Basic Flutter interview questions and answers

1.

Why is Flutter preferred over other mobile app developing tools?

Hide Answer

Flutter is a free and open-source mobile UI framework, introduced by Google in May 2017. Flutter enables you to develop a native mobile app using only one codebase. This implies you may design two separate apps using the same programming language and codebase (for iOS and Android).

Flutter is preferred over other mobile app development tools like Java and React Native due to a myriad of reasons. A few of them are:

- Flutter supports cross-platform development
- Programming in Flutter is extremely easy and flexible
- The building process in Flutter is much faster than that in all its competitors

2.

What are packages and plugins in Flutter?

Hide Answer

A package is a set of classes, interfaces, and sub-packages that allow users to create modular code that can be shared easily. Instead of starting from scratch, using packages can help construct applications fast. In Flutter, you may use a package to add new widgets or functionality to an app.

On the other hand, a plugin is a piece of software that enhances your app's functionality. Plugins play a crucial role in the Flutter ecosystem.

3.

What are the limitations of Flutter?

Hide Answer

Since Flutter is still a relatively new tool, it has some limitations. They are as follows:

- The number of third-party libraries is very small
- The release size of Flutter is larger than expected
- Flutter requires to be used along with another OOP language, Dart which cannot compete with other OOP languages such as Java, C#
- Due to Flutter’s limited complexity, mobile ad platforms do not support it
- Flutter is still not used by a wide audience

4.

Why does it generally take a long time to develop a Flutter app?

Hide Answer

Because Flutter generates a device-specific IPA or APK file, building a Flutter application takes substantially longer the first time. This method, which normally takes a long time, uses Xcode and Gradle to build a file.

5.

What are keys and how do you use them?

Hide Answer

A set of IDs for Widgets, Elements and SemanticsNodes make up the key class. Keys are responsible for preserving the state of widgets when they are updated inside the widget tree. Keys may also be used to modify and rearrange collections of widgets of the same type and defined states.

Otherwise, keys may be superfluous to the code, even if they aren't harmful in any manner. Keys are useful when you need to edit a widget tree with stateful widgets, but not when the tree is entirely made up of stateless widgets.

6.

What are the different types of streams in Dart?

Hide Answer

Streams provide an asynchronous sequence of data. Asynchronous programming uses the idea of streams. They refer to a program's asynchronous succession of data occurrences. We put a value on one end and a listener on the other, similar to a pipe. Several listeners can be combined into a single stream, and when they are placed in the pipeline, they will all get the same data. It is possible to utilize the SteamController to establish new streams or manage existing ones. There are two types of streams:

- **Single Subscription Streams**
  These streams convey events in chronological order. They're thought of as individual sequences inside a greater totality. When the sequence in which events are received matters, such as when reading a file, these streams are employed. Throughout the sequence, there can only be one listener, and the event will not be triggered if there isn't one.
- **Broadcast Streams**
  These broadcasts provide subscribers with information about upcoming events. Subscribers can instantly begin listening to events after subscribing to them. These are flexible streams that allow several listeners to listen at the same time. Furthermore, even after canceling a previous membership, one may listen again.

7.

What is pubspec.yaml file?

Hide Answer

While creating a Flutter project, a particular type of file is always included at the top of the project. This file is known as the pubspec.yaml file, also called ‘pubspec’. This file contains information about a project's dependencies, such as packages and their versions, typefaces, and so on. It ensures that the package version is the same the next time you create the project. You can also place restrictions on the app. This project's configuration file will be used a lot while working with the Flutter project. This specification is written in YAML, a human-readable markup language.

8.

What is Flutter?

Hide Answer

Flutter is an open source software development kit (SDK) UI used for cross-platform applications from a single codebase. Flutter is primarily used for creating natively compiled, multi-platform applications across iOS and Android seamlessly.

9.

Flutter is developed by which company?

Hide Answer

Google

10.

What was the first version of Flutter which ran on an Android operating system?

Hide Answer

Sky is the first version of Flutter which ran on Android OS. It was announced at the 2015 Dart developer summit with the stated goal of being able to render consistently at 120 frames per second.

11.

Can you tell us the four main elements of Flutter?

Hide Answer

The four main elements of Flutter are:

- Flutter engine
- Widgets
- Design-specific widgets
- Foundation Library

12.

When was the first version of Flutter released?

Hide Answer

The first version of Flutter was released in May 2017, but it was first announced in 2015. In December 2018, Google released Flutter 1.0. In December 2019, Flutter 1.12 was released.

13.

Flutter is written in which language?

Hide Answer

Flutter was written in the Dart language. While writing and debugging an application, Flutter runs in the Dart virtual machine, which features a just-in-time execution engine.

14.

What does the acronym SDK stand for?

Hide Answer

Software Development Kit is a set of software tools and programs provided by software and hardware vendors that developers can use to develop applications for specific platforms. An SDK helps developers easily integrate their apps with a vendor's services. SDKs can include APIs(Application Programming Interfaces), sample code, documentation, and other resources that help developers create software applications.

15.

Can you name some best editors for Flutter development?

Hide Answer

Some popular IDEs (Integrated Development Environment) for Flutter include the following:

- Android Studio
- IntelliJ Idea
- Emac
- Visual Studio
- Codemagic

16.

Which operator is used to evaluate and return values between two expressions?

Hide Answer

This “??” operator is used to evaluate and return values between two expressions. This operator first checks the expression 1 and, if it is non-null, returns its value; otherwise, it will evaluate and return the value of expression 2.

17.

Can you tell us how many kinds of widgets there are in Flutter?

Hide Answer

There are two main types of widgets in Flutter. These include:

StatelessWidget- It does not have any state information. It is static throughout its lifecycle. Examples are Row, Text, Column, and Container.

