



PROGRAMAÇÃO II

AULA 1

CONVERSA INICIAL

Prof. Elton Masaharu Sato

Esta etapa está estruturada em duas partes: a primeira contemplará uma introdução ao Flutter, incluindo alguns dos principais conceitos que todo desenvolvedor de Flutter deverá ser familiarizado, bem como alguns dos pontos dos quais tornam o Flutter uma opção interessante para o desenvolvimento de aplicativos.

A segunda parte desta etapa será focada na instalação correta dos programas com um passo a passo detalhado e com imagens para que possamos criar o nosso primeiro projeto.

Antes de começarmos, devemos nos atentar aos recursos que utilizaremos neste texto. Em nosso caso, um computador com os requisitos de *hardware* com, pelo menos, o mínimo necessário para executar *Visual Studio Code* e *Android Studio*, seguindo o Quadro 1.

É plenamente possível utilizar qualquer um dos sistemas operacionais mencionados para o desenvolvimento de Flutter, mas

esse guia utilizará como referência um usuário utilizando o sistema Windows. Os passos a serem seguidos em outros sistemas operacionais serão similares aos apresentados nesta etapa.

Este texto está atualizado para a versão do Flutter março/2022, portanto é possível que haja pequenas alterações nos sites e comandos utilizados para se instalar o Flutter. Este roda em dispositivos Android com Jelly Bean 4.1.x ou superior. Para dispositivos iOS, devem ser iPhone 4S ou mais recentes.

Quadro 1 – Requisitos e Recomendações de *Hardware*

Requisitos	
Sistema Operacional	Windows 7 SP1 ou superior/ macOS 10.14 ou superior/Qualquer distribuição Linux 64-bits que suporte Gnome, KDE, ou Unity DE
CPU	Intel 2ª geração ou superior/AMD Athlon ou superior/Mac ARM, ou com qualquer um dos outros listados
Memória RAM	4 GB (Recomendado 8 GB)
Memória de armazenamento	8 GB (Recomendado SSD para maior velocidade)
Monitor	Recomendado um monitor com resolução de 1920 x 1080 (Full HD) ou superior para melhor visualização

TEMA 1 – UMA BREVE HISTÓRIA DO FLUTTER

Apesar de ser relativamente novo, o Flutter não surgiu do jeito como é conhecido hoje. A sua primeira versão se chamava *Sky*, ou “céu” em inglês, e rodava em aparelhos android em vez de computadores de mesa com Windows, Mac ou Linux.

Essa primeira versão foi apresentada em 2015 durante um evento de desenvolvedores de Dart, a linguagem de programação utilizada pelo Flutter, assim como apresentado pela Figura 1.

Três anos após esse evento, em 2018, o Google havia então anunciado e lançado o Flutter 1.0, a primeira versão estável desse *framework* de desenvolvimento.

Em 2020, o Flutter se consolidou como uma potente ferramenta para desenvolvimento para iOS, tendo melhoras de *performance* de 50%.

E, finalmente, no ano de 2021, Flutter 2 e suas atualizações foram lançados, implementando uma quantidade enorme de ferramentas e bibliotecas, incluindo a última versão do *Material Design*, a mais atual forma de *design* de aplicativos da gigante Google: <https://www.youtube.com/watch?v=PnIWl33YMwA>. Acesso em: 17 maio 2022.

TEMA 2 – O QUE É FLUTTER: LINGUAGEM E FRAMEWORK

Para iniciarmos a conversa, é importante informar que Flutter não é uma linguagem de programação, ele é um *framework* de programação/desenvolvimento. Mas qual a diferença?

De forma lúdica, imaginemos que você, como desenvolvedor de *software*, é o chef de uma renomada cozinha e deve servir os pratos e refeições que seus clientes pedem. Mas você não está sozinho nessa! Você tem ao seu lado poderosos e habilidosos assistentes robôs que te ajudarão a criar e montar os pratos que seus clientes desejam, mas eles precisam que você dê as ordens.

Figura 2 – Cenário do restaurante



Crédito: August_0802/Shutterstock.

Há mais um problema, seus assistentes não entendem português, muito menos inglês ou japonês. Mas eles são muito fluentes em todas as linguagens de programação. Você pode se comunicar com eles usando a linguagem de programação de sua escolha, e quanto mais fluente você for na sua linguagem de programação, melhor você conseguirá comandar seus fiéis assistentes.

Então, vem o primeiro pedido, uma simples porção de batata frita. Você então ordena que seus assistentes peguem as batatas, peguem facas de cozinha e indica quantas vezes eles devem cortar a batata para que tenham a espessura e comprimento ideais para um palito de batata, também os indica o que fazer no caso de uma batata

ser muito pequena ou muito grande, além de batatas com formatos irregulares e tortas.

Ufa! Mas não é só isso, você também precisa informar seus assistentes sobre o processo de fritura, a quantidade de óleo a se colocar na panela, por quanto tempo a panela deve ficar no fogo antes de se colocar as batatas e também quando eles devem retirar as batatas.

Figura 3 – O pedido do cliente



Crédito: DronG/Shutterstock.

Pareceu complicado demais para apenas batatas fritas? Aí é que entra o *framework*, que nada mais é do que um conjunto de ferramentas que ajudam no desenvolvimento de *software*, em nossa analogia da cozinha, o *framework* de desenvolvimento serão utensílios que nos ajudarão a preparar as batatas fritas.

Com o *framework*, então, pediremos para os nossos assistentes colocarem as batatas nos cortadores de batata e, depois, colocarem os palitos na fritadeira automática, pronto!

Viu como foi mais fácil? Espero que esse pequeno exercício de imaginação os tenha ajudado a entender a diferença entre a linguagem de programação e o *framework*, além de enfatizar as vantagens de se utilizar um.

TEMA 3 – O QUE É FLUTTER: CARACTERÍSTICAS

Ok, entendi qual a diferença entre linguagem e *framework*, mas no que o Flutter me ajudará como desenvolvedor? Ótima pergunta, o Flutter foi desenvolvido com alguns objetivos em mente, sendo eles:

- **Rápidos Ciclos de Desenvolvimento:** para sermos capazes de atender à alta e exigente demanda por *softwares*, o Flutter possui ciclos de desenvolvimento extremamente rápidos, utilizando

mecanismos de testes que seus concorrentes têm dificuldade em implementar.

