Flutter Interview Question

**What are the Advantages of Flutter**

1. Hot Reload
2. Fast Development
3. Screen Reader
4. Cross Platform
5. Open and Free source
6. Themes for Android and IOS

**Limitation of Flutter**

1. Third Party Libraries are limited
2. Lack of Overall Support

**Different Build on Flutter**

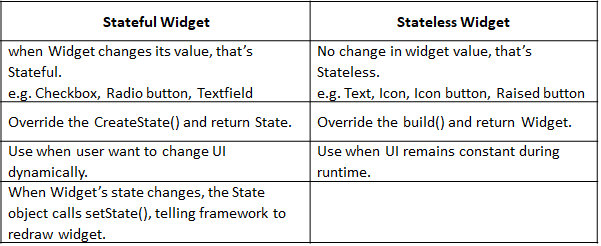
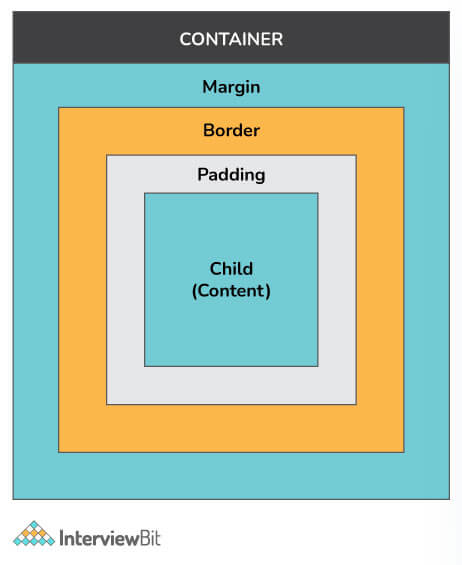
1. Debug Mode :- This mode enables debugging of apps on a physical device, emulator, or simulator. Assertions and service extensions are enabled here. Quick deployment is then achieved by optimizing compilation
2. Profile Mode:- In this mode, some debugging abilities are maintained, enough to analyze the app's performance while testing. Tracing and some extensions are enabled in this case. On emulators and simulators, profile mode is disabled since their behavior does not reproduce real-world performance. The following command can be used to compile the profile mode **flutter run --profile**
3. Release Mode:- When deploying the app, this mode is used to minimize the size of the footprint and maximize optimization. Debugging, assertions and service extensions are disabled here. Faster startup, faster execution, and less size are its key features. The following command can be used to compile the release mode: **flutter run --release**

**What is Widget in flutter**

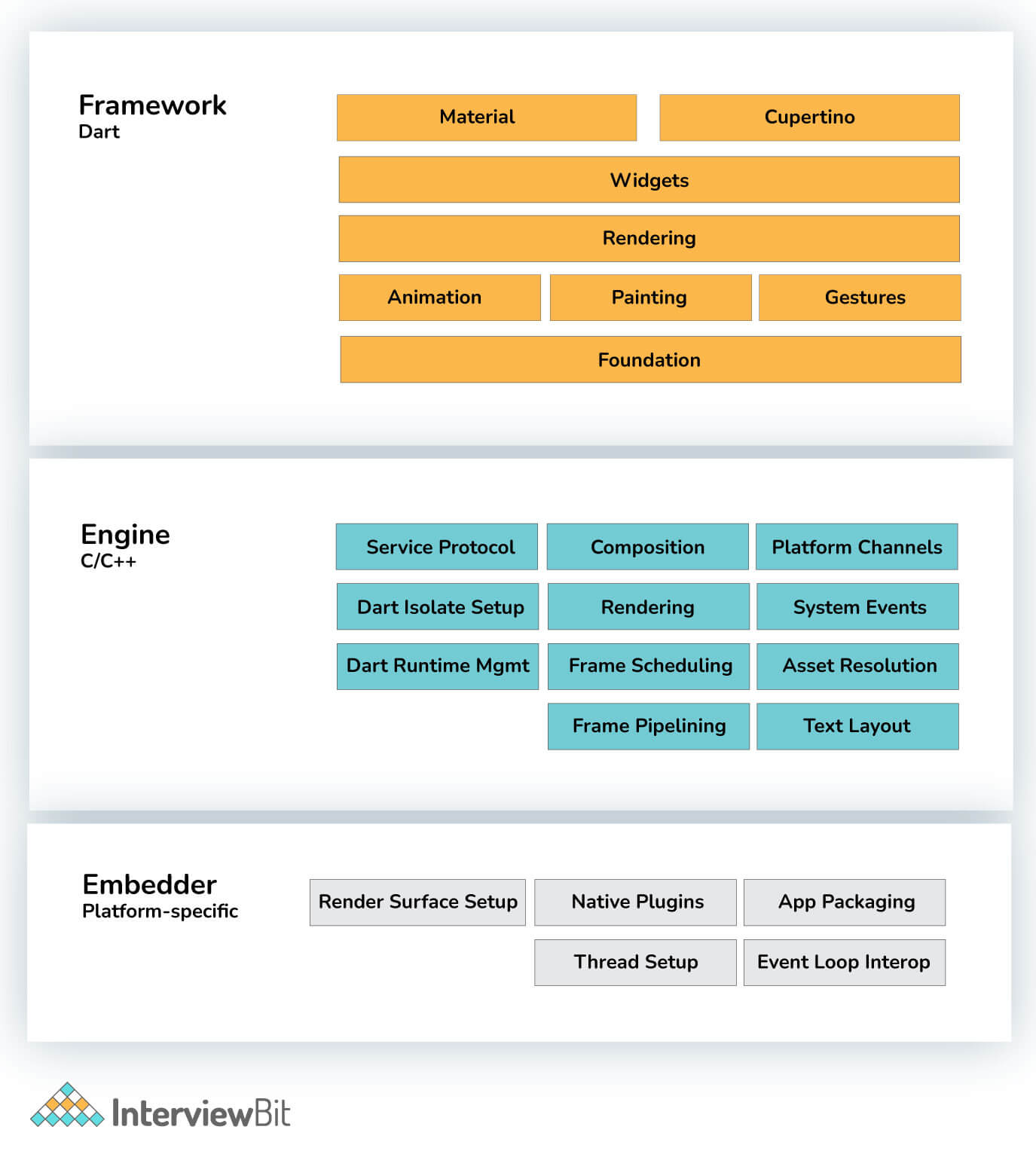
Widgets are the central class hierarchy in the Flutter framework. A widget is an immutable description of part of a user interface. Widgets can be inflated into elements, which manage the underlying render tree. Widgets themselves have no mutable state (all their fields must be final)

