Share this article

Table of Contents

Download Top 100 Flutter Interview Questions and Answers Guide PDF For Free

Flutter Interview Questions and Answers For Freshers/Beginners

Flutter Interview Questions and Answers For

Flutter Interview Questions and Answers – Top 100 | Updated 2023

By Gourav Khanna / March 3, 2023 / Read Time: 28 min

Flutter is a popular choice among developers for building high-quality, cross-platform mobile applications quickly and easily. This article will cover all the important areas covered in an interview for a fresher as well as an experienced candidate. It will benefit not only the candidates appearing for an interview but employers can enjoy significant advantages too. Here, we are sharing a list of **Flutter Interview Questions** which are uniquely handpicked by our **Flutter developers** categorized into **Q&A for Freshers and Q&A for experienced**. So, Lets Begin!

Table Of Contents

Download Top 100 Flutter Interview Questions and Answers Guide PDF For Free

Download Flutter Interview Guide PDF!

Flutter Interview Questions and Answers For Freshers/Beginners

1. How would you define Flutter?

Answer: Flutter is an open-source mobile application development framework created by Google that allows developers to build high-performance, natively compiled applications for mobile, web, and desktop platforms from a single codebase. Flutter uses the Dart programming language and provides a rich set of customizable widgets and tools for building beautiful, fast, and responsive user interfaces.

Flutter also includes features like hot reload, which allows developers to see changes in the app's UI in real-time as they code, making the development process faster and more efficient. Additionally, Flutter has built-in support for integrating with other popular development tools and services, making it easier for developers to create and deploy their apps.

2. What are the two versions of Flutter available?

Answer: Google introduced Flutter in 2017. It is an open-source UI SDK used to create applications for both iOS and Android.

Flutter 2 was released on 3rd March 2021. Apart from existing features it additionally has many other features.

3. Is Flutter used for Front-end or Back-end development?

Answer: Flutter is a great choice for building front-end applications that need to be fast, responsive, and visually appealing. This technology is specifically designed for building mobile, web, and desktop user interfaces.

4. Name some apps built with flutter.

Answer: These are listed below:

Alibaba, Google Ads, Birch Finance, Reflectly, Hamilton Music, Coach Yourself, Hookie, Tencent, DropBox, and CryptoMarket.

5. Mention companies in India using Flutter.

Answer: HCL, Infosys, Tech Mahindra, MindTree, Hike, Capgemini, TCS, Wipro, Bajaj Allianz Life Insurance

6. How would you install Flutter?

Answer:

To install Flutter, follow these steps:

01. Download the Flutter SDK for your operating system from the official Flutter website: https://flutter.dev/docs/get-started/install
02. Extract the downloaded file to a location where you want to install Flutter.
03. Add the Flutter bin directory to your PATH environment variable. The Flutter bin directory contains the flutter tool, which is used to run Flutter commands. To add the Flutter bin directory to your PATH, follow the instructions for your operating system:

- **Windows:** a. Open File Explorer and right-click on “This PC”. b. Select “Properties” and then click on “Advanced system settings”. c. Click on the “Environment Variables” button. d. Under “System variables”, scroll down and find the “Path” variable. Click on “Edit”. e. Click on “New” and enter the path to the Flutter bin directory (e.g. C:\flutter\bin). f. Click “OK” to save the changes.
- **macOS and Linux**: a. Open a terminal window. b. Enter the following command: export PATH=”$PATH: [PATH_TO_FLUTTER_DIRECTORY]/flutter/bin”

