


Getting Started with flutter

*Created By:-
The EasyLearn Academy*

Call us for training on +91 9662512857



Flutter Framework...

- Flutter framework was first introduced to world by Google in 2015.
- Its code name was "Sky" and it ran on the Android operating system.
- The first stable release was delivered on the 4 of December 2018.
- The framework is written using C, C++, Dart languages.
- Flutter uses Google's Skia Graphics Engine for user interface rendering.
- This graphics engine is in Google Chrome, Chrome OS, Chromium OS, Mozilla Firefox, Mozilla Thunderbird, Android, Firefox OS.

Flutter Framework...

- Flutter runs using Dart virtual machine (VM) on Windows, Linux, and macOS operating systems.
- The Dart VM uses a just-in-time (JIT) code compilation which provides a development-time-saving feature such as hot-reloading.
- While the developer writes and debugs the mobile application, the JIT compilation injects new code into the running application. It provides a stateful hot-reload feature, where, in most cases source code changes can be immediately reflected in the running application without requiring a restart or any loss of state. This saves the developers' a lot of time in the end.
- In case of installing the application, the Dart VM uses the ahead-of-time (AOT) compilation which further converts Dart code into a native platform-dependent machine code making Flutter's high performance on mobile devices possible.
- Flutter is based upon principle that **Everything's a widget.**

Flutter Framework...

- **Remember that everything is a widget and it is the basic building block of the application.**
- A widget can be a unique button, style element, or a separate pop-up screen, etc.
- Flutter does not have separate controllers, views, or layouts.
- Flutter use composition approach.
- The composition approach is better than inheritance.
- Often widgets are composed of other smaller widgets and that is the composition based approach.
- Flutter API allows you to combine multiple widgets (Widget tree) to create screen as per your need.



Flutter Framework...

- A widget tree is group of widget that represent user interface components.
- These widgets could be stateless or stateful.
- A useful feature that helps in managing the applications' states.
- There are two sets of widgets that one can use to develop app , Cupertino (iOS) and Material (Android).
- These sets of widgets are responsible for the user interface (UI) and include every component that you may need for Android and iOS development.
- These widgets are not connected with the native API of iOS or Android but work as standalone Flutter components with appropriate rendering speed and animation. So user will not be affected by poor user experience, because of it.

What is stateful and stateless widget?

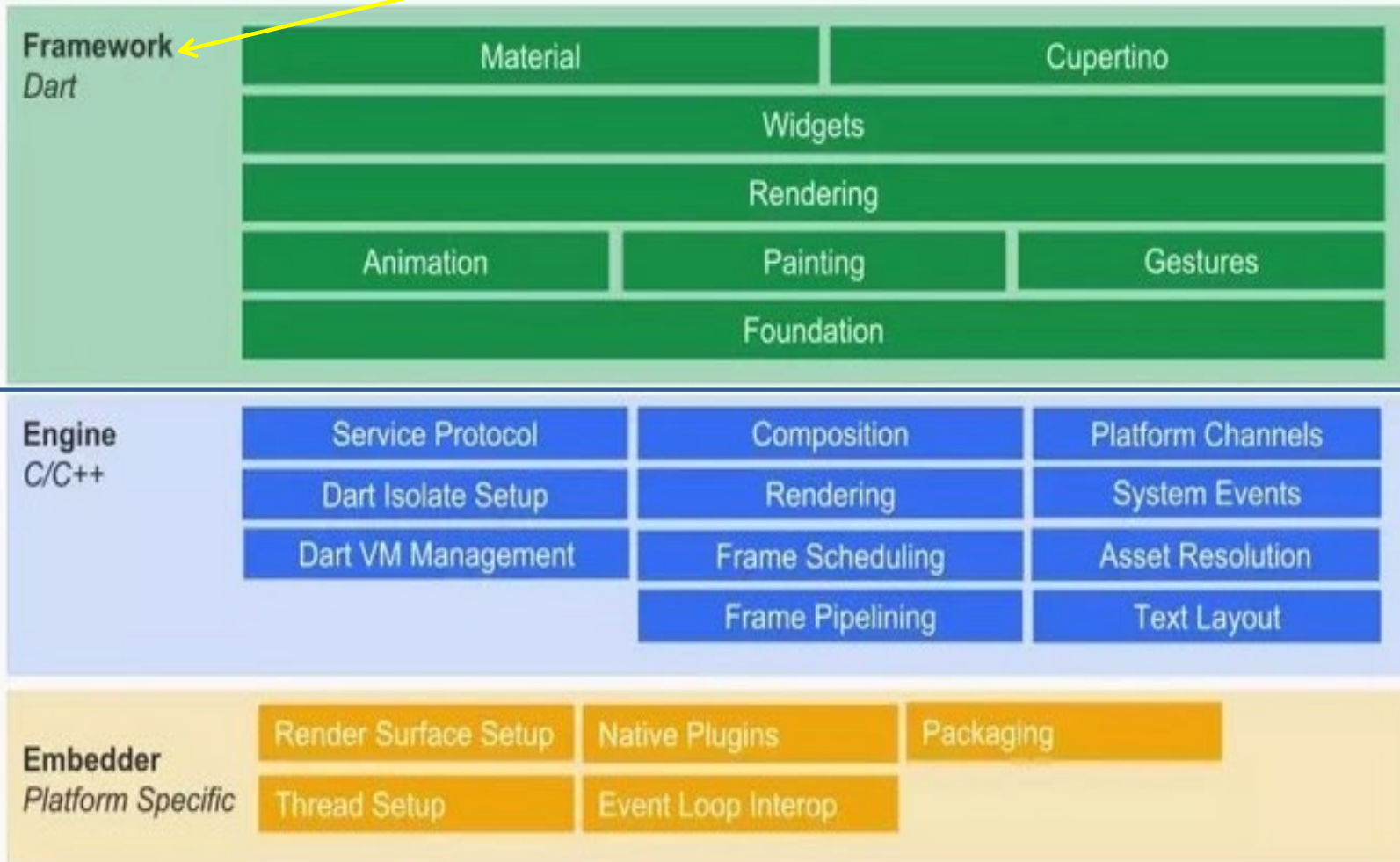
- Stateful Widgets are dynamic widgets.
- They can be updated during runtime based on user action or data change.
- Stateful Widgets have an internal state and can re-render if the input data changes or if Widget's state changes.
- For Example: Checkbox, Radio Button, Slider are Stateful Widgets
- Stateless Widget are the widgets whose state can not be altered once they are.
- For example, we use Text or the Icon in our flutter application where the state of the widget does not change in the runtime.
- It is used when the UI depends on the information within the object itself.

