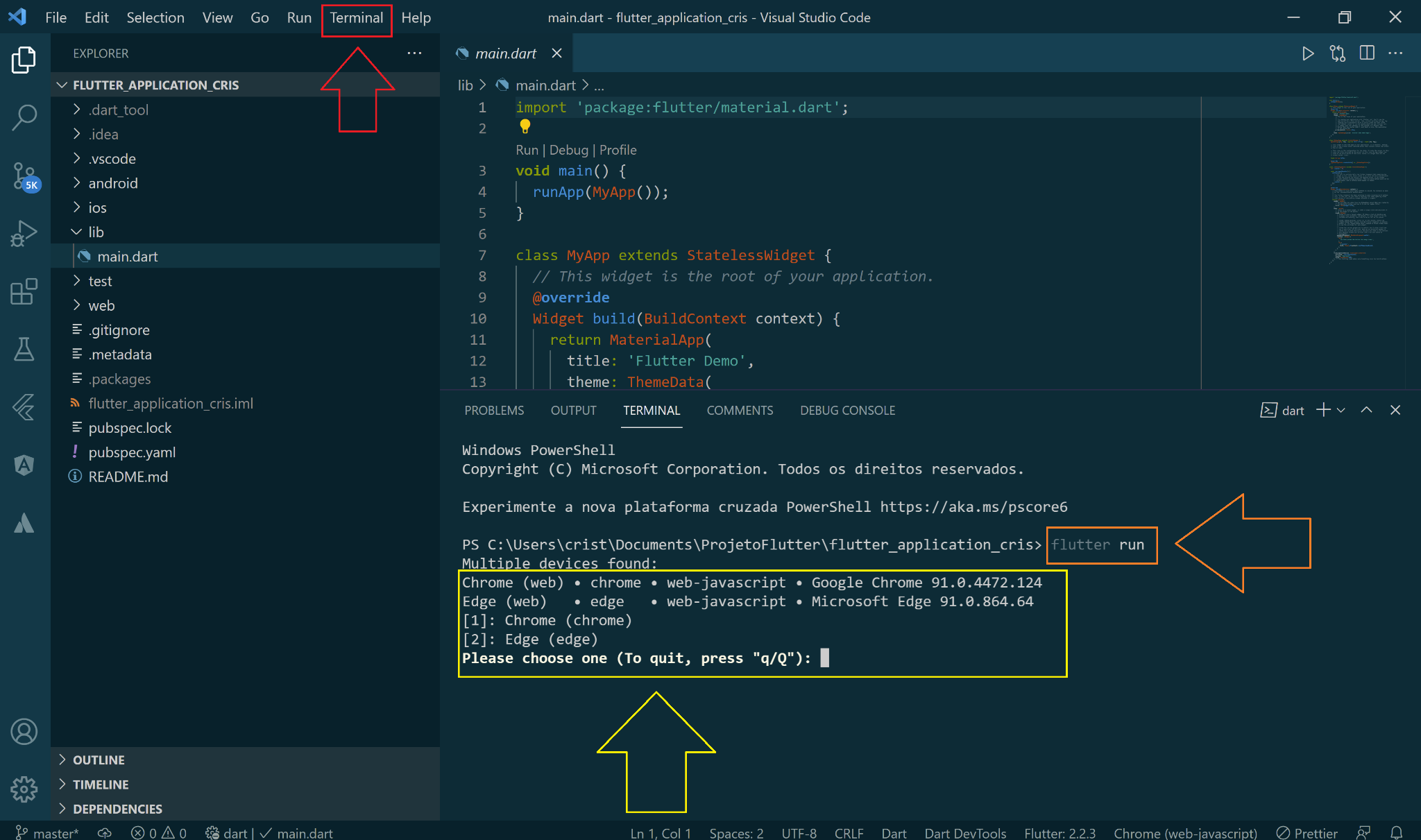
Antes desta tela, o VsCode solicita a inserção do nome do projeto. Não esqueça, não podemos usar caracteres maiúsculos, em função das padronizações do Dart.