StatefulWidget- It has state information. It contains two classes: the state object and the Widget. It is dynamic because it can change the inner data during the Widget's lifetime. Examples are Radio, Form, Checkbox, and TextField.

18.

Can you name the command used to compile the release mode?

Hide Answer

This command “Flutter run -- release” is used to compile the release mode. Release mode for a web app means that the app is compiled with the dart2js compiler for best performance.

19.

Name the different types of build modes in Flutter.

Hide Answer

There are three types of build modes in Flutter. These include:

Debug- It is used to test the apps. On Android Studio, you can find a green play button on the top panel.
“Flutter run”

Profile- In this mode, some debugging ability is maintained - enough to profile your app's performance, and also, it has the performance as the release mode.
“Flutter run -- profile”

Release- It is used for deploying the app on marketplaces.
“Flutter run - - release”

20.

Which widget in Flutter is a box that comes with a different size?

Hide Answer

SizedBox is a widget in Flutter that allows us to specify an arbitrary size for a widget. It is basically used to add empty space between widgets. SizedBox also supports specifying a specific aspect ratio for its child.

21.

Suppose you have to represent real world behavior in Flutter, which animation would you use?

Hide Answer

The choice of animation in Flutter would depend on the specific behavior that needs to be represented. Flutter provides several animation options that can be used to represent real-world behavior. These include Physics-based animation, Tween animation, Curved animation, and Hero animation.

22.

Name some popular apps that use Flutter.

Hide Answer

Some of the most popular apps that employ Flutter include:

- Google Ads
- Hamilton
- KlasterMe
- Reflectly

23.

Which folder is used to write Android apps in Flutter?

Hide Answer

Android Folder is used to write Android apps in Flutter. The Android Folder contains files and folders required for running the application, and these files are autogenerated during the creation of a Flutter project.

24.

What is the use of the Await function?

Hide Answer

The Await function is an asynchronous function. Its job is to wait until it gets the final value. Await is used with async functions and that run asynchronously, giving users the choice to wait for the asynchronous mode to finish before continuing.

25.

Can you tell us which function compiles and updates the app?

Hide Answer

In Flutter, there are several functions and commands that can be used to compile and update the app. However, the most commonly used function for this purpose is hot reload.

hot reload is a development feature in Flutter that allows you to update your app's code in real-time, without restarting the entire app.

26.

Can you tell which function is accountable for starting the program?

Hide Answer

main () function is used to start a program. This function is highly critical as, without it, one cannot write any function.

27.

What is the purpose of animation controller class in Flutter?

Hide Answer

The Animation Controller class in Flutter is used to control and manage animations. It provides a way to start, stop, or pause an animation, and allows you to define the duration, speed, and direction of an animation.

The primary purpose of the Animation Controller class is to define an animation that can be used in your app's user interface. The class provides methods for defining and manipulating animations, such as setting the duration and the curve used to control the animation's progress.

28.

How can you test a single Widget?

Hide Answer

Using the Widget tests technique, you can make sure that various portions of the user interface work as intended without the need for a physical device or simulator. This technique is the best way to isolate small parts of your app and find out whether your code is behaving as expected

29.

Can you use WidgetsApp for basic navigation?

Hide Answer

Yes, a material app widget builds a navigator, which manages a stack of widgets identified by strings, also known as routes objects and gives you two ways for managing the stack.

30.

Which widget allows us to refresh the screen?

Hide Answer

RefreshIndicator Widget enables us to refresh the screen. When the user pulls down on the widget, the onRefresh callback is triggered, which typically involves fetching new data from a server or updating the UI in some way.

31.

Can you state a few examples of stateless widget?

Hide Answer

Some examples of a stateless widget include the following:

Text - This displays a string of text with a single style.

Container: This widget can contain other widgets and provide padding, margins, and other layout properties.

Icon - This is used for the list of available material icons that can be used with this class.

32.

Explain the term “Tree shaking” in Flutter.

Hide Answer

Tree shaking is a method of removing the unused module in the bundle during the development process. Tree shaking serves as a sort of optimization technique that optimizes the code by removing the dead code.

While importing or exporting codes, there might be dead codes hanging around. Removing these dead codes reduces the code size which in turn improves the performance of the application.

33.

What is used to import packages for your project?

Hide Answer

pubspec.yaml file is used to import packages in the Dart code. It allows us to set the constraints for the application. This file contains project dependencies, general project settings, and project assets.

34.

What’s the use of Navigation.push in Flutter?

Hide Answer

The role of Navigation.push in Flutter is to add a route to a bundle of other routes which are managed by the navigator.

35.

Why HTTP package is used in Flutter?

Hide Answer

HTTP package is used in Flutter for making HTTP requests to web servers. It allows Flutter developers to send HTTP requests and receive HTTP responses from APIs or web servers. Flutter apps need to communicate with APIs or web servers to fetch data or send data to the server. The HTTP package provides a convenient and easy-to-use way to make HTTP requests in Flutter.

36.

Explain profile mode in Flutter?

Hide Answer

The profile mode in Flutter is used to test the functionality of an app while launching it. It compiles and launches your app almost identically to release mode but with additional functionality to allow debugging performance problems.

37.

What's the role of BuildContext in Flutter?

Hide Answer

In Flutter, the BuildContext is an object that provides access to the location of a widget in the widget tree hierarchy and to various services such as Theme, MediaQuery, and Navigator. The BuildContext is used by widgets to access the properties of their parent widget, such as its size, position, and theme. It is is also used to navigate between screens using the Navigator widget.

38.

Can you tell us which class is responsible for implementing the basic material design visual layout structure for an app in Flutter?

Hide Answer

The scaffold class is a widget in Flutter that is used to implement the basic material design visual layout structure. The scaffold class makes it faster to create a general-purpose mobile application. Moreover, it contains almost everything we need to create a functional and responsive Flutter application

39.

