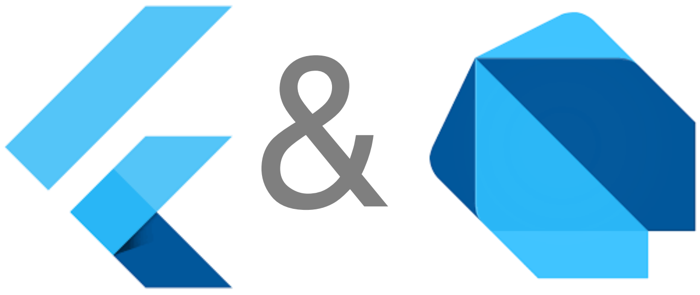
Chapter I

Introduction to Flutter Framework



I. Introduction to Flutter & Dart

I. Brief History of Flutter & Dart

Flutter

Flutter is an open-source UI software development kit created by Google. It is used to develop applications for Android, iOS, Windows, Mac, Linux, Google Fuchsia.[5] and the web.

The first version of Flutter was known as codename "Sky" and ran on the Android operating system. It was unveiled at the 2015 Dart developer summit, with the stated intent of being able to render consistently at 120 frames per second.

Note:

Dec 2018 => 1.0

Flutter's engine, written primarily in C++, provides low-level rendering support using Google's Skia graphics library. Additionally, it interfaces with platform-specific SDKs such as those provided by Android and iOS.[8] The Flutter Engine is a portable runtime for hosting Flutter applications. It implements Flutter's core libraries, including animation and graphics, file and network I/O, accessibility support, plugin architecture, and a Dart runtime and compile toolchain. Most developers will interact with Flutter via the Flutter Framework, which provides a modern, reactive framework, and a rich set of platform, layout and foundation widgets.

Dart

General-purpose programming language developed by google in 2011. It is a client optimized language used to build beautiful mobile and web applications.Flutter apps are written in the Dart language and make use of many of the language's more advanced features.

Flutter runs an debug in the Dart virtual machine, featuring a just-in-time execution engine, allowing for "hot reload", with which modifications to source files can be injected into a running application, which support
for stateful hot reload,

Release versions of Flutter apps are compiled with ahead-of-time (AOT) compilation on both Android and iOS, making Flutter's high performance on mobile devices possible.

Note:

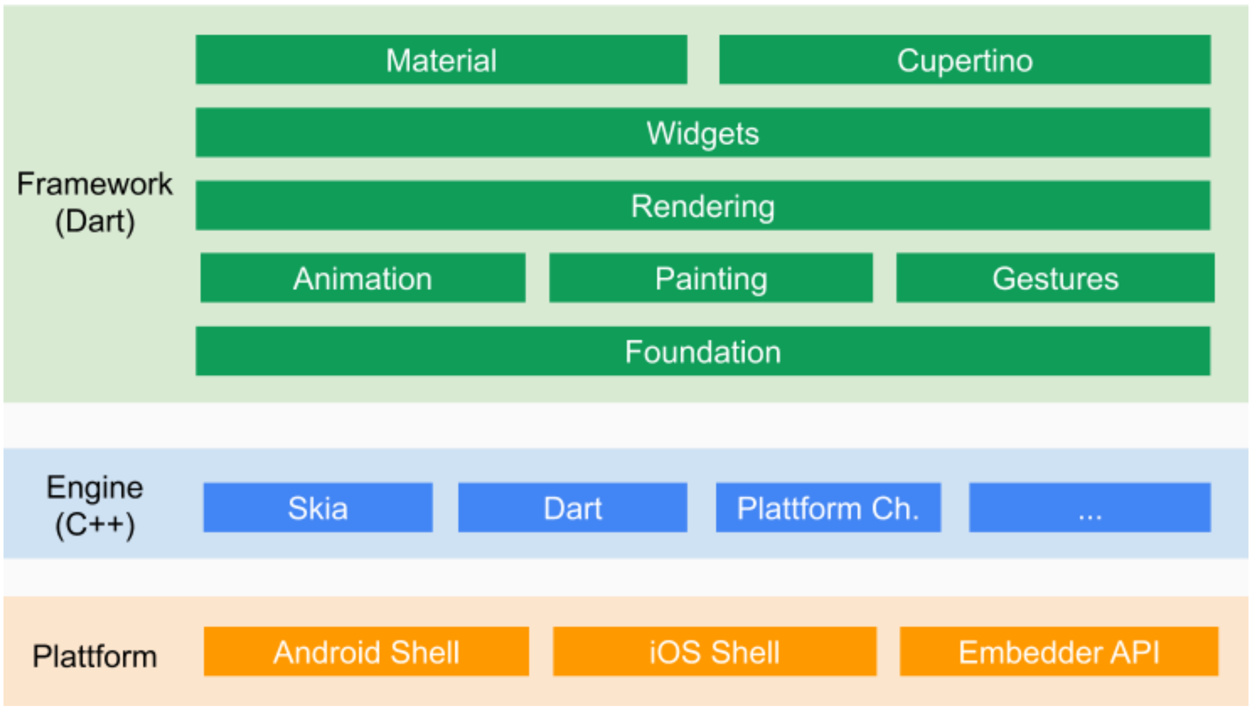
UI design in Flutter involves using composition to assemble / create "Widgets" from other Widgets. smaller Widgets are also made up of even smaller Widgets, and each has a build() method of its own. This is how Flutter makes use of Composition.