04. Verify that Flutter is installed by running the following command in a terminal window: flutter doctor

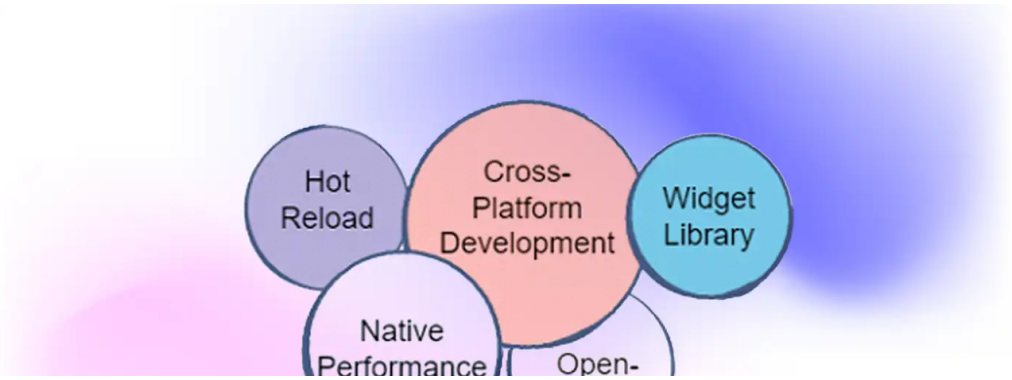
This command checks your environment and displays a report of the status of your Flutter installation. If there are any issues, the command will provide guidance on how to resolve them.

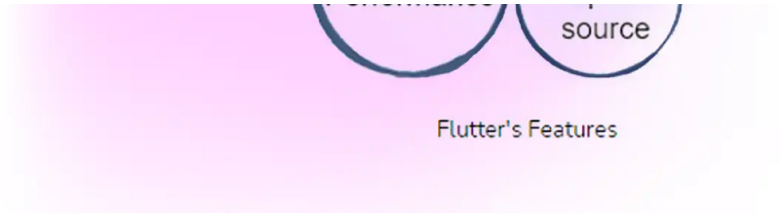
That’s it! You have successfully installed Flutter on your system.

7. What are the features of Flutter?

Answer: Some features of Flutter include:

- **Hot Reload**: This makes it possible to see the changes in the code instantly reflected on the UI. This speeds up the process to work on the outlook of the application enabling developers to correct errors that save cost and effort.
- **Cross-Platform Development**: Flutter enables developers to write code that works on different platforms. Two different applications can use the same codebase. In addition to sharing the UI code, the UI itself is also shareable. This makes maintenance of the single codebase much more accessible as opposed to different codes for different platforms.
- **Widget Library**: Everything in Flutter is defined as a widget. A widget can be a color, padding, or menu. Flutter can create complex widgets that can be customized according to the requirement of the application. Built-in widgets are also available for usage.
- **Native Performance:** Flutter can easily incorporate third-party integrations and API.
- **Open Source**: Google introduced Flutter as an Open-Source platform. It is free of cost and detailed documentation and communities available online.





8. What are widgets in Flutter?

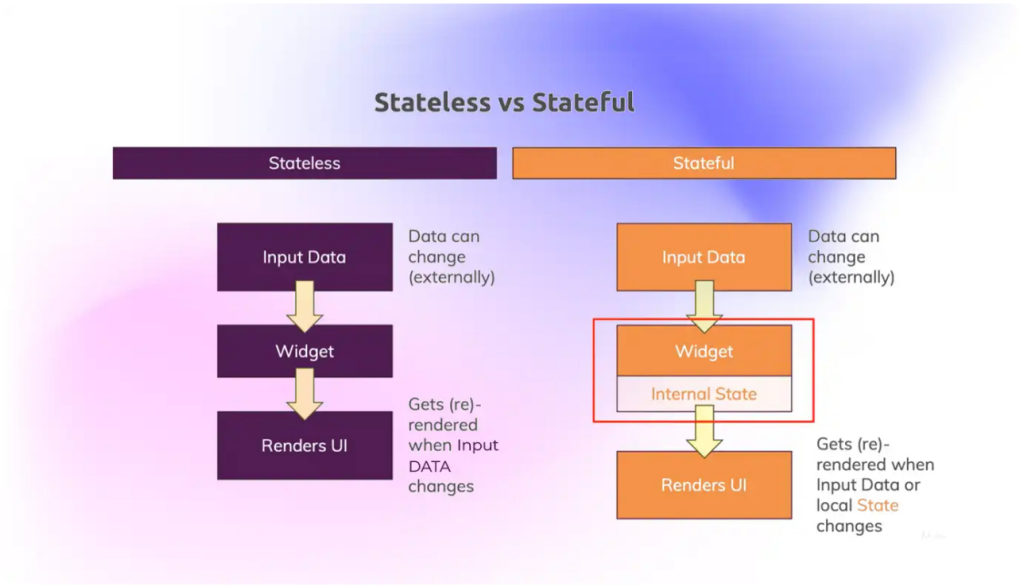
Answer: Widgets are the basic building blocks of a Flutter app. They are responsible for rendering the interface and handling user interactions. Flutter provides a rich set of pre-designed widgets, including basic UI components and complex animations.