- **Alta Qualidade e Integração:** os aplicativos desenvolvidos em Flutter podem ser lançados tanto para Android quanto para iOS, e um aplicativo lançado em Android parecerá um aplicativo de Android, enquanto um aplicativo lançado em iOS parecerá um aplicativo de iOS.
- **Uma Única Linguagem:** quando se desenvolve para múltiplas plataformas, encontra-se o problema de ter que dominar múltiplas linguagens de programação para cada plataforma. O Flutter utiliza a linguagem Dart para rodar em qualquer aparelho, independentemente de ser Android ou iOS.
- **Substituir os Métodos Menos Eficientes de Produzir para Múltiplas Plataformas:** como o próprio nome diz, vários dos métodos atuais, como React Native, Xamarin e Ionic têm problemas de *performance*, *design*, ou produtividade, várias das coisas que afetarão a experiência do usuário.
- **Curtíssimo Tempo de Compilação:** para os veteranos na área, eles sabem da dificuldade de testar os aplicativos, tendo que gastar muito mais tempo compilando e testando funcionalidades que não precisam mais ser testadas do que de fato programando

e desenvolvendo. Logo mais explicaremos uma das funcionalidades que torna Flutter tão especial.

- **Gratuito e Open Source:** utilizar Flutter é de graça! Não é necessário pagar nenhum tipo de taxa de adesão ou *royalties* por aplicativo produzido por ele. E *Open Source* significa que ele é em ambos seguro, pois qualquer um pode verificar o seu código para procurar por funções nocivas e também é altamente evolutivo, pois a comunidade do Flutter está lá para garantir que o Flutter atenderá as necessidades dos seus usuários, você!

TEMA 4 – O QUE É FLUTTER: COMPARAÇÃO COM OUTROS *FRAMEWORKS*

Vimos que existem vantagens de usar um *framework* como o Flutter, mas além do Flutter, existem outros *frameworks* de desenvolvimento e, nesta seção, focaremos nas diferenças entre o Flutter e sua concorrência.

Mas, então, por que usar Flutter em vez das outras alternativas? O Flutter possui algumas características únicas que a fazem especial e não há nenhum outro *framework* que tem essas mesmas capacidades.

1. **Pixel por Pixel:** o Flutter cria cada pixel da tela por conta própria, logo você como desenvolvedor não precisa se preocupar tanto com alterações em ferramentas fora do Flutter. Outros *frameworks* costumam utilizar os elementos e ferramentas disponibilizadas e, portanto, ficam restritos às suas limitações e são altamente dependentes delas, caso a fábrica decida parar de dar suporte para algumas de suas ferramentas, os *frameworks* dependem delas e podem ter problemas com seus aplicativos.
2. **UI Adaptativo** (*User Interface*, ou Interface do Usuário): o Flutter irá por conta própria emular os elementos de *design* da plataforma para o qual se está desenvolvendo. O desenvolvedor que produz um aplicativo para múltiplas plataformas não precisa refazer os códigos para cada plataforma, ele escreve um código só para todas. Devido a isso, a comunidade Flutter se esforça bastante para que o Flutter proporcione a melhor experiência ao usuário de seus aplicativos, utilizando elementos de interface que são idênticas àsquelas usadas pela plataforma do usuário.
3. **Dart AOT e JIT** (*Ahead Of Time* e *Just In Time*): a linguagem de programação Dart oferece muitas vantagens, então, dividiremos ela em partes. AOT e JIT são formas com a qual o Dart pode ser compilado, enquanto se desenvolve a aplicação, o Dart irá compilar em JIT, que permite rápidas alterações enquanto o

código está rodando. Já a compilação AOT permite alta *performance*, removendo todas as ferramentas de desenvolvedor que não são usadas pela aplicação. Essa incrível capacidade de alterar o código enquanto ele roda se chama *Hot Reload* e veremos mais sobre isso no próximo capítulo, quando focaremos em Dart.

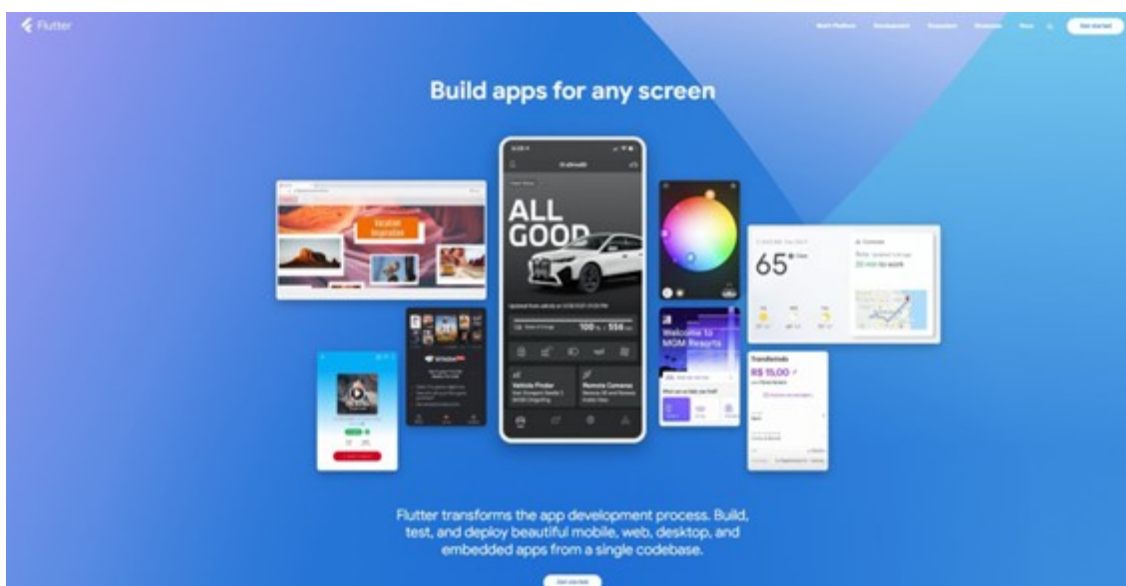
4. **Dart é Simples:** Dart é uma linguagem de fácil aprendizado. Apesar de uma linguagem com muitas funções avançadas, ela é bastante flexível e também muito robusta. Além de ser uma linguagem viva, que está sempre melhorando e com documentação muito bem descrita.
5. **Dart e Flutter Tem o Suporte da Google:** Isso mesmo, a gigante Google dá suporte para ambos o Dart e Flutter, o que as colocam em um patamar de visibilidade, segurança, estabilidade e integração que outras linguagens e *frameworks* não conseguem alcançar.
6. **Competitividade:** altamente competitivo é utilizado por várias das maiores empresas do mundo. Entre elas, estão empresas de gerenciamento bancário, automóveis, *videogames*, *marketplaces* e de tecnologia em ambos *hardware* e *software*.

TEMA 5 – INSTALAÇÃO

Nesta etapa, faremos uma abordagem mais orgânica para o aprendizado de Flutter. Começaremos instalando Flutter para, então, aprendermos a linguagem Dart na prática, sem a necessidade de o leitor abstrair e imaginar como um programa Dart seria codificado.