Complex widgets can be created by combining many simpler ones, and an app is actually just the largest Widget of them all (often called "MyApp"). The MyApp Widget contains all the other Widgets, which can contain even smaller Widgets, and together they make up your app.

The Flutter framework contains two sets of widgets which conform to specific design languages. **Material Design widgets** implement Google's design language of the same name, and **Cupertino widgets** implement Apple's Human Interface Guidelines iOS design

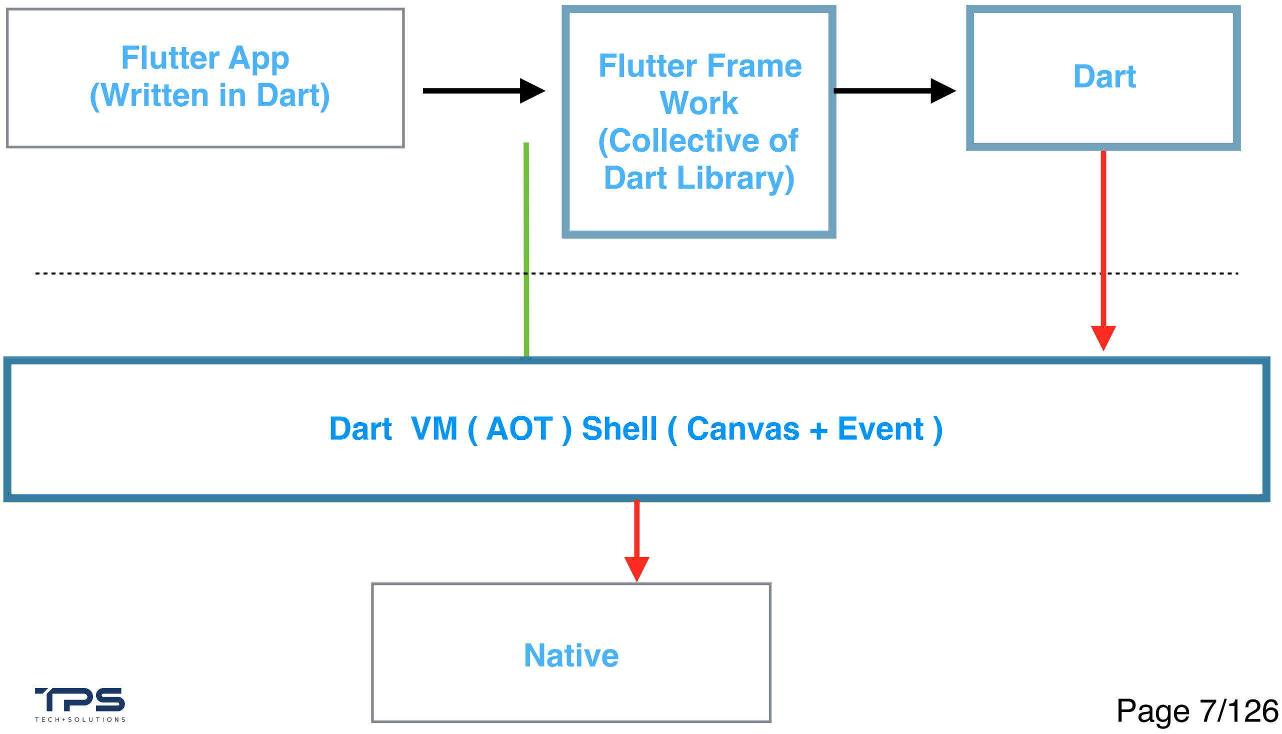
Dart	=> JIT (Development)
	=> AOT (Release)
Flutter	=> Dart 2.0
	a garbage-collected,
	object-oriented language with a sound type system and type inferencing

