Introduction to Flutter

MNNIT Computer Coding Club

Objective

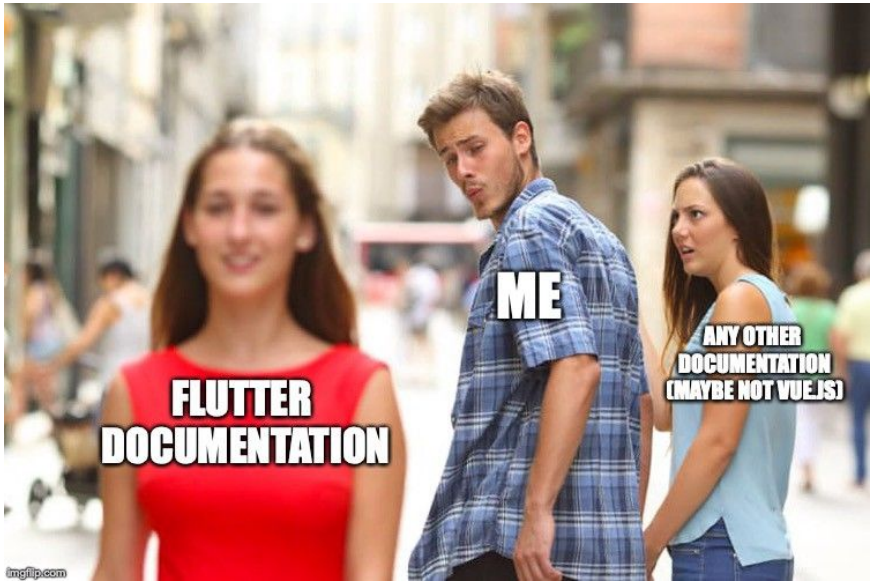
To make you answer the following questions yourselves:

1. Why to learn Flutter?
2. What actually is Flutter?
3. How to develop your first Flutter app?
4. What is a Widget?
5. What is a Container?
6. What is a Layout?

Made by Google

Reference

1. Flutter Docs: https://flutter.dev/docs
2. Lots of online tutorials ;)



Introduction to Flutter

According to Wikipedia, Flutter is an open-source UI software development kit created by Google. It is used to develop cross platform applications for Android, iOS, Linux, Mac, Windows, Google Fuchsia, and the web from a single codebase.

Dart programming language is used to code Flutter.

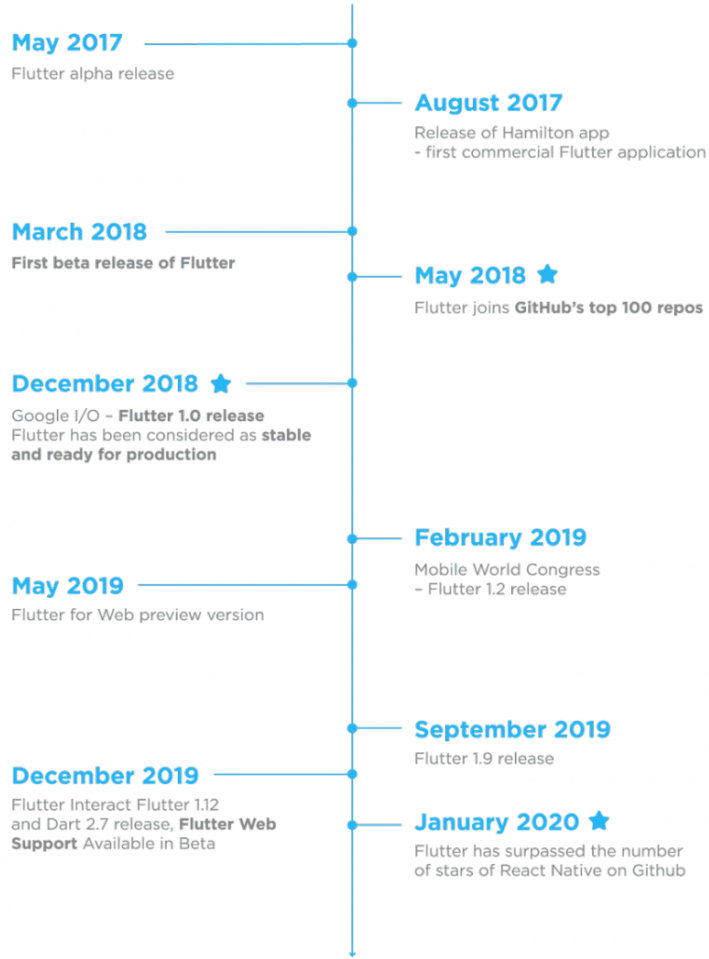
Introduction to Dart

1. Dart is a client-optimized language for fast apps on any platform.
2. Dart is free and open source.
3. Dart is a reactive language that talks similar to python in terms of ease of coding while keeping the power of native java under the hood.