Can you tell us what will be the default return type, if you don’t specify the return type for a function?

Hide Answer

In Flutter, Dart is the programming language used to develop the framework. In Dart, if you don't specify the return type for a function, the default return type will be dynamic. This means, if you declare a function in Flutter without specifying a return type, Dart assumes that the function can return any type of value and assigns the dynamic type to the return value by default.

40.

What’s the name of a constructor for a class in Flutter?

Hide Answer

The name of a constructor for a class is the same as that of the class itself. Constructor is a special strategy that is used to create objects with predetermined values for the properties. There are three types of constructors in Flutter: named constructor, standard constructor, and factory constructor.

41.

What’s the core of the Flutter layout mechanism?

Hide Answer

The main core of the Flutter layout mechanism is the Widgets. The widgets are the building blocks of the user interface, and they are arranged in a tree-like structure known as the widget tree. The image, text, icon, and even the layout of your application are all widgets.

42.

Which widget can be used to display one after another in scroll format?

Hide Answer

The ListView Widget is used to display a large number of items as a scrollable list. ListView is an advanced version of the Column widget that automatically provides scrolling when the list of items won't fit on the screen.

43.

What is a layout in Flutter?

Hide Answer

A layout in Flutter refers to how widgets are arranged on the screen. The layout is determined by the constraints passed down from the parent widget, and it determines the position and size of the child widget.

44.

Can you tell us how we can test a single widget in Flutter?

Hide Answer

Widget tests allow building and interacting with widgets in a test environment. It is used to check whether the Widget works as expected or not. Hence, using this testing UI component, you can test a single widget.

45.

What's the latest version of Flutter?

Hide Answer

3.7 is the latest version of Flutter. Along with an improved framework, this new version also has great new features like enhanced material 3 support, cascading menus and menu bars,impeller preview, custom context menus, and DevTools updates among others .

46.

Is Flutter a front-end framework or back-end?

Hide Answer

Flutter is primarily a front-end framework. It provides tools and widgets to build beautiful and engaging user interfaces, and it supports various platforms such as iOS, Android, and the web. However, the framework does provide some back-end functionalities. Flutter applications can communicate with web APIs or connect to backend services through third-party packages or plugins.

47.

What’s the function of the future in Dart?

Hide Answer

A future function is used to predict a potential error or value that may occur in the future. It can be completed with a value or an error. Here, developers can plug call backs for each case. Some programming languages use “promise” for the same function.

48.

What’s the role of image.network() constructor in Flutter?

Hide Answer

The Image.network() constructor is a widget in Flutter that allows you to display images from the network in your application. It is used to load and display images from a URL.
Example:
Image.network('https://piccol.photos/259?image=8’,')

49.

Can you tell us what you know about Rune In Dart?

Hide Answer

In dart programming language strings are a sequence of UTF-16 which is a 16 bit unicode Transformation Format. Rune refers to an integer that is used to describe any Unicode code point.

50.

Is Flutter free?

Hide Answer

Yes. It is a free and open source mobile UI framework created by Google. Here, with only one codebase, you can create native mobile applications.

51.

Why do we use const keyword in Flutter?

Hide Answer

We use the keyword to declare constants. Basically, when we know the value of a variable at compile time. This means the compiler knows in advance what value to store .

52.

What are some operators commonly used in Dart?

Hide Answer

Here are some of the operators that are commonly used to build apps in Dart:

= (assignment operator): Used to assign a value to a variable.
?? (null-aware operator): Used to check whether a variable is null or not and provide a default value if it is null.
! (bang operator): Used to assert that an expression is not null.
. (dot operator): Used to access properties and methods of an object.

53.

Why do we use a ticker in Flutter?

Hide Answer

Tickers are used in Flutter as they provide a secure way of updating the UI at 60 frames per second. It is a class that listens to the frameCallback and calls a tick function that forwards the elapsed duration between the current frame and last frame to the ticker listener.

54.

How can you reduce execution time for running code?

Hide Answer

It will depend on the size and adaptability of the app packages. According to which we can integrate just-in-time compilers to run code which will enhance the app's performance. Dart code execution is faster than many other languages where Ahead-of-time compilers also shorten execution times. And, for merging sequential streams, integrating these compilers helps in run-time reduction and functionality increase in a variety of applications.

55.

What is Dart?

Hide Answer

Dart is a general-purpose, object-oriented programming language that was initially developed by Google in 2011. It is designed to be fast, efficient, and scalable, making it ideal for building a wide range of applications, including web and mobile apps, server-side applications, and command-line tools.

56.

What process will you use to reduce widget rebuild?

Hide Answer

On rebuilding, the state of the widget changes. This, however, helps the user to see the UI reflect state changes. At this point recreating sections of the user interface that do not need to be changed is unnecessary.

To avoid the needless rebuilding of the widgets one can divide the widget tree into small individual widgets each having its own build process. Here const constructor can be used to inform Flutter that the dont need to be rebuilt.

57.

Define spacer widget?

Hide Answer

In Flutter, the Spacer widget is a flexible widget that takes up available space within a Row, Column, or Flex layout. It is commonly used to create vertical or horizontal space between other widgets, or to distribute available space between multiple widgets.

Intermediate Flutter interview questions and answers

1.

Tell me about Flutter Provider.

Hide Answer

Widgets are used to build the provider. Inside the provider, one can use all the objects as if they are a part of it with the new widget subclasses it creates. Primarily, it works on the concept of PUB-SUB which means one provider with many subscribers.

2.

What are the advantages of using Flutter?

Hide Answer

Flutter has many advantages:

- **Fast development**: With the help of Flutter, you can save a lot of time while developing software. Hot reload is a flutter feature with which you can make changes to your code and see the results in real time.
- **Flexible user interface**: Flutter provides your application with a sleek, beautiful, and modern look that is sure to impress your users.
- **Provides native performance**: Flutter applications run faster and smoother on devices.
- **Widgets**: When building your application in Flutter, you get access to a wide range of pre-loaded widgets that can be utilized in your app.