As imagens e outras referências deste tópico apresentam conteúdos em inglês e é altamente recomendado que o leitor possua inglês básico ou intermediário. Considerando os leitores que não possuem conhecimento em inglês, este tópico apresentará um passo a passo bastante detalhado. **Passo 1:** vamos começar abrindo o *site do Flutter* em <https://flutter.dev>. Acesso em: 17 maio 2022.

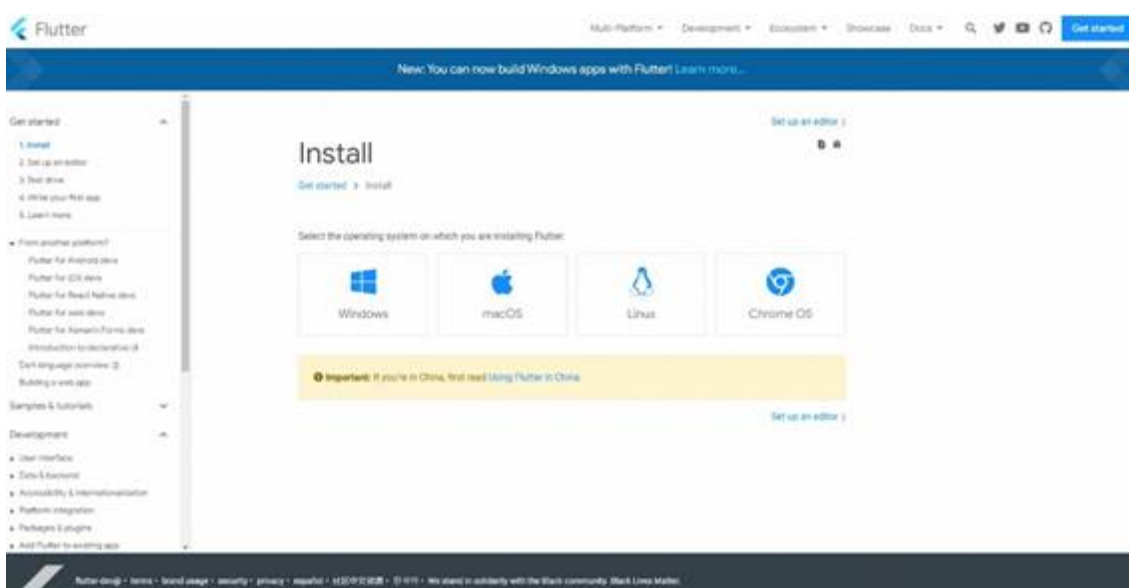
Figura 4 – Página Principal do *Site do Flutter*



Fonte: <https://flutter.dev>.

Nessa página, clique em “Get Started”. Ambos os botões na parte de baixo da tela, quanto na parte superior direita levam para a próxima página. **Passo 2:** escolha o sistema operacional para o qual se deseja desenvolver Flutter.

Figura 5 – Página de escolha de Instalação do Flutter

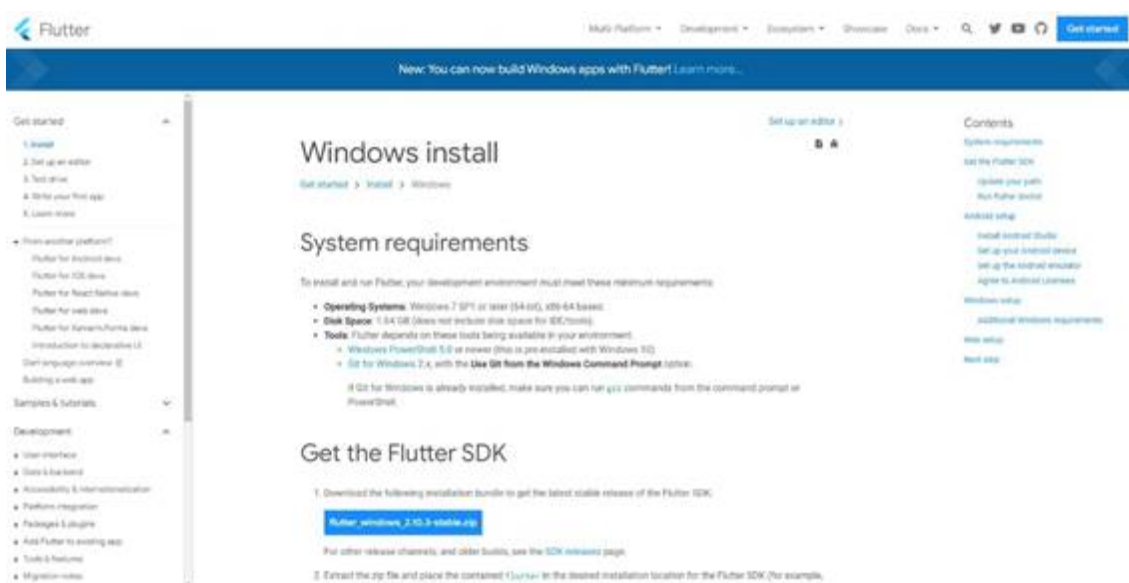


Fonte: <<https://docs.flutter.dev/get-started/install>>.

Para esse passo a passo, instalaremos em uma máquina com o sistema operacional Windows. Portanto, vamos clicar em “Windows”. Quem for desenvolver em Mac deve seguir os passos encontrados no *link*: <<https://docs.flutter.dev/get-started/install/macos>>, os que forem desenvolver em linux devem seguir os passos do *link*: <<https://docs.flutter.dev/get-started/install/linux>> (Acesso em: 17 maio 2022) ,

e para aqueles que desenvolverão pelo ChromeOS, devem seguir os passos do *link*: <https://docs.flutter.dev/get-started/install/chromeos> (Acesso em: 17 maio 2022). **Passo 3:** observar os passos de instalação recomendadas do *site*.

Figura 6 – Página de Instalação do Flutter para Windows



Fonte: <<https://docs.flutter.dev/get-started/install/windows>>.

No lado direito, encontramos a ordem de passos recomendadas do *site* e nos utilizaremos dele para a nossa instalação. É importante notar que caso esse passo a passo tenha sido desatualizado, sempre siga as instruções recomendadas pelo *site*! **Passo 4.1:** verificando requerimentos – PowerShell.

O PowerShell já vem instalado nas versões mais recentes do Windows, do Windows 7 SP1 ou mais recente. Mas para verificar se o PowerShell está instalado, o desenvolvedor pode clicar no ícone do Windows no canto inferior esquerdo da tela de seu computador e digitar "PowerShell", entre as opções de programas disponíveis, deve aparecer "powershell.exe".

Caso o usuário deseje instalar uma versão mais recente do PowerShell (opcional), ou por algum motivo o PowerShell não esteja instalado, é possível instalar a versão mais recente do PowerShell pelo *site* da Figura 7. O *link* do PowerShell também pode ser encontrado debaixo de "System Requirements" da Figura 6.