9. What are the different types of widgets in Flutter?

Answer: Mainly, there are two types of widgets: Stateful Widgets and Stateless Widgets

10. Difference between Stateful and Stateless Widgets.

Answer: Stateful widgets hold the state of the device and can be rebuilt when the state changes whereas stateless widgets are built only when it is created or when the parent changes.



11. How to access Screen size in Flutter?

Answer: We can access screen size and other properties like pixel density, aspect ratio, etc by MediaQuery syntax MediaQuery.of(context).size.width.

12. What is Provider and how does it works?

Answer: The provider is a simple way for state management, to work on the concept of PUB-SUB which means there is one provider and multiple subscribers which are called consumers here, whenever there is a change, with NotificationChangeListener will update all the consumers.

13. What programming language is used in flutter?

Answer: A programming language named **Dart** is used in Flutter.

14. What are the pros of using Flutter?

Answer: Same UI and Business Logic in All Platforms. ...

– Reduced Code Development Time. ...

– Increased Time-to-Market Speed. ...

– Similar to Native App Performance. ...

– Custom, Animated UI of Any Complexity Available. ...

– Own Rendering Engine. ...

– Simple Platform-Specific Logic Implementation. ...

-The Potential Ability to Go Beyond Mobile.st Development: Flutter's "hot reload" feature allows developers to quickly and easily test changes to the code, which can significantly speed up the development process.

01. Cross-platform Development: Flutter allows developers to build apps for both iOS and Android using a single codebase, which can save time and resources compared to developing separate apps for each platform.
02. User-friendly UI: Flutter comes with a rich set of pre-designed widgets and tools for creating beautiful and responsive user interfaces, which can enhance the user experience of an app.
03. Performance: Flutter uses the Dart programming language, which is optimized for fast performance, and the framework itself is designed for high performance, making it a great choice for building fast, responsive apps.
04. Open-source: Flutter is an open-source project, which means that it has a large and active community of developers who contribute to the framework and help each other solve problems.
05. Hot Reload: The hot reload feature of Flutter enables developers to see the changes they make to the code in real-time, without having to restart the app, which speeds up the development cycle.

15. What are the cons of using Flutter?

Answer: -Few third-party packages. ...

– Inability to call Native APIs from Dart directly. ...

– Requirement of Dart for development. ...

– Lack of code push. ...

– Little overall Support. ...

– Few digital platforms. ...

– Limited complexity. ...

– Vector graphics and animation support.

16. Explain the Flutter architecture.

Answer: The architecture of Flutter, Google's UI toolkit for building natively compiled applications for mobile, web, and desktop from a single codebase, consists of several key components:

Dart Language: Flutter uses Dart, a statically typed, object-oriented programming language with C-style syntax. Dart is used for building applications in Flutter, and it provides features such as garbage collection, null safety, and a rich standard library.