Flutter Architecture

- Flutter is a popular open-source UI kit by Google, which allows building cross-platform apps.
- Flutter allows code reuse across both Android and iOS operating systems. Therefore developers are able to produce high-quality apps that feel comfortable on different platforms.
- During development, Flutter apps run in a VM that offers stateful hot reload of changes without needing a full recompile.
- For release, Flutter apps are compiled directly to machine code, whether Intel x64 or ARM instructions, or to JavaScript
- if targeting the web. The framework is open source, with a permissive BSD license
- It allows use of third-party packages that supplement the core library functionality

Flutter System Overview

Let us understand this



Let's understand it.

- Flutter is arranged as a layered, extensible system. It functions as a sequence of independent libraries, with each of them depending on the layer below it.
- Developers work with Flutter through the Flutter framework written in the Dart language. It has the layout, a rich set of platforms, as well as basic libraries divided into multiple layers.
- **Basic building block services** and **foundational classes**, such as animation, gestures, and painting, offer commonly used to access foundation classes.
- **The rendering layer** is used to with layout and building a tree of renderable objects.
- **The widgets layer** is based on reactive programming model. Each class in the widgets layer work with a render object in the rendering layer. in addition, the widgets layer allows defining combinations of reusable classes.

Widget composition

- Widgets are composed of many single-purpose widget blocs that produce powerful effects when combined.
- Flutter uses the same basic concept (a Widget) in the widgets layer in order to represent animations, drawing to the screen, layout, navigation, state management, themeing, and user interactivity.



Widget hierarchy

- Widgets form a composition-based hierarchy.
- Each widget is located inside its parent and receives context from there.
- This hierarchy is present all the way up to the root widget. In addition, the widget hierarchy allows to create design with any feasible combinations.
- When responding to various events, apps update their user interface by commanding the framework to replace a widget in the hierarchy. The framework efficiently updates the user interface after comparing the new and old widgets.

Widget classes and categories

- The two main category of widget are stateful and stateless.
- Based on their features, widgets in Flutter can also be grouped into multiple categories:
- **Layout widgets**
- Flutter provides a large number of widgets with a layout feature to allow composing multiple widgets into a single widget.
- The most popular layout widgets include Center, Column, Container, Row, and Stack.
- **Platform independent / basic widgets**
- Flutter provides has many basic widgets to create both complex and simple platform-independent. Examples are text, image, and icon.
- **Platform-specific widgets**
- They include Android-specific widgets and iOS-specific widgets.
- Android-specific widgets are called *Material widgets*. They are configured based on *Material Design guidelines* by Android OS.
- iOS-specific widgets are called *Cupertino* widgets. They are designed in accordance with *Human Interface Guidelines* by Apple.