Figura 7 – Página de instalação do PowerShell



Fonte: [https://docs.microsoft.com/en-](https://docs.microsoft.com/en-us/powershell/scripting/install/installing-powershell-on-windows?)

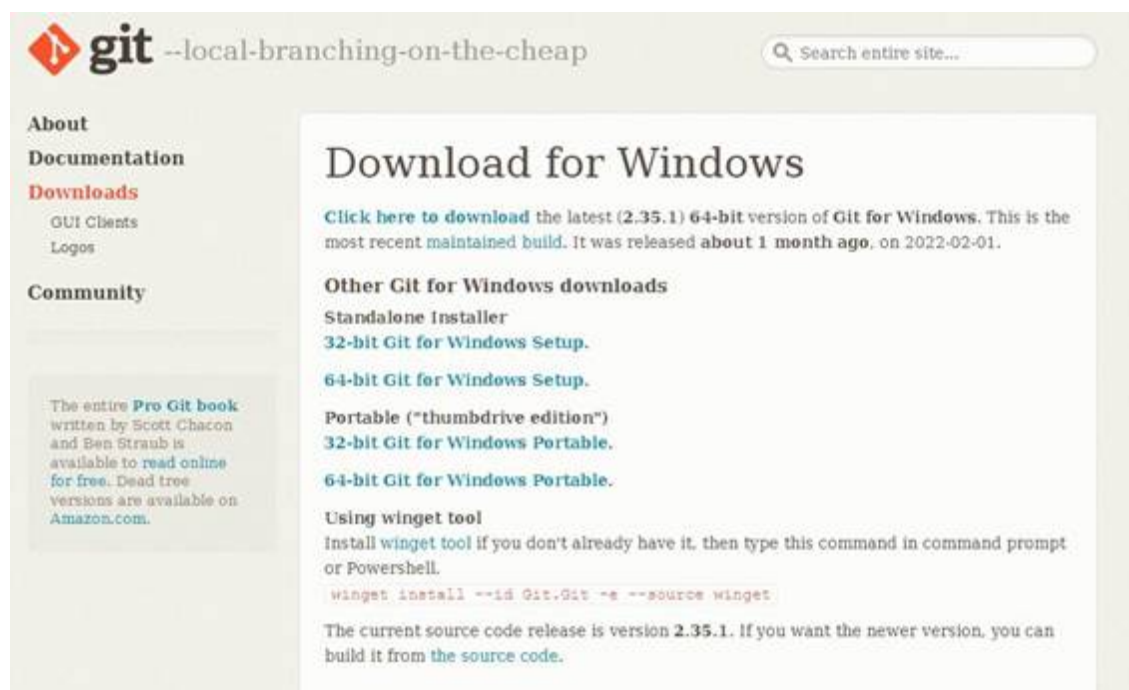
[us/powershell/scripting/install/installing-powershell-on-windows?](https://docs.microsoft.com/en-us/powershell/scripting/install/installing-powershell-on-windows?)

view=powershell-7.2#msi.

Passo 4.2: verificando requerimentos – Git.

Diferentemente do PowerShell, Git não vem normalmente instalado com o Windows, portanto é necessário que ele seja instalado antes de prosseguirmos com o Flutter, caso o leitor já não possua o Git. O *link* do Git também pode ser encontrado debaixo de “System Requirements” da Figura 6.

Figura 8 – Página de instalação do Git

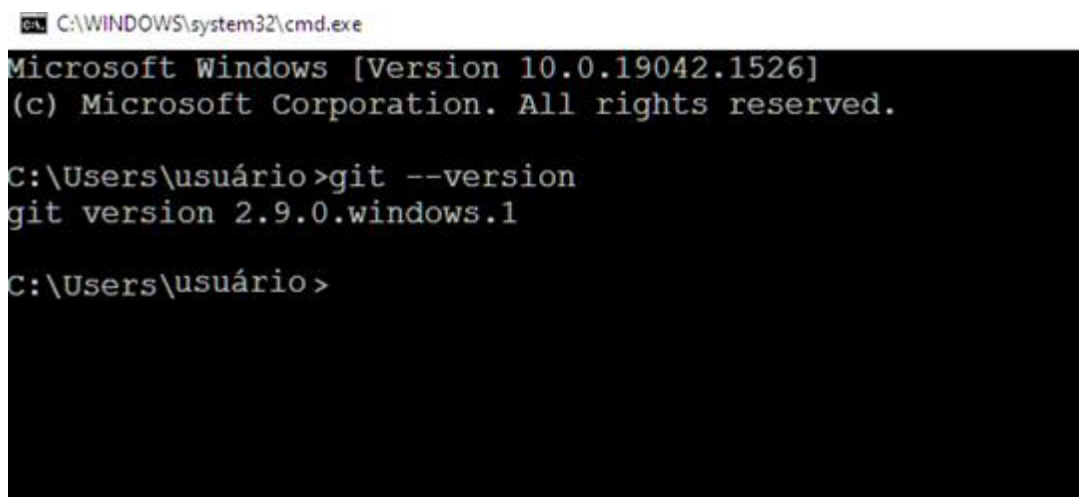


Fonte: <<https://git-scm.com/download/win>>.

Ao clicar em “*Click here to download*”, iniciará o *download* de um executável que instalará o Git. Para verificar se o Git foi instalado com

sucesso ou não, é possível executar o comando "*git-version*" no *prompt* de comando (*prompt* de comando pode ser aberto digitando "cmd" na barra de pesquisa do Windows). Se o Git tiver sido instalado com sucesso, deverá aparecer a versão do Git assim como mostrado pela Figura 9.

Figura 9 – *Prompt* de comando com a versão do Git



```
C:\WINDOWS\system32\cmd.exe
Microsoft Windows [Version 10.0.19042.1526]
(c) Microsoft Corporation. All rights reserved.

C:\Users\usuário>git --version
git version 2.9.0.windows.1

C:\Users\usuário>
```

Fonte: <<https://maisgeek.com/como-verificar-e-atualizar-sua-versao-git/>>.

Passo 5: *Download* do Flutter

Figura 10 – Seção da página de *download* do Flutter

Get the Flutter SDK

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

`flutter_windows_2.10.3-stable.zip`

For other release channels, and older builds, see the [SDK releases](#) page.

2. Extract the zip file and place the contained `flutter` in the desired installation location for the Flutter SDK (for example, `C:\Users\<your-user-name>\Documents`).

Warning: Do not install Flutter in a directory like `C:\Program Files\` that requires elevated privileges.

If you don't want to install a fixed version of the installation bundle, you can skip steps 1 and 2. Instead, get the source code from the [Flutter repo](#) on GitHub, and change branches or tags as needed. For example:

```
C:\src>git clone https://github.com/flutter/flutter.git -b stable
```

You are now ready to run Flutter commands in the Flutter Console.