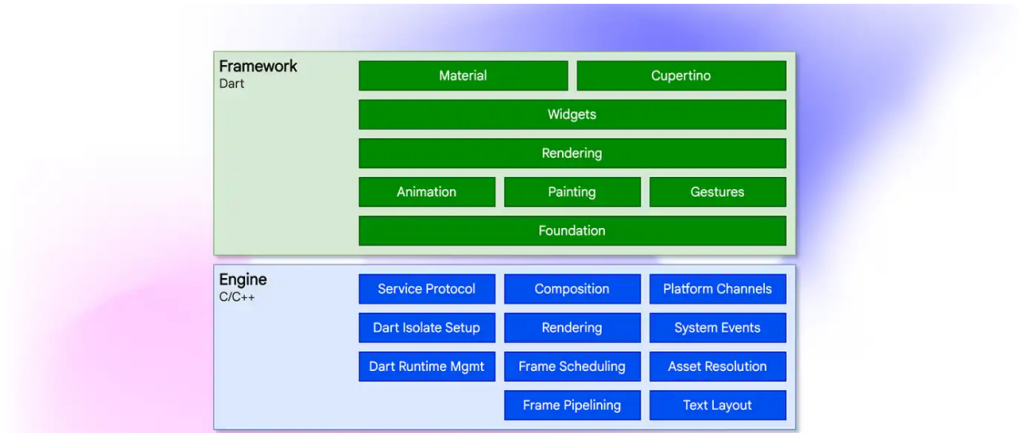
Flutter Engine: The Flutter engine is the core component of the framework, responsible for rendering graphics and animations, handling gestures, and managing the application's lifecycle.

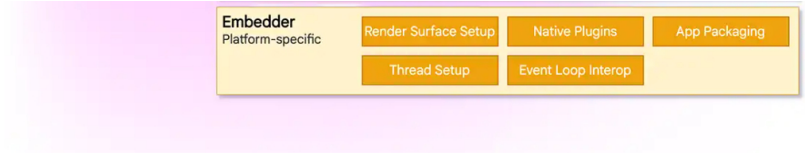
Widgets: Flutter uses a widget-based approach to building UIs, where widgets are the basic building blocks of an interface. There are two types of widgets in Flutter: stateless and stateful.

Render Tree: Flutter uses a high-performance render tree to draw the widgets on the screen. The render tree is a tree-like structure that describes the layout and appearance of the widgets.

Flutter Framework: The Flutter framework is a collection of libraries and tools that provide a set of common functionality to build applications, such as handling navigation, data persistence, and network communication.

Flutter Package Ecosystem: Flutter has a thriving package





17. What are the supported databases in Flutter?

Answer: -Firebase Realtime Database

– hive

– ObjectBox

– SQLite

– Moor

18. What is a dart programming language?

Answer: Dart is a client-optimised, object-oriented programming language developed by Google. It was introduced in 2011 and is used to build web, server, and mobile applications. Dart has a syntax that is similar to other C-style languages, making it easy to learn for developers who are familiar with languages such as Java or JavaScript.

Dart is statically typed, which means that the type of a variable must be declared before it can be used. This makes the code more predictable and helps catch errors early in the development process. Dart also provides features such as asynchronous programming, garbage collection, and a rich standard library.

19. What is Flutter's guiding principle?

Answer: Flutter, Google's UI toolkit for building beautiful, natively compiled applications for mobile, web, and desktop, is guided by several principles that inform its design and development:

01. **Beautiful and Engaging User Interfaces**: Flutter's goal is to make it easy for developers to create beautiful, engaging user interfaces that run smoothly on multiple platforms. To achieve this, Flutter provides a rich set of customizable widgets and tools for building animations and effects.
02. **Fast Development and Hot Reload**: Flutter's fast development cycle is a key part of its guiding principle. The "hot reload" feature allows developers to see the changes they make to the code instantly, without having to restart the app. This makes it possible to iterate quickly and improve the user experience faster.
03. **Native Performance**: Flutter's goal is to provide the performance of a native app with the ease of development of a cross-platform framework. Flutter uses its own graphics engine to render the user interface, which means that it can achieve native-level performance even on low-end devices.
04. **Accessibility:** Flutter is designed to be accessible to everyone, including people with disabilities. The framework provides tools and widgets that make it easy to build accessible interfaces, and it follows accessibility guidelines and best practices.
05. **Reactive Programming:** Flutter's architecture is based on reactive programming, which allows for real-time updates to the user interface in response to changes in data. This approach makes it possible to build dynamic and responsive interfaces that respond quickly to user interactions.