- **Huge community support**: If you want to reach a global audience, then Flutter is for you because it supports internationalization. With it, you can create an application that is available in multiple languages.

3.

What is the purpose of the Navigator in Flutter and how is it used?

Hide Answer

Navigator is a widget in Flutter that manages a stack of screens or "routes" in an application. It is used to navigate between different screens and to manage the state of the navigation stack. Navigator is typically used in conjunction with MaterialApp or CupertinoApp to create a navigation hierarchy for an application.

4.

Can you tell us some examples of App State?

Hide Answer

Some examples of App State include:

- Login info
- Notifications in a social networking app
- User preferences

5.

Explain Streams in Flutter?

Hide Answer

Future and Stream are classes in Dart that define asynchronous programming. It behaves asynchronously as it doesn't return the next event when requested. Instead, it informs when the next event will be returned when ready.

6.

What is the lifecycle of a StatefulWidget?

Hide Answer

A StatefulWidget's lifecycle is as follows:

- createState
- initState
- didChangeDependencies
- build

- didUpdateWidget
- setState
- deactivate
- dispose

7.

Define the term Cookbook.

Hide Answer

A cookbook provides specific programming solutions for some common problems experienced by developers during the development process. Each recipe can be used independently during the development process.

8.

What do you mean by Container class?

Hide Answer

The container class is the convenience widget that enables positioning, sizing, and painting of widgets. A container class can include multiple widgets and it enables developers to manage those widgets according to their convenience.

9.

What is Flutter Architecture?

Hide Answer

Flutter architecture is comprised of three layers:

Flutter Framework Layer: This is the highest layer in the Flutter architecture and includes the widgets and the Material Design widgets. It also includes APIs for animations, gestures, and routing.

Engine Layer: This is the core layer of the Flutter architecture that contains the rendering engine, which handles graphics and text rendering. It also includes the Skia graphics library, used to draw graphics on the screen.

Platform Layer: This is the lowest layer in the Flutter architecture and provides access to the native platform APIs. It includes plugins and packages that allow Flutter to interact with the device's hardware and software, such as the camera, location services, and sensors.

10.

Can you give some reasons why Flutter is better than its alternatives?

Hide Answer

Flutter has several advantages over its alternatives. Here are some of the main reasons why Flutter is a great choice for building mobile apps:

- Flutter offers a hot-reload feature that enables developers to see changes in their code instantly without having to wait for the app to recompile.
- Flutter allows you to build apps that work seamlessly on multiple platforms, including Android, iOS, and the web.
- Flutter offers a rich set of customizable widgets, allowing developers to create beautiful and responsive designs that adapt to different screen sizes and orientations.
- Flutter uses a compiled programming language, Dart, which is optimized for performance and runs natively on the device.

11.

Explain how you will deploy a Flutter app to the Google Play/App Store?

Hide Answer

For deploying Flutter apps to the Google Play Store or App Store, you will need to know the guidelines suggested by each. In both cases, it's important to make sure your app meets the requirements and guidelines of the respective app store before submitting it for review. This includes things like app content, user data privacy, user interface, and more.

Deploying to Google Play Store:

- Create a Google Play Developer account
- Create a signing key
- Configure your app
- Submit your app for review
- Publish your app

Deploying to App Store:

- Create an Apple Developer account
- Create a signing certificate and provisioning profile
- Configure your app
- Submit your app for review
- Publish your app

12.

Explain the Primary Axis and Cross Axis Alignment.

Hide Answer

An alignment defines how a row or a column is aligned with respect to the horizontal or vertical axis. For example, for a row its main axis is horizontal and so its cross axis will be vertical.

- **PrimaryAxisAlignment**: the rows run horizontally, and the columns run vertically.
- **CrossAxisAlignment**: the rows run vertically, and the columns run horizontally.

13.

Can you tell us a way to access screen size in the future?

Hide Answer

You can access the screen size using the MediaQuery class. The MediaQuery class provides information about the current device's screen size and orientation.

14.

What’s the difference between container and sizedBoxe?

Hide Answer

Container and SizedBox are both widgets in Flutter that can be used to size and position child widgets, but they have some differences.

The Container widget is a multi-purpose widget that can be used to apply decoration, padding, margin, and constraints to a child widget. The Container widget can also be used as a layout widget to position its child widget within its bounds.

The SizedBox widget, on the other hand, is a simple widget that has a fixed width and height.The SizedBox widget is useful when you need to add fixed-size gaps in between widgets or to set a minimum or maximum size for a widget.

15.

When do you use release mode and what’s the function of release mode?

Hide Answer

A release mode is used when one needs to optimize and produce the codes without any debug data. The release mode is used for deploying the app.
The command used to compile the release mode is Flutter run --release

16.

Can you name some popular database packages in Flutter?

Hide Answer

Here are two widely used packages in Flutter:

Firebase: It is used to use or modify the cloud database

Sqflite: It is used to modify the SQLite database.

17.

What are the advantages of a Flutter Inspector?

Hide Answer

The advantages of Flutter inspector are as follows:

Widget mode: With this button, you can choose different widgets from your application and inspect them one at a time. After choosing any Widget from the Widget Tree you can view the blueprint of that Widget with the help of the Layout Explorer tab, or you can check different properties and nested widgets of the selected Widget with the help of the Details Tree tab.

Refresh Tree: Whenever you make any changes in your application, and you hot reload those changes, they are not immediately reflected in your Flutter Inspector tool. To see the changes, you will have to click the Refresh Tree button.

Slow animation: You can reduce the speed of animation between layouts.