Fonte: <<https://docs.flutter.dev/get-started/install/windows>>.

Aqui você encontrará a versão estável mais recente do Flutter. Não se esqueça de extrair o arquivo para uma pasta para a qual você deseja deixar instalado o Flutter. O *site* recomenda extrair a pasta em "C:\Usuários\<seu nome de usuário>\Documentos".

AVISO: não instale o Flutter em uma pasta como "C:\Arquivos de Programas", pois essas pastas normalmente requerem privilégios de administrador para ser acessado e pode acabar impedindo o Flutter de rodar corretamente mais tarde.

Como foi mencionado, a comunidade do Flutter é bastante ativa, portanto, não se preocupe caso a versão que você instalar estiver muitas versões à frente, pois a comunidade se esforça para manter as

transições entre as versões a mais suave possível, para que não haja problemas de compatibilidade com versões anteriores.

Caso o leitor deseje muito utilizar a mesma versão utilizada nesse livro, o leitor pode clicar em “SDK releases” como aparece na Figura 10 e, então, uma página como a Figura 11 aparecerá. A versão utilizada nesse livro é a 2.8.1.

Figura 11 – Página de versões do Flutter

Flutter version	Architecture	Ref	Release Date	Dart version
2.10.3	x64	7e9793d	3/2/2022	2.16.1
2.10.2	x64	097d331	2/19/2022	2.16.1
2.10.1	x64	db747aa	2/9/2022	2.16.1
2.10.0	x64	5f105a6	2/3/2022	2.16.0
2.8.1	x64	77d935a	12/16/2021	-
2.8.0	x64	4f44000	12/16/2021	-

Fonte: <<https://docs.flutter.dev/development/tools/sdk/releases>>.

Passo opcional: atualizar as variáveis de caminho de ambiente do Windows. Este possui um recurso chamado de "*Environment Variable Path's*" (ou "variáveis de caminho do ambiente") que ele utiliza para encontrar em que programas executáveis estão localizados.

O próximo passo será a execução do Flutter Doctor para verificar se a instalação não tem nenhum problema, e o Flutter Doctor pode ser executado tanto pelo PowerShell quanto pelo Prompt de Comando do Windows, mas para rodar no *Prompt* de Comando, é necessário colocar o Flutter como uma variável de ambiente. Caso o leitor utilize o PowerShell, esse passo pode ser pulado.

Para adicionar o Flutter à sua lista de variáveis de caminho, siga os seguintes comandos:

1. Clique na barra de pesquisa no canto inferior esquerdo e digite "*Edit Environment Variables*" e selecione a primeira opção.
2. Na parte de cima, estão as variáveis de usuário, selecione a variável "*Path*" e clique em Editar.
3. Clique em Novo, e coloque o caminho da pasta "bin" em que você instalou Flutter. Por exemplo: "C:\Usuários\Nome do Usuário\Documentos\Flutter\bin".

E pronto! Se tudo ocorreu corretamente, você poderá executar o próximo passo no *Prompt* de Comando. **Passo 6:** checar a instalação com o Flutter Doctor.

Abra o PowerShell, ou o *Prompt* de Comando se você realizou o passo opcional anterior e rode o seguinte comando: "*flutter doctor*".

Figura 12 – Exemplo de Erro do Flutter *Doctor*

```
[...] Android toolchain - develop for Android devices
  • Android SDK at D:\Android\sdk
  X Android SDK is missing command line tools; download from https://goo.gl/XxQghQ
  • Try re-installing or updating your Android SDK,
    visit https://docs.flutter.dev/setup/#android-setup for detailed instructions.
```

Fonte: <<https://docs.flutter.dev/get-started/install/windows>>.

O Flutter *Doctor* é uma potente ferramenta de análise. Assim que se executa o comando anterior, o programa começará a checar a instalação e te indicará as soluções para os possíveis erros que possam ocorrer, assim como apresentado pela Figura 12.

Para essa parte, é muito útil um conhecimento básico de inglês para seguir as instruções do Flutter *Doctor*. Caso não possua conhecimento em inglês, as instruções são bastante diretas, portanto, é possível passar o texto por um tradutor.

Figura 13 – Exemplo de Flutter *Doctor* com a instalação correta

```
Doctor summary (to see all details, run flutter doctor -v):
[✓] Flutter (Channel stable, 2.8.1, on Microsoft Windows [Version 10.0.19042.1526], locale pt-BR)
[✓] Android toolchain - develop for Android devices (Android SDK version 32.0.0)
[✓] Chrome - develop for the web
[✓] Android Studio (version 2020.3)
[✓] VS Code, 64-bit edition (version 1.63.2)
[✓] Connected device (2 available)

• No issues found!
```

Fonte: foto da tela do PowerShell <<https://flutter.dev/>>.

Caso a instalação tenha ocorrido sem problemas, ou todos os erros tenham sido corrigidos, deverá aparecer uma tela como a da Figura 13. **Passo 7 (Recomendado):** instalando Android *Studio*.

Figura 14 – *Setup* do Android

Android setup

Note: Flutter relies on a full installation of Android Studio to supply its Android platform dependencies. However, you can write your Flutter apps in a number of editors; a later step discusses that.

Install Android Studio

1. Download and install [Android Studio](#).
2. Start Android Studio, and go through the 'Android Studio Setup Wizard'. This installs the latest Android SDK, Android SDK Command-line Tools, and Android SDK Build-Tools, which are required by Flutter when developing for Android.
3. Run `flutter doctor` to confirm that Flutter has located your installation of Android Studio. If Flutter cannot locate it, run `flutter config --android-studio-dir <directory>` to set the directory that Android Studio is installed to.

Set up your Android device

To prepare to run and test your Flutter app on an Android device, you need an Android device running Android 4.1 (API level 16) or higher.

1. Enable **Developer options** and **USB debugging** on your device. Detailed instructions are available in the [Android documentation](#).
2. Windows-only: Install the [Google USB Driver](#).
3. Using a USB cable, plug your phone into your computer. If prompted on your device, authorize your computer to access your device.
4. In the terminal, run the `flutter devices` command to verify that Flutter recognizes your connected Android device. By default, Flutter uses the version of the Android SDK where your `adb` tool is based. If you want Flutter to use a different installation of the Android SDK, you must set the `ANDROID_ID_SDK_ROOT` environment variable to that installation directory.