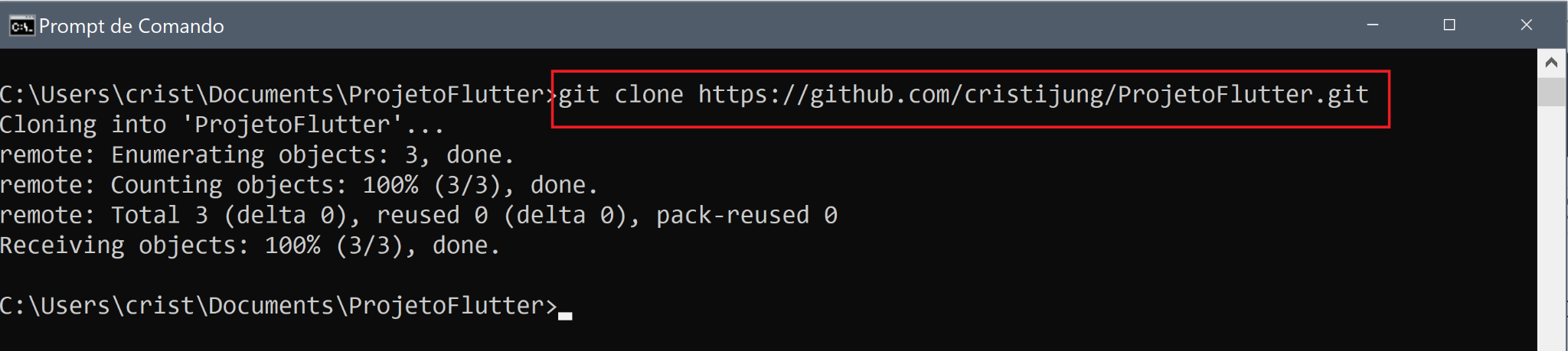
**Passo 5:**

Para rodar a aplicação pelo VsCode:

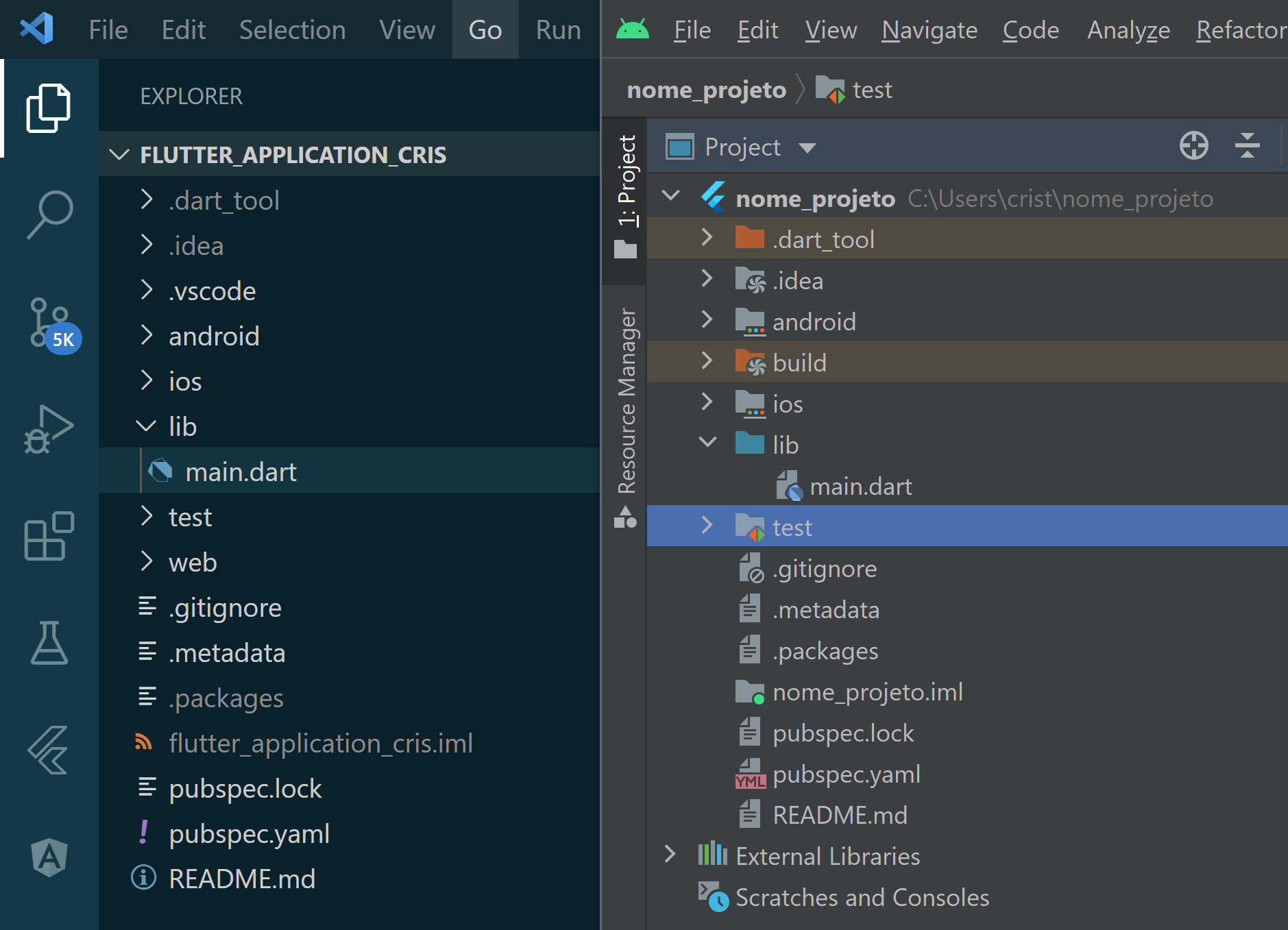
Abrimos o terminal e digitamos **flutter run** e escolhemos o navegador.