These principles are what drive the development of Flutter, and they help to ensure that the framework is fast, flexible, and easy to use.

20. Describe your technique for reducing execution time running code.

Answer: Depending on the size and adaptability of the app packages, I try to integrate just-in-time compilers to run code, as it helps enhance the app's performance. Dart executes code more quickly than other widely used languages. Ahead-of-time compilers can also shorten execution times. When merging sequential streams, I found that integrating these compilers results in the greatest run-time reduction and functionality increase in applications like game development

21. How is flutter better than its alternatives?

Answer: Flutter uses its special programming language, Dart, whereas other app development tools utilize mainstream languages, on top of pre-existing frameworks. Flutter's native app assembly makes creating cross-platform apps much easier and less labor-intensive. The hot reload capability enables developers to make real-time code changes, which is highly helpful, especially for testing and debugging. As the application does not require restarting to reflect changes, it saves time. Flutter is an affordable solution for businesses looking to

create native apps for multiple operating systems.

22. What skills are required to use Flutter?

Answer: To use Flutter effectively, you should have knowledge of the following skills:

Object Oriented Programming, Mobile app development, Reactive Programming, Widgets, Layouts, design, Debugging, and Testing.

23. How does Flutter differ from React native?

Answer: React Native and Flutter are amazing choices for developing a cross-platform application, however, they share several differences which make them unique from each other.

One major difference is the programming languages both frameworks use. React Native is based on JavaScript and uses JSX. However, Flutter uses Dart programming language.

The architecture for both frameworks differs. Flutter uses a 2D graphic rendering library, Skia, whereas, React Native uses the Flux architecture, which also requires JavaScript bridging.

24. What are the different build modes in Flutter?

Answer: There are three different build modes in Flutter:

Debug-This is the most common mode for testing apps.

Release-This mode is used for deploying the app on marketplaces

Profile-This is the mode you are looking for

25. What is the use of Ticker in Flutter?

Answer: 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.

26. Explain Flutter inspector.

Answer: The XML file allows us to view our app's blueprint and properties. There is a powerful tool called Flutter Inspector for Flutter applications that allows you to visualize the blueprint of your widgets and their properties. Using it, you can diagnose various layout issues and understand the current layout. Flutter Inspector offers the following benefits:

- Select widget mode
- Toggle platform
- Show paint baselines
- Show debug paint
- Refresh widget
- Enable slow animations
- Show/hide performance overlay

27. What type of tests can you use in Flutter?

Answer: Flutter, being a modern cross-platform development framework, provides various testing tools and techniques to ensure the quality of your app. Some of the tests used in Flutter include:

01. **Unit tests:** These tests check the functionality of individual classes or functions in isolation. They are run on the command line and are faster than other tests.
02. **Widget tests**: These tests check the behavior and rendering of individual widgets. They are similar to unit tests and run on the command line.
03. **Integration tests:** These tests check the interaction between various parts of the app, such as between widgets, services, and databases. They are slower than unit and widget tests but provide more comprehensive testing.
04. **End-to-end (E2E) tests**: These tests run on a real device or emulator and test the entire app from the user's perspective. E2E tests are used to catch issues that are not discovered by other tests.
05. **Performance tests:** These tests measure the performance of your app, including its speed and resource usage.

Each of these tests has its own strengths and weaknesses, and it's important to choose the right test for each scenario to ensure that your app is thoroughly tested.

28. What are the keys in Flutter and when to use them?

Answer: The key is an identifier for Widgets, Elements, and SemanticsNodes.A new widget will only be used to update an existing element if its key is the same as the key of the current widget associated with the element.

Keys must be unique amongst the Elements with the same parent.

Subclasses of Key should either subclass LocalKey or GlobalKey.

Keys are useful when manipulating collections of widgets of the same type.

If you find yourself adding, removing, or reordering a collection of widgets of the same type that hold some state, then, you should use a key.