II. How Flutter Work together with Dart



Alternatively : https://flutter.dev/docs/resources/technical-overview

https://flutter.dev/docs/resources/faq#what-is-inside-the-flutter-sdk


Flutter App (Written in Dart)
Flutter Frame Work (Collective of Dart Library)
Dart
Dart VM (AOT) Shell (Canvas + Event)
Native

III.Anatomy of Flutter App

Everything inside of Flutter app is a widget and you build widgets upon widgets just like Lego blocks on Lego blocks in order to create your app.

Scaffold => a blank screen for an app.

An AppBar	=> At the top, and this is a pre-built widget that simply looks like an AppBar
Container,	=> Just a box and it's going to contain the content of an app.
Column	=> Container that stack things vertically.
Row	=> Container that put things horizontally (side by side)
Icon Widget	=> Use an icon widget to display predefined icon
Image Widget	=> Use to put image , located in assets folder
Network Image	=> User to put image for web

IV.Setup & Installation

Ref: https://flutter.dev/docs/get-started/install

I. Windows Installation Work Flow

1. **Download Flutter SDK**

   eg: **flutter_windows_v1.9.1+hotfix.4-stable.zip**

2. **Unzip and place contained folder *flutter* into c:/src**

3. **Edit Window's environment variable name PATH**

   By typing 'env' in command prompt to load edit Environment Variable window ,then select PATH entry and click 'Edit' button Append the text **;C:/src/flutter/bin.** And save.

4. **Test the Flutter Command**

   Load command prompt window and test the command '**flutter**' .if command show flutter with its available option, flutter installation is success

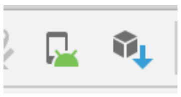
5. **Checking Android Development Option**

   In command prompt, type ' **flutter doctor'** to check any additional requirement

6. **Install Android SDK & Requirements**

   Flutter relies on a full installation of Android Studio to supply its android platform dependencies and then you can write flutter apps using variety of editors in later steps.

   1. Download and Install android studio
      1. https://developer.android.com/studio
      2. Install Android respective SDK using Android Studio



(File->Setting->Android SDK and check required SDK Platform and apply)

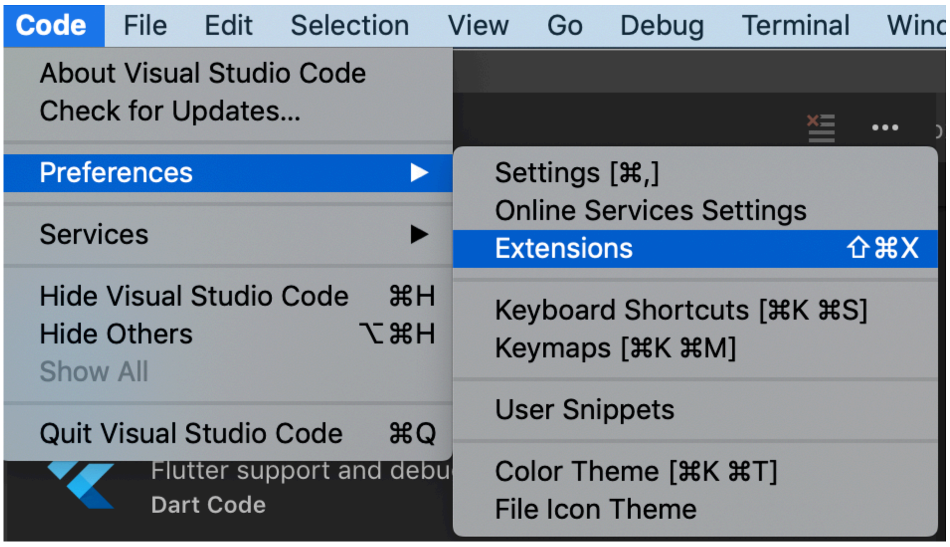
This will install platform tool into default sdk path:

C:\Users\USERNAME\AppData\Local\Android\Sdk

3. Create Android Emulator using existing sdk platform and system image

7. Install Code Editor & Setup (Visual Studio Code)

1. Download Visual Studio Code Editor and install https://code.visualstudio.com/
2. Run VS Code & Intall Flutter & Dart extension



OR View > Command Palette....

Type Install and select extension

Type Fluter in search field

Select Flutter and then click install

(Which wil install Dart Plug In too)

II. Mac Installation Work Flow

Ref: https://flutter.dev/docs/get-started/install

1. **Download Flutter SDK**

   eg: **flutter_windows_v1.9.1+hotfix.4-stable.zip**

2. **Unzip and place contained folder *flutter* into**
   1. USERNAME/src
3. **Edit environment variable name PATH**

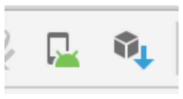
4. **Test the Flutter Command**

   Load terminal prompt window and test the command '**flutter**' .if command show flutter with its available option, flutter installation is success

5. **Checking Android Development Option**

   Flutter relies on a full installation of Android Studio to supply its android platform dependencies and then you can write flutter apps using variety of editors in later steps.

6. **Download and Install android studio**
   1. https://developer.android.com/studio
   2. Install Android respective SDK using Android Studio



(File->Setting->Android SDK and check required SDK Platform and apply)

This will install platform tool into default sdk path:

/Users/YOURUSERNAME/Library/Android/sdk

3. Create Android Emulator using existing sdk platform and system image

7. iOS Setup

Install the latest stable version of Xcode

Run command in terminal prompt:

```
sudo xcode-select --switch /Applications/Xcode.app/Contents/Developer
```

This is the correct path for most cases, when you want to use the latest version of Xcode. If you need to use a different version, specify that path instead. Make sure the Xcode license agreement is signed by either opening Xcode once and confirming or running `sudo xcodebuild -license` from the command line.

8. Install Code Editor & Setup (Visual Studio Code)

1. Download Visual Studio Code Editor and install
   https://code.visualstudio.com/
2. Run VS Code & Intall Flutter & Dart extension

V. Creating Flutter App from Scratch

In VSC Terminal , browse to flutter SDK Path and run following command to create new flutter app

```
> flutter create helloworld
```

this will create a new flutter app and its folder structure

browse VSCode to the newly created app’s folder: helloworld.

I. Simulation

In VSC Terminal , debug to test on simulator

Debug -> Start Debugging

On Android , virtual device is required to create and on iOS , a simulator is load first before running if flutter is not able to detect simulator or emulator or by running command :

> **open -a Simulator**

II. Device

Running on Android Device, no additional setup is required but on iOS , need to setup developer account and install necessary tool ahead. Ref: https://flutter.dev/docs/get-started/install/macos

VI.Scaffolding a flutter App

Modified main.dart to following as Simple Structure and run the app in Android Emulator and iOS Simulator.

VII.Assets Management & the Pubspec File

open pubspec.yaml file with VSC and find the text ‘assets’ and add the following line eg. You must create assets folder under project root folder and add the actual file

```
# assets:
  #  - images/a_dot_burr.jpeg
  #  - images/a_dot_ham.jpeg
```

Or

```
assets:

  - assets/gradiant.png
  - assets/gradiant2.png
  - assets/facebookbtn.png
  - assets/googlebtn.png
```