**Uma observação**, aqui poderemos escolher uma pasta criada e já com repo clonado do GitHub, o mesmo pode ser feito com o AS. É importante fazermos isso, pois desta forma já estaremos com o ambiente de trabalho já configurado e pronto para comitar no Git.



**Estrutura de pastas e arquivos de um projeto Flutter**



Como podemos ver, um projeto tem características específicas e na comparação das IDEs que estamos trabalhando em aula temos as importantes pastas:

* **android: E**sse diretório, como o próprio nome já sugere, armazena os arquivos ao Android. Ou seja, o código-nativo Java ou Kotlin, os ícones da aplicação, arquivos de configuração, entre outros.
* **ios:** Esse diretório, tem a mesma finalidade do diretório android, porém para para sistemas iOS.
* **lib:** Esse diretório é único diretório (Exceto o test) em que nós devemos fazer alterações. É neste diretório que colocaremos todo o nosso código-fonte e recursos. Por mais que possa parecer estranho, o diretório lib é onde devemos desenvolver nossa aplicação Flutter.
* **res:** Esse diretório contém alguns recursos importantes para a aplicação, como por exemplo strings de internacionalização do aplicativo.
* **test:** Como o próprio nome já sugere, é aqui que colocamos nossos testes unitários.

**Depois os arquivos:**

* **.gitignore:** Se você conhece um pouco de controle de versões com Git, já deve estar familiarizado com esse arquivo. Mas caso nunca tenha ouvido falar dele, esse arquivo é responsável por descrever quais arquivos não devem ser monitorados pelo Git.
* **.metadata:** Dados que o Android Studio usa para rastrear o projeto.
* **.packages:** O Flutter vem com seu próprio gerenciador de pacotes do flutter, o Pub. Esse arquivo mantém o rastreamento das dependências de pacotes do nosso projeto.
* **nome\_do\_projeto.iml:** Esse arquivo recebe o nome do seu projeto, e é responsável por manter as configurações de projeto do Android Studio.
* **pubspec.yaml:** Esse arquivo mantém a descrição de todos os pacotes que são utilizados em nosso projeto Flutter. Ele corresponde ao **package.json** do NPM e também pode ser usado como um importante pipeline cuja função é automatizar a produção de aplicações em um servidor.
* **README.md:** Esse arquivo é um arquivo markdown e pode ser utilizado da forma que desejarmos. Normalmente é utilizado para guardarmos informações ou instruções sobre o projeto. Ele só é visível enquanto em desenvolvimento, não pode ser utilizado para passar instruções para o usuário do aplicativo.

**Material Design**

Material Design é uma linguagem de design desenvolvida pela Google para ajudar as equipes a criar experiências digitais de alta qualidade para Android, iOS, Flutter e web.

O Material Design é inspirado no mundo físico e em suas texturas, incluindo como eles refletem a luz e projetam sombras. As superfícies materiais imaginam os meios de papel e tinta.

[Material Design](https://material.io/design)

[Flutter Cupertino (iOS-style) widgets](https://flutter.dev/docs/development/ui/widgets/cupertino)

Material Design usa componentes do Flutter nativos do Flutter para o desenvolvimento de aplicativos para Android e IOS, vamos conhecer eles!