29. How many child widgets can be added to the container widget?

Answer: Only one child widget

30. Why does a flutter app usually take a long developing time?

Answer: The first time you build a Flutter application, it takes much longer than usual since Flutter creates a device-specific IPA or APK file. Xcode and Gradle are used in this process to build a file, which usually takes a lot of time.

31. In the Flutter development, we can insert three rows in a column and an image in every row. Is it true or False?

Answer: True

32. Which one is better: Flutter or React Native?

Answer: The choice between Flutter and React Native will depend on your specific needs and requirements. If you are already familiar with JavaScript, React Native may be a good choice for you. If you are looking for a fast and visually appealing app, Flutter might be the way to go. It's always best to do your own research and evaluate both options to see which one is best for your project.

33. What are the resources to learn Flutter?

There are many resources available to learn Flutter, including:

01. **Official Flutter documentation**: The Flutter team provides comprehensive documentation on their website that covers everything from getting started to advanced topics. The documentation includes code samples, tutorials, and API references.
02. **Flutter YouTube channel**: The Flutter team maintains an official YouTube channel with videos covering a range of topics, including getting started with Flutter, building Flutter apps, and using Flutter with other technologies.
03. **Flutter codelabs**: Google provides codelabs that guide you through building Flutter apps step-by-step. These codelabs are interactive and provide a hands-on way to learn Flutter.
04. **Online courses**: There are many online courses available on platforms like Udemy, Coursera, and Pluralsight that teach Flutter. These courses can range from beginner-level to advanced and typically include video lectures, quizzes, and hands-on projects.
05. **Online communities**: There are many online communities, such as the Flutter Dev Google Group, Flutter Discord server, Flutter subreddit, and Flutter Community on GitHub, where you can ask questions, get help, and interact with other Flutter developers.
06. **Books**: There are several books available on Flutter, including "Flutter in Action" by Eric Windmill, "Flutter for Beginners" by Alessandro Biessek, and "Flutter Cookbook" by Erica Stormer. These books cover various topics related to Flutter development.

34. What is flutter used for?

Answer: Flutter is Google's open-source technology for creating mobile, desktop, and web applications with a single codebase. Unlike other popular solutions, flutter is not a framework or library, it's a complete SDK i.e. software development kit.

35. How do I prepare for the Flutter interview?

Answer: 1. Review the basics: Brush up on the basics of Dart, the programming language used in Flutter. Also, familiarize yourself with the core concepts of Flutter, such as widgets, streams, Futures, and the Flutter

architecture.

2. **Study the Flutter documentation:** The Flutter documentation is an excellent resource that provides detailed information on how to use Flutter to build mobile applications. Make sure you have a good understanding of the topics covered in the documentation.

3. **Practice coding**: Start building small applications using Flutter to get a hands-on experience with the framework. Try to solve real-world problems, and work on projects that you can use in your portfolio.

4. **Brush up on architecture patterns**: Know the most commonly used architecture patterns in Flutter, such as BLoC, Provider, and Scoped Model.

5. **Familiarize yourself with common tools**: Know how to use common tools such as the Flutter CLI, Android Studio/Visual Studio Code, and the Flutter inspector.

6. **Prepare for common interview questions**: Review common Flutter interview questions and be prepared to answer them. Some of the common questions include: How does Flutter handle updates to its widgets? What are streams in Flutter? How does Flutter handle state management?

7. **Get familiar with the company:** Research the company you're interviewing with and try to understand its development process, as well as its product offerings.

36. What is Flutter Framework?

Answer: The Flutter framework consists of both a software development kit (SDK) and a widget-based UI library. This library consists of various reusable UI elements, such as sliders, buttons, and text inputs.

37. What is Dart's concept?

Answer: Dart is an object-oriented language. It supports object-oriented programming features like classes, interfaces, etc. A class in terms of OOP is a blueprint for creating objects. A class encapsulates data for the object. Dart gives built-in support for this concept called class.

38. How is Dart executed?

Answer: Dart Web enables running Dart code on web platforms powered by JavaScript. With Dart Web, you compile Dart code to JavaScript code, which in turn runs in a browser—for example, V8 inside Chrome. Dart web contains two compilation modes: An incremental development compiler enabling a fast developer cycle.