Invert Oversized Image: This will help to find oversized images. If developers want to create an application and they want to run it as fast as possible, but with heavy size and high-resolution images, it affects the performance of the application, so it will find the image and reduce the size of that image.

Debug paint- forms a border around each Widget.

18.

Define factory constructors.

Hide Answer

A factory constructor invokes another constructor, which is allowed to return an instance of a derived class, null or an existing instance. Factory constructors are defined using the factory keyword.

19.

List the responsibilities of FlutterActivity.

Hide Answer

The roles of a FlutterActivity are:

- Configuring the appearance of the status bar
- Displaying Android launch screen
- Displaying flutter splash screen
- Choosing an execution path for the Dart app bundle
- Adding transparency, which is an optional responsibility.
- Saving and restoring instance state.

20.

Name the different types of grid view in Flutter.

Hide Answer

The different types of grid views in Flutter are:

- count()
- custom()
- builder()
- extent()

21.

How would you implement a custom transition between screens in Flutter?

Hide Answer

You can use the PageRouteBuilder class to create custom transitions between screens. This class allows you to define custom animations, such as sliding or fading, for the page transition.

22.

Can you explain how to use the Provider package for state management in Flutter?

Hide Answer

The Provider package is a state management library that allows you to share data between widgets efficiently. You can use the Provider package to create a data model, expose it using a provider, and then access it in your widgets using the Provider.of method.

23.

What is the difference between Expanded and Flexible?

Hide Answer

Flexible takes the least space needed to fit in a child widget. On the other hand, expanded takes the rest of the size in the widget.

24.

Can you tell us some differences between const and final in Flutter?

Hide Answer

The only difference between const and final is that the const variables are evaluated at compile-time and are immutable whereas final variables are evaluated at runtime and can only be set once.

25.

How would you optimize the performance of a Flutter app?

Hide Answer

Some techniques to optimize the performance of a Flutter app include using the const keyword to make widgets immutable, avoiding unnecessary widget rebuilds, using the Provider package for efficient state management, and minimizing the number of expensive operations in the build method.

So, if your app is small, you can pass your data using the constructor of the widget class, but for a larger app, this is not an easy task. Unknowingly we use inherited widgets in many places while developing the flutter app.Theme.of(context), Navigator.of(context), and MediQuery.of(context).

26.

Can you describe how to implement internationalization in a Flutter app?

Hide Answer

Flutter provides the Intl package, which allows you to add support for multiple languages to your app. You can use the package to define messages in different languages, format numbers, dates, and times, and apply pluralization rules based on the language.

27.

Differentiate between setState and Provider?

Hide Answer

The setState() is used for managing the local state. Calling the setState() function notifies the framework about the change in the state of the object and that may affect the user interaction in the subtree.

Whereas, provider is a state management technique in Flutter that allows widgets to access data from a central location (i.e., a "provider"). Providers can be used to manage the application state, such as user authentication or data fetched from an API.

28.

Is it possible to nest a Scaffold? Why or why not?

Hide Answer

Yes, it is possible. Scaffold is a widget, so you can put it anywhere you want. Scaffold provides APIs like Drawer, SnackBar, BottomNavigationBar, FloatingActionButton, etc. However, it is not necessary to nest scaffolds as they can be used on their own.

29.

Can you explain how to use streams and futures in Flutter? Also, tell us some similarities and differences between Future and Stream?

Hide Answer

Streams and futures are powerful tools for handling asynchronous operations in Flutter. You can use a stream to listen for a series of events, such as user input or network responses, and react to them in real-time. A future, on the other hand, represents a value that may not be available yet and allows you to execute asynchronous code and retrieve the result when it's ready.

Similarities between future and stream:

- Both work asynchronously
- Both have some potential value

Differences between future and stream:

- Future has one response whereas a stream can have multiple responses.
- Futures are used in HTTP calls.
- A Stream is a series of futures.

30.

Can you tell us the difference between WidgetsApp and MaterialApp?

Hide Answer

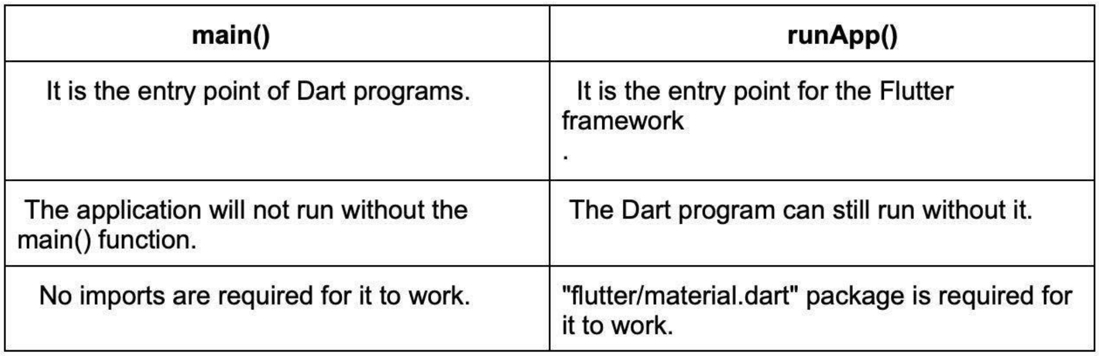
The main difference between WidgetsApp and MaterialApp is that the widget offers basic navigation, whereas MaterialApp is built on top of WidgetsApp.

31.

Can you state some difference between runApp() and main()?

Hide Answer

Main(): Main() function starts the program. You cannot write a program in Flutter without using the main() function whereas runApp() is used to launch the software. RunApp() allows you to return the widgets that are connected to the screen as the widget tree's root.



main()	runApp()
It is the entry point of Dart programs.	It is the entry point for the Flutter framework .
The application will not run without the main() function.	The Dart program can still run without it.
No imports are required for it to work.	"flutter/material.dart" package is required for it to work.

32.

What's the need of mixins?