**Types of Widgets in Flutter  
  
Stateless Widget**: A widget that does nothing is a Stateless Widget. In essence, they are static and don’t store any state. Thus, they don't save values that may change.

**Stateful Widget**: A widget that does anything is a Stateful Widget. Stateful widgets are dynamic by nature, which means they can monitor changes and update the UI accordingly.

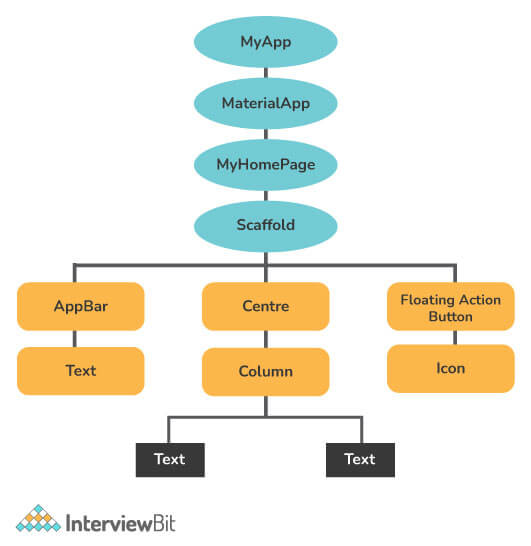
  
  
**Explain Container class in Flutter**

Container class in flutter is a convenience widget that combines common painting, positioning, and sizing of widgets. A Container class can be used to store one or more widgets and position them on the screen according to our convenience. Basically, a container is like a box to store contents. A basic container element that stores a widget has a margin, which separates the present container from other contents. The total container can be given a border of different shapes, for example, rounded rectangles, etc. A container surrounds its child with padding and then applies additional constraints to the padded extent (incorporating the width and height as constraints, if either is non-null).

**Flutter Architecture**  
  
 **Upper layers:** The Dart-based platform that takes care of app widgets, gestures, animations, illustrations, and materials;