[Widget catalog](https://flutter.dev/docs/development/ui/widgets)

**O que são widgets?**

Toda interface de um aplicativo desenvolvido com o Flutter é criada com base em um conjunto de Widgets. Basicamente, um widget é um componente visual para definir a interface de um aplicativo.

Pense em uma tela do aplicativo como um LEGO, onde cada widget é uma pequena peça e, ao final, este conjunto de peças representará uma interface completa!

**Para esse comportamento, o Flutter divide seus widgets em duas categorias:**

* **Layout:** Widgets responsáveis por determinar a organização e posicionamentos de outros widgets. Estes widgets servem para delimitar áreas em nossas telas que serão preenchidos por outros widgets;
* **Interface:** Widgets responsáveis por criar componentes visuais que serão exibidos para os usuários. Estes widgets servem para determinar os componentes que comporão a interface do aplicativo.

**Widgets de Layout**

Como dito anteriormente, os widgets de layout servem para determinar a estrutura da interface dos aplicativos e a organização de outros widgets que comporão a tela do app. Estes widgets são de grande importância no Flutter, já que uma estrutura bem definida de uma página é fundamental para a construção de sua interface.

O Flutter possui diversos widgets de layout, alguns dos principais são:

* **Scaffold:** Um widget responsável por criar um layout “padrão” para o app, contendo uma appBar e o conteúdo da tela.
* **Stack:** Widget responsável por “empilhar” widgets na tela do topo ao final da interface.
* **Container:** Widget responsável por comportar novos widgets de layout. O Container é muito utilizado para comportar widgets em sua estrutura separando-os com bordas.

**Widgets de Interface**

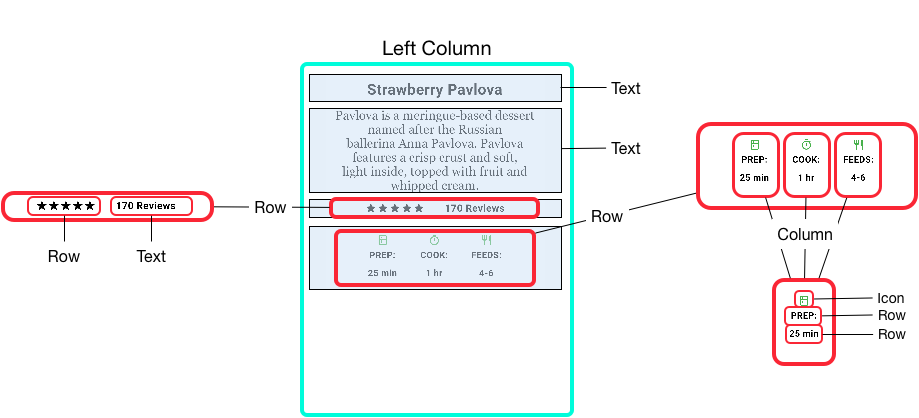
Diferente dos widgets de layout, os widgets de interface servem para criar componentes visuais que serão incorporados ao widgets de layout e, assim, organizados na tela. São componentes visuais, como botões, labels, ícones, textos e qualquer outro componente que compõe a interface de um aplicativo.

De acordo com o site [Treinaweb](https://www.treinaweb.com.br/blog/flutter-o-que-sao-widgets-e-qual-sua-importancia), **o Flutter possui centenas de widgets de interfaces documentados e prontos para serem utilizados em nossos aplicativos. Destes, podemos dar destaque para 2 conjuntos de widgets amplamente utilizados, o Material e o Cupertino. Basicamente, o Material é responsável por estilizar os widgets de interface com base nos padrões do Material Design da Google. Já o Cupertino é responsável por estilizar os widgets de interface com base nos padrões de design do iOS.**

Vale lembrar que o time do Flutter é responsável pela construção e manutenção desses estilos, sendo assim, caso haja alguma mudança nos componentes destas plataformas, é questão de tempo até serem implementados no Flutter.

**Layout + Interface**

Como falamos, os widgets de layout e interface são utilizados em conjunto. Basicamente, a construção de interfaces são compostos por widgets de interface inseridos em widgets de layout, como pode ser visto na imagem abaixo:

*Imagem do site Treinaweb*

O Flutter faz uso de uma hierarquia de elementos visuais, os nossos Widgets, de forma a que eles sejam empilhados um dentro do outro formando uma espécie de árvore de Widgets. Árvore essa que forma a tela visualizada pelo usuário. Na imagem abaixo podemos ver um exemplo desta árvore de widgets do Flutter.