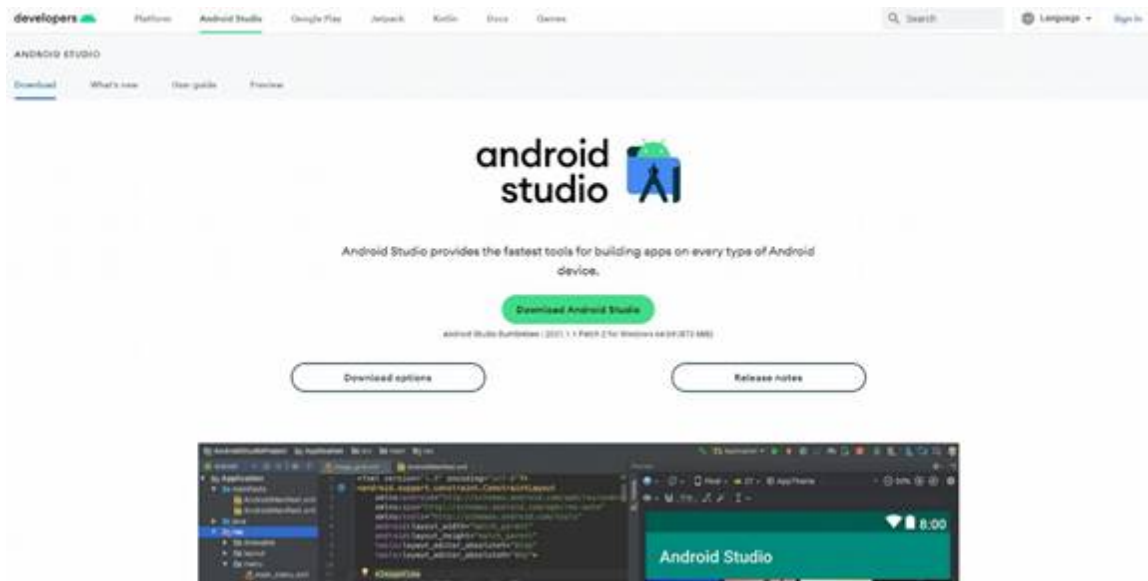
Fonte: <<https://docs.flutter.dev/get-started/install/windows>>.

Esse passo não é necessário, mas é altamente recomendado. Caso o leitor não queira utilizar um emulador, é possível utilizar o seu próprio aparelho Android para rodar os programas feitos nesse estudo.

Para os que forem desenvolver utilizando o próprio dispositivo Android em vez de um emulador, devem instalar o Google USB *Driver* pelo *link*: <https://developer.android.com/studio/run/win-usb> (Acesso em: 17 maio 2022) para poder realizar os *debugs*.

AVISO: os desenvolvedores que vão utilizar o próprio aparelho devem se certificar que o seu aparelho roda uma versão do Android 4.1 ou superior. Deve-se também habilitar as opções de desenvolvedor (*Developer Options*) e o *debug* via USB (*USB debugging*).

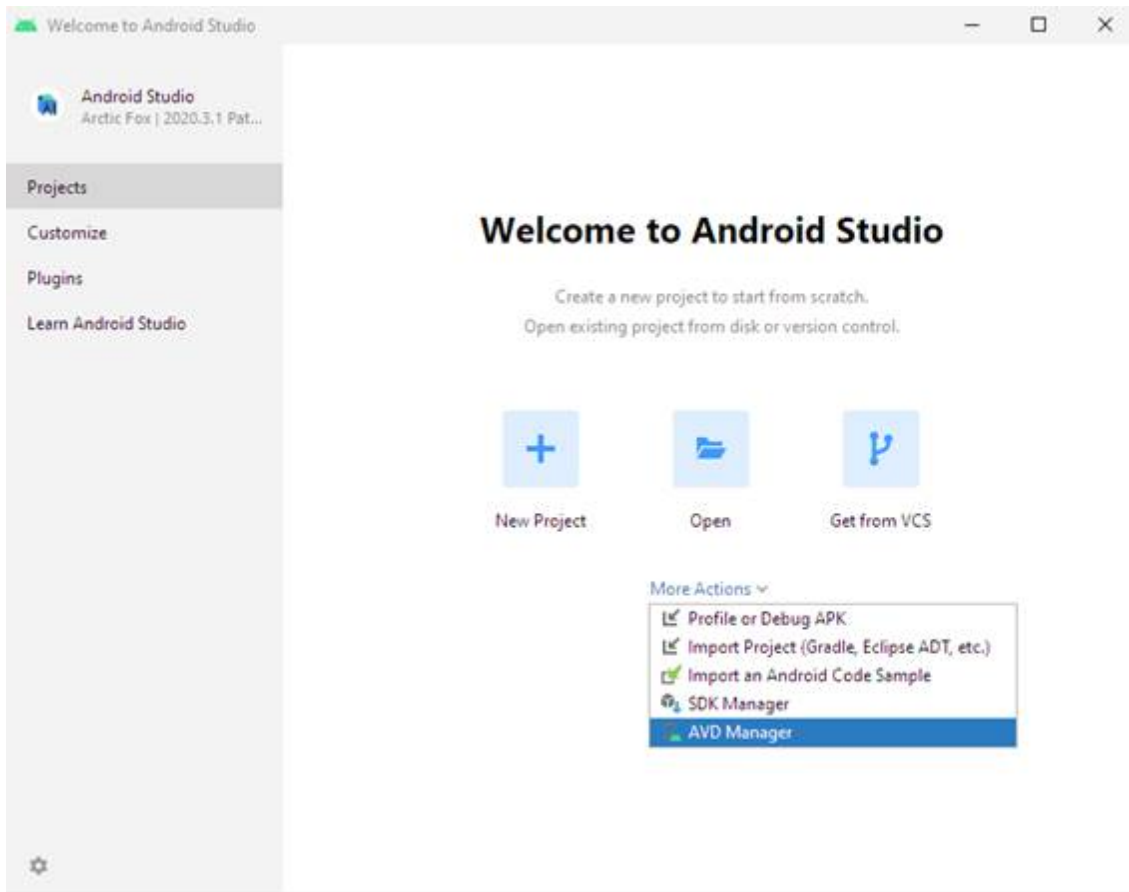
Para os que vão utilizar um emulador (recomendado), falta agora instalarmos o Android *Studio*, um programa muitíssimo útil para se desenvolver qualquer aplicativo para dispositivos android. Seguindo a Figura 14, vamos baixar e instalar o Android *Studio* pelo *link* "<<https://developer.android.com/studio>>" (Acesso em: 17 maio 2022).

Figura 15 – Página Principal do *Android Studio*

Fonte: <<https://developer.android.com/studio>>.

Assim como mostra a Figura 15, clicaremos no botão central da tela para baixar a versão mais recente do *Android Studio*. Assim que tivermos o programa baixado e instalado devemos preparar o programa para o desenvolvimento de aplicativos, primeiramente abriremos o programa e clicaremos em *AVD Manager*, assim como mostra a Figura 16.