Hide Answer

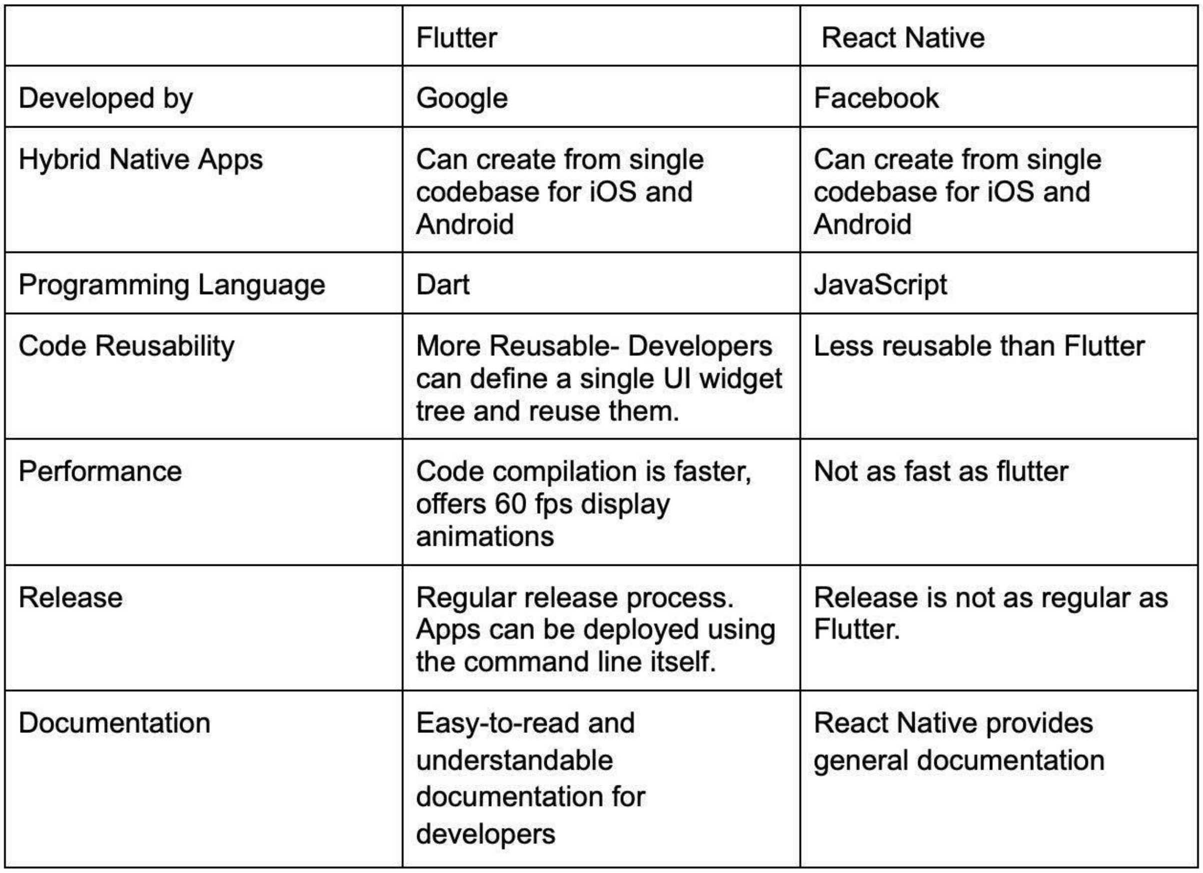
Multiple inheritances are not supported in Dart. Hence, you would need mixins to use multiple inheritance in Flutter, as they allow you to write reusable class code in multiple class hierarchies.

Advanced Flutter interview questions and answers

1.

Flutter Vs React: Which is better?

Hide Answer



	Flutter	React Native
Developed by	Google	Facebook
Hybrid Native Apps	Can create from single codebase for iOS and Android	Can create from single codebase for iOS and Android
Programming Language	Dart	JavaScript
Code Reusability	More Reusable- Developers can define a single UI widget tree and reuse them.	Less reusable than Flutter
Performance	Code compilation is faster, offers 60 fps display animations	Not as fast as flutter
Release	Regular release process. Apps can be deployed using the command line itself.	Release is not as regular as Flutter.
Documentation	Easy-to-read and understandable documentation for developers	React Native provides general documentation

It's difficult to decide which framework to choose as they both are exceptionally good and loved by companies and developers. However, you can pick one based on your particular needs.

2.

Why do we need separate directories for iOS and Android in Flutter?

Hide Answer

Here’s why you would need two different directories for Android and iOS files in Flutter:

Android Folder: You need an Android folder to keep files of an Android project. When any Flutter framework creates the code, the compiler converts the Flutter code into native code that is then imported to Android projects creating a native Android app.

iOS: This stores all the files for Mac projects. If you’re creating a project for the iOS platform, the iOS folder will be used. iOS apps can only be developed using the Xcode IDE and macOS in Flutter, which is why you’d need the iOS folder.

3.

Can you explain how to use Flutter's built-in navigation system to create a multi-level navigation hierarchy?

Hide Answer

Flutter's navigation system allows you to create a multi-level navigation hierarchy by using the Navigator widget to manage a stack of pages. You can push new pages onto the stack using the Navigator.push method and remove them using the Navigator.pop method

4.

Can you describe how to use Flutter's gesture recognition system to detect user input?

Hide Answer

Flutter's gesture recognition system provides a set of pre-defined gestures, such as tap, drag, and scale, that allow you to detect user input and react to it. You can use the GestureDetector widget to listen for gestures and the GestureDetector.onTap method to handle taps. You can also create custom gestures by subclassing the GestureRecognizer class.