History of Flutter

Flutter launched as a project called Sky which at the beginning worked only on Android. Flutter's goal is enabling developers to compile for every platform using its own graphic layer rendered by the Skia engine.

First described in 2015, Flutter was released in May 2017.


May 2017
Flutter alpha release
August 2017
Release of Hamilton app
- first commercial Flutter application
March 2018
First beta release of Flutter
May 2018
Flutter joins GitHub's top 100 repos
December 2018
Google I/O – Flutter 1.0 release
Flutter has been considered as stable and ready for production
February 2019
Mobile World Congress
– Flutter 1.2 release
May 2019
Flutter for Web preview version
September 2019
Flutter 1.9 release
December 2019
Flutter Interact Flutter 1.12
and Dart 2.7 release, Flutter Web Support Available in Beta
January 2020
Flutter has surpassed the number of stars of React Native on Github

History of Flutter

On March 3, 2021, Flutter Engage Edition: 2.0 was released.

Currently as of 25th October Flutter 2.5.3 is the latest version.

Installation

Find the installation instructions here:
https://flutter.dev/docs/get-started/install

Why Flutter?

1. One codebase for iOS, Android, Web and Desktop.
2. Flutter is the only mobile SDK that provides reactive views without requiring a JavaScript bridge.
3. Flutter apps look and feel great.
4. Make a change in the app and see them in the blink of an eye. All thanks to Hot-Reload.

Widgets

Everything in Flutter is a Widget.

1. Widget is a description of a part of UI. Each element on a screen of the Flutter app is a widget.
2. In flutter, Widget is a way to declare and construct UI.
3. Flutter widgets are built using a modern framework that takes inspiration from React.
4. If you are familiar with the Android development then you might make the immediate connection with the views.
5. But dear like view, Widget is not just a piece of UI. Widget is a lot more than just structural elements like buttons.

Central Idea of Widget

1. The central idea is that you build your UI out of widgets.
2. Widgets describe what their view should look like given their current configuration and state.
3. When a widget's state changes, the widget rebuilds its description, which the framework ***diffs*** against the previous description in order to determine the minimal changes needed in the underlying render tree to transition from one state to the next.

Hello World Flutter App

1. The minimal Flutter app simply calls the runApp() function with a widget.
2. The runApp() function takes the given Widget and makes it the root of the widget tree.
3. In this example, the widget tree consists of two widgets, the Center widget and its child, the Text widget.

Categories of Widget

1. There are mainly 14 categories in which the flutter widgets are divided. These include Input, Scrolling, Assets etc.
2. You can visit https://flutter.dev/docs/development/ui/widgets to have a look on them.
3. We will discuss few basic widgets here.

Text

The Text widget lets you create a run of styled text within your application.

Row and Column

These flex widgets let you create flexible layouts in both the horizontal (Row) and vertical (Column) directions. The design of these objects is based on the web's flexbox layout model.

Container

1. The Container widget lets you create a rectangular visual element.
2. A container can be decorated with a BoxDecoration, such as a background, a border, or a shadow.
3. A Container can also have margins, padding, and constraints applied to its size.

Material Components

1. Be sure to have a uses-material-design: true entry in the flutter section of your pubspec.yaml file.
2. It allows you to use the predefined set of Material icons.
3. Many Material Design widgets need to be inside of a MaterialApp to display properly, in order to inherit theme data. Therefore, run the application with a MaterialApp.
4. For using Material Widgets, refer: https://flutter.dev/docs/development/ui/widgets/material

Handling Gestures

1. Most applications include some form of user interaction with the system.
2. The first step in building an interactive application is to detect input gestures.
3. See how that works by creating a simple button.

Gesture Detector

1. The GestureDetector widget doesn't have a visual representation but instead detects gestures made by the user.
2. When the user taps the Container, the GestureDetector calls its onTap() callback.
3. You can use GestureDetector to detect a variety of input gestures, including taps, drags, and scales.
4. Many widgets use a GestureDetector to provide optional callbacks for other widgets. For example, the IconButton have onPressed() callback that is triggered when the user taps the widget.
5. For details refer: https://flutter.dev/docs/development/ui/advanced/gestures

Changing widgets in response to user input

1. We want to build a simple app in which we have a button with a text (initially this is 0). Everytime, we press the button, we will increment the text by 1.
2. So far, we have used only stateless widgets.
3. In order to build more complex experiences—for example, to react in more interesting ways to user input—applications typically carry some state.
4. Flutter uses StatefulWidgets to capture this idea.


I SEE WIDGETS
EVERYWHERE


Don't Be afraid
to Start Over.
It's A Chance to Build
Something Better.

Thank You.