Figura 16 – Tela do *Android Studio* com foco no *AVD Manager*



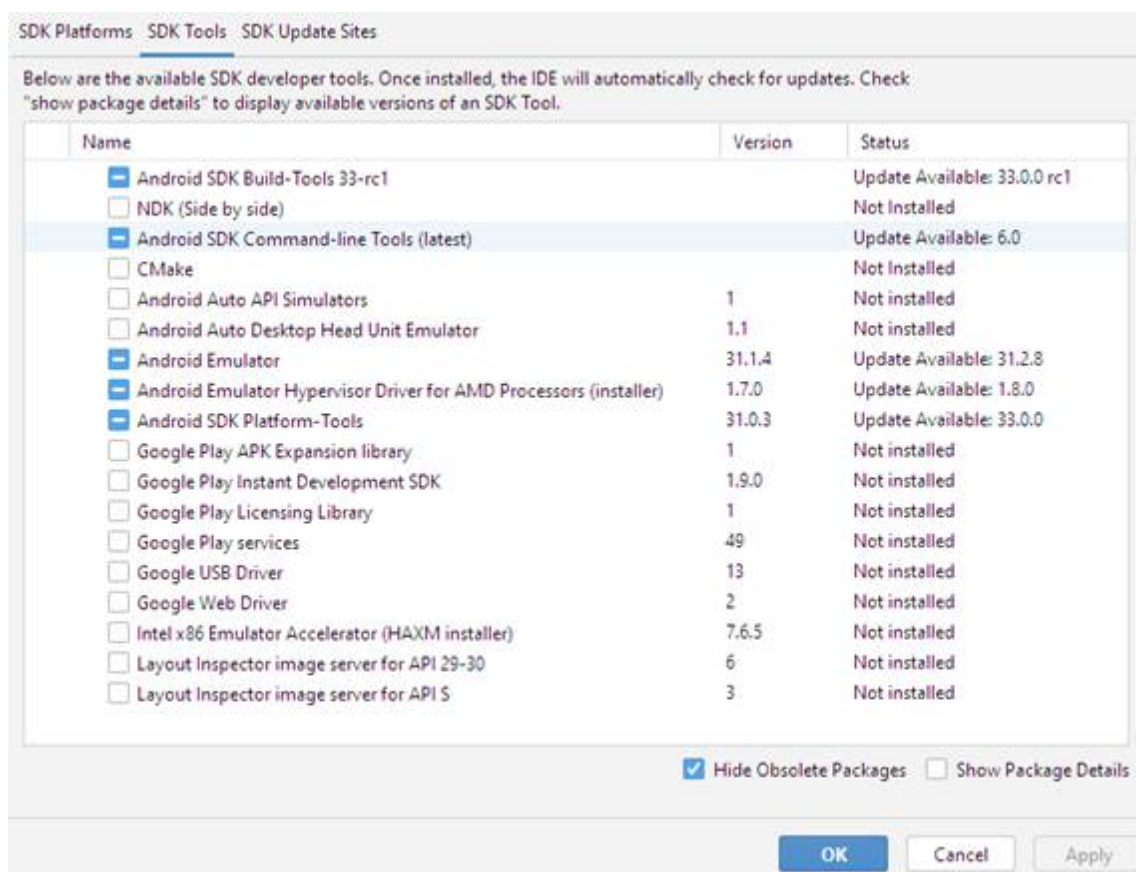
Fonte: Android Studio – <<https://developer.android.com/studio>>.

No *AVD Manager*, criaremos um novo dispositivo clicando em "*Create Virtual Device*". Neste momento, o desenvolvedor poderá escolher qual dispositivo será emulado, para esse tutorial, utilizaremos um Pixel 2 com o sistema operacional Android 8.1 Oreo, que chamaremos de "*Flutter Emulator*". É recomendado que se escolha um dispositivo com um sistema x86 ou x86_64.

Agora vamos abrir o *SDK Manager* para finalizar as nossas configurações. Certifique-se de instalar as seguintes ferramentas, de acordo com a Figura 17:

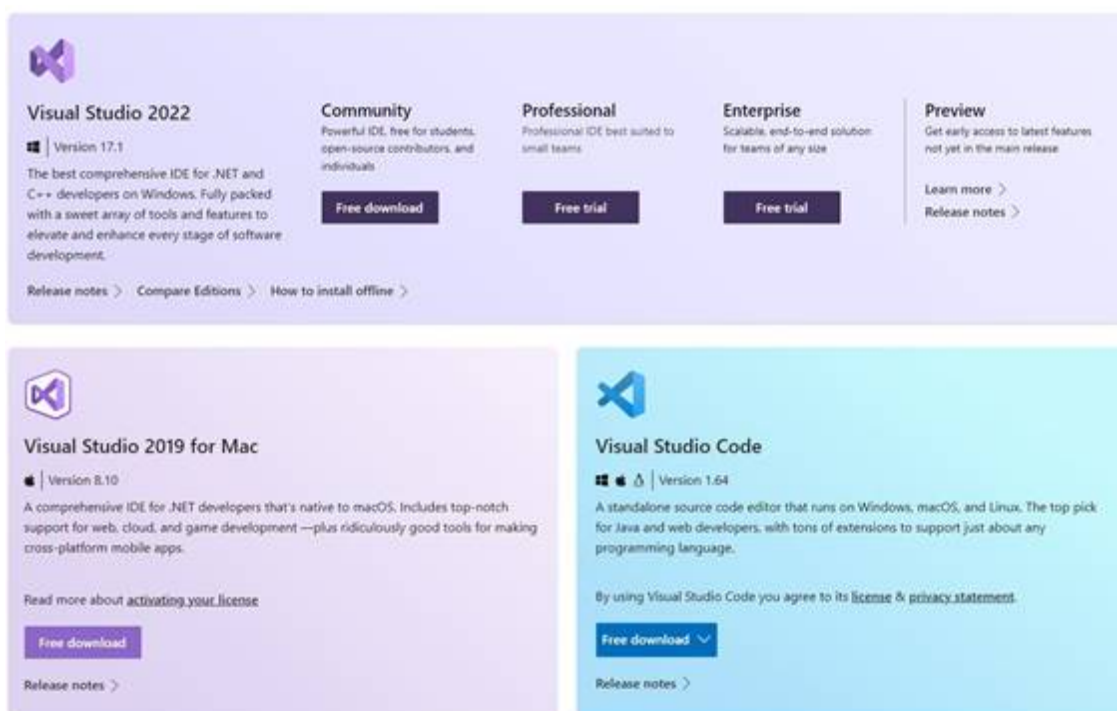
- Android 8.1 Oreo (ou o sistema operacional que foi escolhido, caso não tenha sido escolhido o Android 8.1 anteriormente)
- "Android SDK *Build-Tools*" em SDK Tools
- "Android SDK *Command-line Tools*" em SDK Tools
- "Android *Emulator*" em SDK Tools
- "Android *Emulator Hypervisor Driver*" em SDK Tools
- "Android SDK *Platform-Tools*" em SDK Tools

Figura 17 – Tela de configurações do SDK Tools



Passo 8: instalando o *Visual Studio Code*. Estamos chegando nos finais. O *Visual Studio Code* (VS Code) é o último programa importante para desenvolvermos aplicativos em um sistema operacional Windows. Entre no link [<https://visualstudio.microsoft.com/downloads/>](https://visualstudio.microsoft.com/downloads/) e clique em "Free Download" no quadrado azul, assim como mostra a Figura 18.