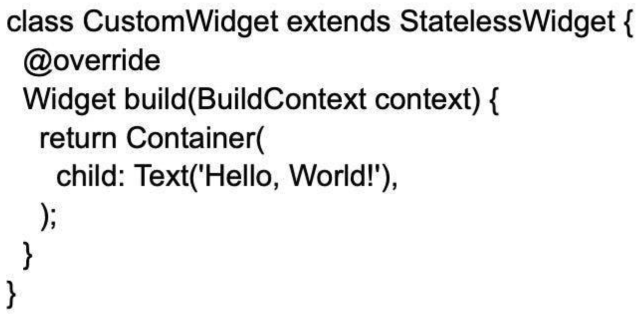
5.

How would you create a custom widget in Flutter, and what are the benefits of doing so?

Hide Answer

To create a custom widget in Flutter, you need to create a new class that extends either StatefulWidget or StatelessWidget, depending on whether your widget needs to be stateful or not. The benefits of creating custom widgets include reusability, encapsulation of complex logic, and improved code organization.

Example code:



```
class CustomWidget extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Container(
      child: Text('Hello, World!'),
    );
  }
}
```

6.

What is the difference between hot reload and hot restart?

Hide Answer

In Flutter, Hot Reload is the easiest and quickest way to make any changes, create UIs, fix bugs, and add features to the app. A Hot Reload performs all these changes rapidly and makes changes in the UI without disturbing the entire app.

Here's how to perform Hot Reload:

- Run the Flutter editor using command prompt or editor.
- Once the project is created, you can use debug mode and perform hot reload
- In windows, use "ctrl+\" to perform hot reload or using the button. For Mac devices, you can use "cmd+s" to perform hot reload. If you're using the command prompt enter 'r' to run.

A Hot Restart is different from Hot Reload as it demolishes the preserved states of the app and compiles the code from scratch.
Hot Restart takes more time as compared to Hot Reload but takes less time than restart function.
Here's how to perform Hot Restart:

- Run the editor using command prompt
- Make some changes and run the hot restart
- To start hot restart use the hot reload button or press ctrl+shift+.

7.

Can you describe how to use the Flutter animations API to create custom animations?

Hide Answer

The Flutter animations API provides a set of classes and widgets that allow you to create custom animations, such as tween animations or physics-based animations. You can use the AnimationController class to manage the animation's state, and the Tween class to define the animation's values over time.

8.

Can you describe in detail the Flutter IDEs?

Hide Answer

Some of the most popular Flutter IDEs include the following:

- IntelliJ IDEA is feature-rich, boasts extensive support for many languages, and provides smart coding assistance. Most beginner developers would probably get by with the free Community Edition, but the paid Ultimate Edition is required to access additional features.
- Android Studio works like IntelliJ IDEA but for Android only. Unlike the latter, though, Android studio is free and open-source. With smart coding assistance and a built-in debugger, it seems the best choice when programming for Android devices.
- Visual Studio code, or simply VS Code, is a free tool backed by Microsoft. It speeds up development and lowers development time. It works with Windows, macOS, and Linux.

As this is one of the most critical Flutter interview questions to guage a developer's knowledge, you can also ask what their favorite IDE is and their justification for picking one.

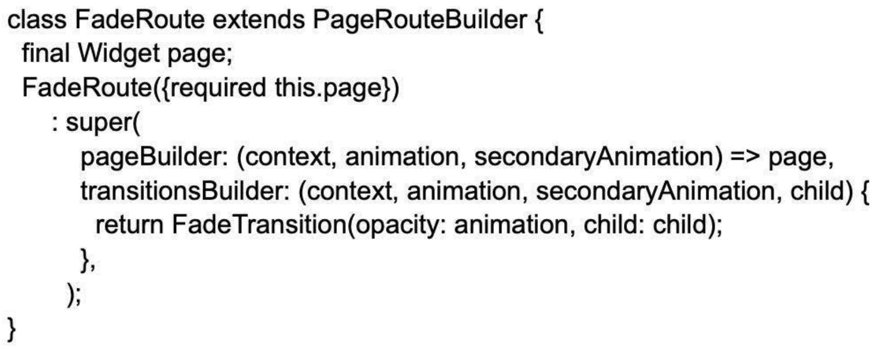
9.

How do you implement a custom transition between screens in Flutter?

Hide Answer

To implement a custom transition between screens, you can use the PageRouteBuilder class and provide a custom transitionBuilder function.

For example, to create a fade transition:



```
class FadeRoute extends PageRouteBuilder {
  final Widget page;
  FadeRoute({required this.page})
      : super(
          pageBuilder: (context, animation, secondaryAnimation) => page,
          transitionsBuilder: (context, animation, secondaryAnimation, child) {
            return FadeTransition(opacity: animation, child: child);
          },
        );
}
```

Then, to use the custom transition, you can simply push the new route:

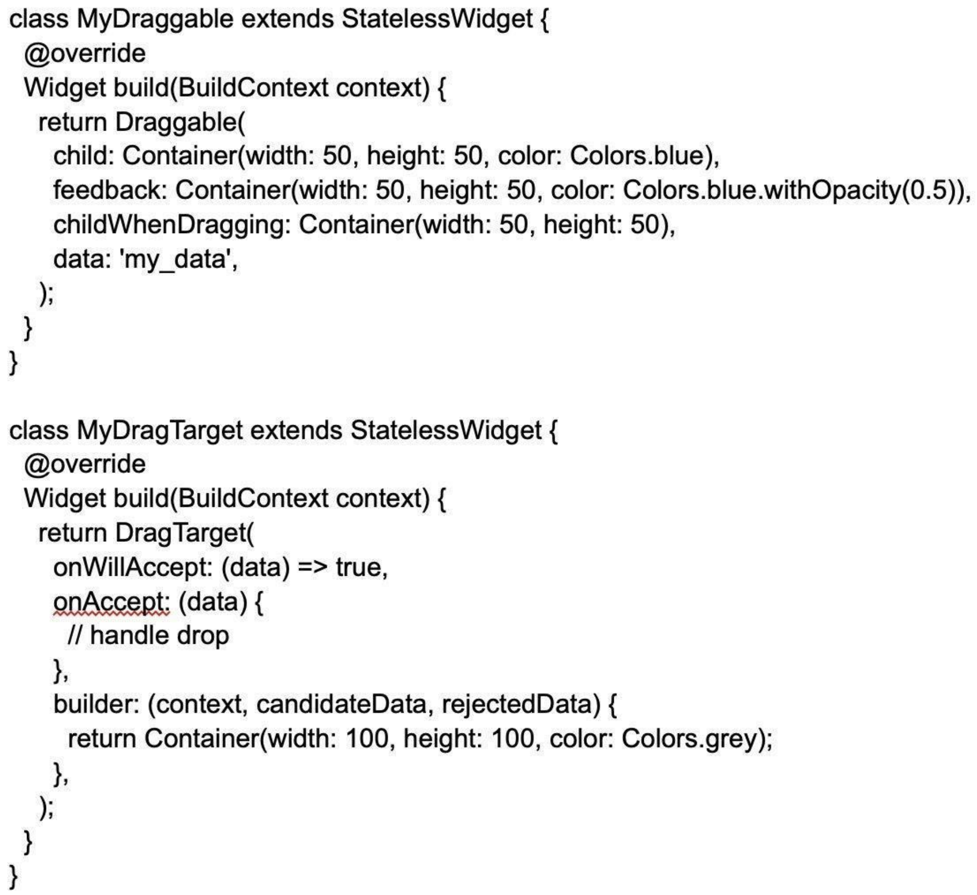
Navigator.of(context).push(FadeRoute(page: MyNewPage()));

10.

How do you implement a draggable widget in Flutter?

Hide Answer

To implement a draggable widget in Flutter, you can use the Draggable and DragTarget widgets. For example, to create a draggable widget that can be dropped onto a specific target:



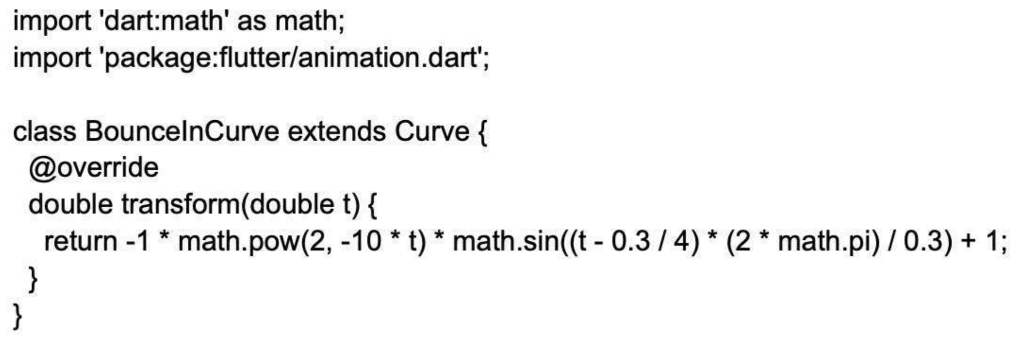
```
class MyDraggable extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Draggable(
      child: Container(width: 50, height: 50, color: Colors.blue),
      feedback: Container(width: 50, height: 50, color: Colors.blue.withOpacity(0.5)),
      childWhenDragging: Container(width: 50, height: 50),
      data: 'my_data',
    );
  }
}

class MyDragTarget extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return DragTarget(
      onWillAccept: (data) => true,
      onAccept: (data) {
        // handle drop
      },
      builder: (context, candidateData, rejectedData) {
        return Container(width: 100, height: 100, color: Colors.grey);
      },
    );
  }
}
```

11.

How do you implement a custom animation curve in Flutter?

Hide Answer

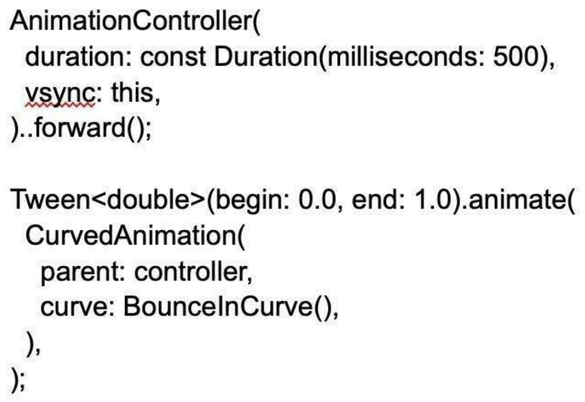
To implement a custom animation curve in Flutter, you can create a class that extends the Curve class and provide a custom implementation of the transform method. Here is an example of how to create a bounce-in curve:



```
import 'dart:math' as math;
import 'package:flutter/animation.dart';

class BounceInCurve extends Curve {
  @override
  double transform(double t) {
    return -1 * math.pow(2, -10 * t) * math.sin((t - 0.3 / 4) * (2 * math.pi) / 0.3) + 1;
  }
}
```

In the transform method, you can use any mathematical formula to create a custom curve. In this example, we're using a mathematical formula that simulates a bounce-in effect.

Then, to use the custom curve in an animation, you can simply pass it to the curve property of the Tween class:



```
AnimationController(
  duration: const Duration(milliseconds: 500),
  vsync: this,
)..forward();

Tween<double>(begin: 0.0, end: 1.0).animate(
  CurvedAnimation(
    parent: controller,
    curve: BounceInCurve(),
  ),
);
```

In this example, we're using the BounceInCurve as the curve property of the CurvedAnimation widget, which is then passed to the parent property of the Tween widget. This will create an animation with the bounce-in effect.