**Flutter engine:** Handles the display and formatting of text.

**Built-in service:** Used for the management of plugins, packages, and event loops

**Flutter widget**

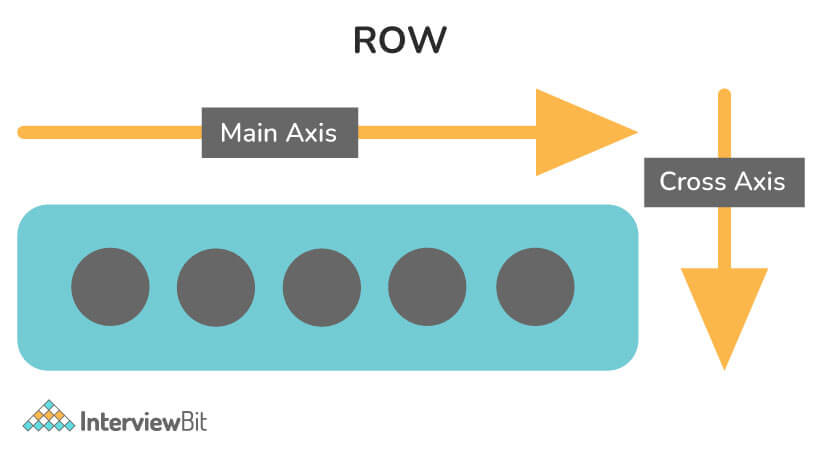
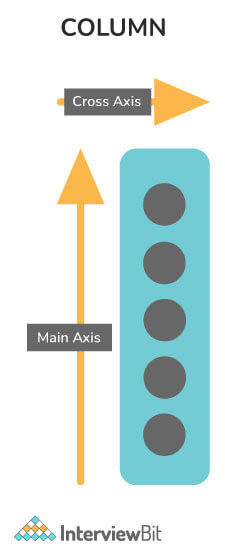
**Write the difference between runApp() and main() in flutter.**

main(): This function starts the program. Flutter does not allow us to write any program without the main() function.

runApp(): Using runApp(), you are able to return the widgets that are connected to the screen as a root of the widget tree that will be rendered on the screen. This function is called in the main function, which is the driver of the app  
  
**When to use mainAxisAlignment and crossAxisAlignment.**

The mainAxisAlignment is how items are aligned on that axis, whereas crossAxisAlignment is how items are aligned on the other axis. Row and column widgets can align their children according to our preferences using the crossAxisAlignment and the mainAxisAlignment properties.

As Children of the Row Widget are arranged horizontally.   
**For Row:**   
mainAxisAlignment = Horizontal Axis   
crossAxisAlignment = Vertical Axis

  
**For Column:**mainAxisAlignment = Vertical Axis   
crossAxisAlignment = Horizontal Axis  
  


**What is the use of Ticker in Flutter?**We use a ticker to tell how often our animation is refreshed in Flutter. Signals are sent at a constant frequency, such as 60 times per second, using this type of signal-sending class. We understand it better with our watch, which ticks constantly. For each tick, a callback method is provided that has the time since the first tick at each second since it was started. The tickers are synchronized immediately, even if they begin at different times  
  
**What is the use of Mixins?**

Multiple inheritances are not supported by Dart. Thus, we need mixins to implement multiple inheritances in Flutter/Dart. The use of mixins makes it easy to write reusable class code in multiple class hierarchy levels. Mixins can also be used to provide some utility functions (such as RenderSliverHelpers in Flutter)  
  
**What do you mean by Streams?**

In asynchronous programming, streams are used to provide a sequence of data in an asynchronous manner. Similar to a pipe, we put a value on one end and a listener receives it on the other. Several listeners can be put into one stream, and they'll all get the same value when they're put in the pipeline. It's possible to create and manage streams through the SteamController  
  
**Write difference between Hot reload and Hot restart?**

**Hot Reload**: It is considered an excellent feature of flutter that takes approximately one second to perform its functionality. With this function, you can make changes, fix bugs, create UIs, and add features easily and quickly. By utilizing the hot reload feature, we can quickly compile the new code in a file and send it to Dart Virtual Machine (DVM). As soon as DVM completes the update, it updates the app's UI. The preserved state is not destroyed in hot reload.

**Hot Restart**: It has a slightly different functionality as compared to a hot reload. In this, the preserved states of our app are destroyed, and the code gets compiled again from the beginning. Although it takes longer than a hot reload, it's faster than a full restart function.

**Explain pubspec.yaml file.**

The pubspec.yaml file, also known as 'pubspec', is a file that is included when you create a Flutter project and is located at the top of the project tree. This file contains information about the dependencies like packages and their versions, fonts, etc., that a project requires. It makes sure that the next time you build the project, you will get the same package version. Additionally, you can set constraints for the app. During working with the Flutter project, this configuration file of the project will be required a lot. This specification is written in YAML, which can be read by humans.

The following are included in this file:

1. General project settings, like name of the project, version, description, etc.
2. Dependencies within a project.
3. The assets of the project (e.g., images, audio, etc.).

**What is await in Flutter? Write it's usage**.

Until the async method is finished, await interrupts the process flow. Await generally means: Wait here until this function is finished so that you can get its return value. Await can only be used with async. Using this, all currently running functions are put on hold until PF nature is complete.

**Explain BuildContext.**

BuildContexts are used to identify or locate widgets in widget trees. Each widget has its own BuildContext, i.e., one BuildContext per widget. Basically, we're using it to find references to other widgets and themes. In addition, you can utilize it to interact with widget parents and access widget data