Figura 18 – Página de *Downloads* do VS Code

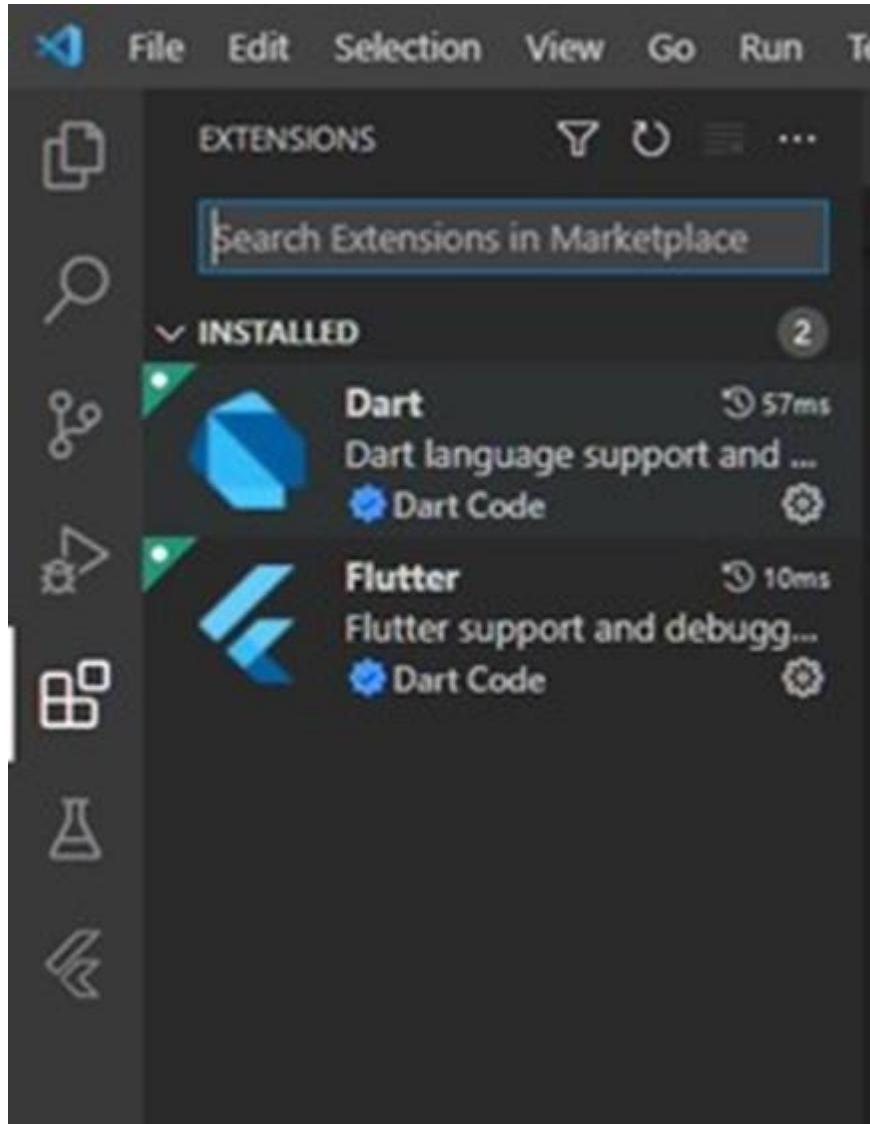


Fonte: [<https://visualstudio.microsoft.com/downloads/>](https://visualstudio.microsoft.com/downloads/).

Assim que tiver o VS Code baixado e instalado, só precisamos agora configurá-lo para estarmos pronto para finalmente começarmos

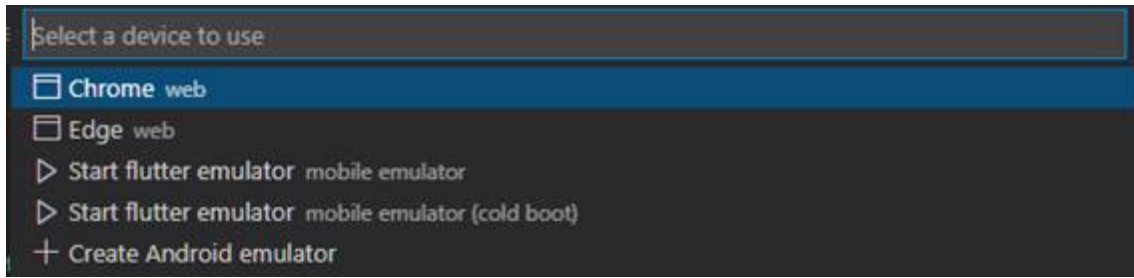
a desenvolver o nosso primeiro aplicativo.

Figura 19 – Extensões necessárias, Tela do VS Code



Assim como mostra a Figura 19, procure e instale essas duas extensões. A seguir, vamos certificar que temos um emulador de Android. Na parte inferior direita, em que aparece "No Device", clique nele e uma janela similar à da Figura 20 aparecerá.

Figura 20 – Opções de Emuladores, foto da Tela do VS Code



Caso não tenha nenhum emulador de Android, clique em "*Create Android Emulator*", e o VS Code deverá criar para você um emulador.

FINALIZANDO

Vimos durante esta etapa as origens e por que utilizar Flutter, assim como um detalhado passo a passo para se instalar o Flutter. Caso o leitor encontre algum problema durante a instalação que não foi coberto por esse texto, a comunidade do Flutter é grande e praticamente todos os erros possíveis possuem uma solução na internet.

REFERÊNCIAS

AMADEO, R. Google starts a push for cross-platform app development with Flutter SDK. **Ars Technica**, 2021.

FLUTTER. **Build apps for any screen.** Disponível em: <flutter.dev>. 2022.

GOOGLE Developers. **Sky:** An Experiment Writing Dart for Mobile (Dart Developer Summit 2015). 30 abr. 2015. Disponível em: <<https://www.youtube.com/watch?v=PnlWI33YMwA>>. Acesso em: 17 maio 2022.

LELEL, W. M. Why Flutter Uses Dart. **HackerNoon**, 2018.