39. What are available trees in Flutter?

Answer: The three trees that make the Flutter framework beautiful and strong are:

Widget Tree

Element tree

Render Object tree

40. Explain the build process of Flutter.

Answer: Flutter isn't compiled directly into iOS or Android apps.

Apps are launched based on a combination of a rendering engine (built on C++) and Flutter (built on Dart). All files generated this way attach to each app and SDK assemblies software for a specific platform.

So, these were the **flutter interview questions** for beginners/freshers. Now, we will move ahead on to **Flutter Interview Question and Answers** for Experienced/Advanced Developers.

Flutter Interview Questions and Answers For Experienced/Senior/Advanced Developers

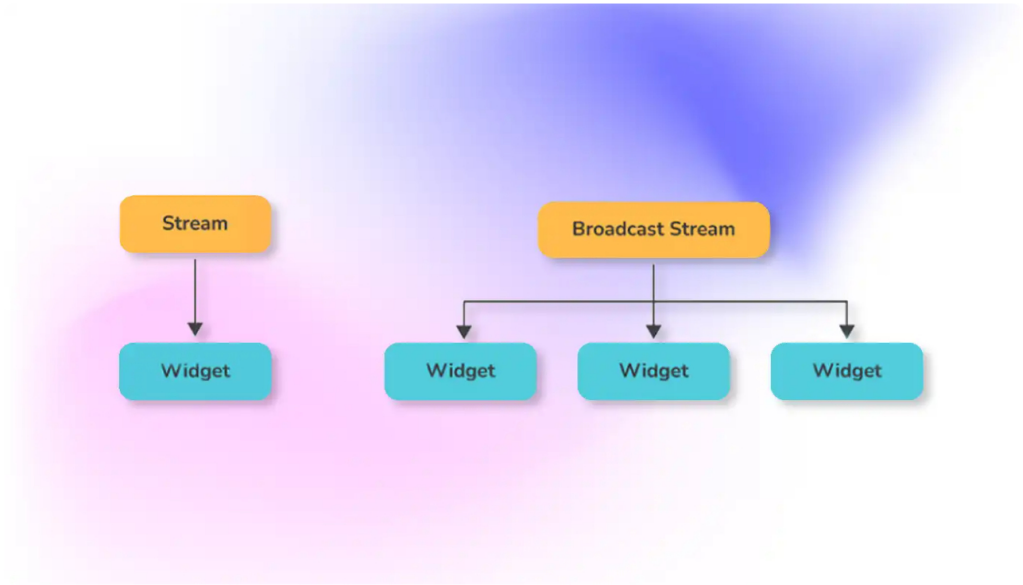
41. What do you mean by Streams?

Answer: 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 steam controller.

The Stream API provides the await for and listen() methods for processing streams. Streams can be created in many ways, but they can only be used in the same manner. Here is an example:

```
Future<int> sumStream(Stream<int> stream) async {
  var sum = 0;
  await for (var value in stream) {
    sum += value;
  }
  return sum;
}
```

42. What are the different types of Streams?



Answer: The streams' functionality is part of Dart and is inherited by Flutter. In Flutter, there are two kinds of streams:

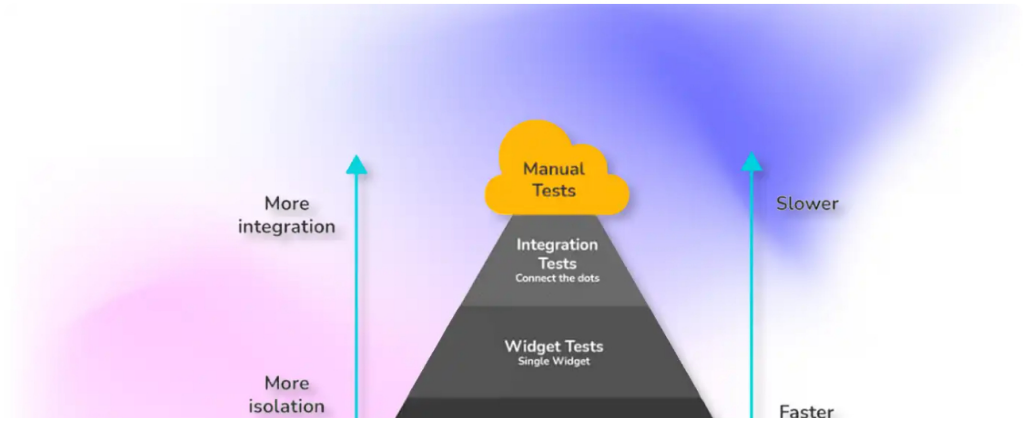
Single Subscription Streams: These streams deliver events sequentially. They are considered as sequences contained within a larger whole. These streams are used when the order in which events are received matters, such as reading a file. There can be only one listener throughout the sequence, and without a listener, the event won't be triggered.

Broadcast Streams: These streams deliver events to their subscribers. Upon subscribing to events, subscribers are immediately able to start listening to them. These are versatile streams that allow several listeners to listen simultaneously. Furthermore, one can listen again even after canceling a previous subscription

43. What do you mean by Widget testing?

Answer:

- **Unit tests**: Using unit testing, you can test a class or method. Unit tests do not check for rendering to screen, interacting with external services, or user interactions.
- **Widget tests:** Using widget testing, you can test a single widget. This ensures that the widget's UI looks as expected and responds appropriately to events. In other words, it ensures that the widget design, rendering, and interaction with other widgets are up to the mark.
- **Integration tests:** Using Integration testing, you can test the critical flows of the entire app. It is important to check whether all widgets and services work together as expected. You can also use it to measure and benchmark the performance of your app.



44. What do you understand about Tween animation?

Answer: The shortened version of in-between animation is tween animation. The start and endpoints of an animation must be specified in tween animation. Using this method, the animation can begin at the beginning and can progress through a series of values until it reaches the endpoint. Transition speed and duration are also determined by using the tween animation. Calculating the transition from the beginning to the end will be easier with the widget framework.

45. What is the difference between Sideboxed and container?

Answer: Container: In this parent widget, multiple child widgets can be easily controlled and handled by adjusting their size, padding, and color efficiently. We can wrap a widget in a container widget if it needs any styling, like a color, a shape, or a size constraint, etc.

Sized Box: This is a specific size box. It does not allow us to set the widget's color or decoration, unlike Container. In this case, we only need to resize the widget that is passed as a child. In other words, it forces its child widget to have a specific size.

46. Explain Flutter Provider.

Answer: The provider is built using widgets. You can use all the objects in the provider as if they were just part of Flutter with the new widget subclasses it creates. This also means that the provider is not cross-platform. The provider is the simplest way to handle state management. Basically, it works on the concept of PUB-SUB i.e., there is one provider and several subscribers

47. How can we create HTTP requests in Flutter?

Answer: To create HTTP requests, use the HTTP package (import 'package:http/http.dart' as http;). In the following manner, we can make the Requests:

```
http.get('https://jsonplaceholder.typicode.com/albums/1');
```

It will return a Future <http.Response>.

48. Explain BuildContext.

Answer: 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

49. Explain pubspec.yaml file.

Answer: 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:

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

50. Difference between runApp() and main() in Flutter.

Answer: main() is the entry point of a Flutter application. It is the first function that is called when the app starts running. The main() function typically sets up the root widget for the app, creates an instance of the MyApp class, and calls runApp() with the MyApp instance as an argument.

runApp() is a Flutter function that takes a Widget as an argument and inserts it as the root widget of the Flutter app. The runApp() function is called with the root widget of the app, and it sets up the widget tree, starts the

framework, and begins rendering the user interface.

In other words, the main() function is the starting point of the Flutter app, while runApp() is responsible for rendering the user interface. The main() function sets up the necessary environment for the app to run and then calls runApp() with the root widget, which starts the rendering process.

51. What is the difference between “async” and “await” in Flutter?

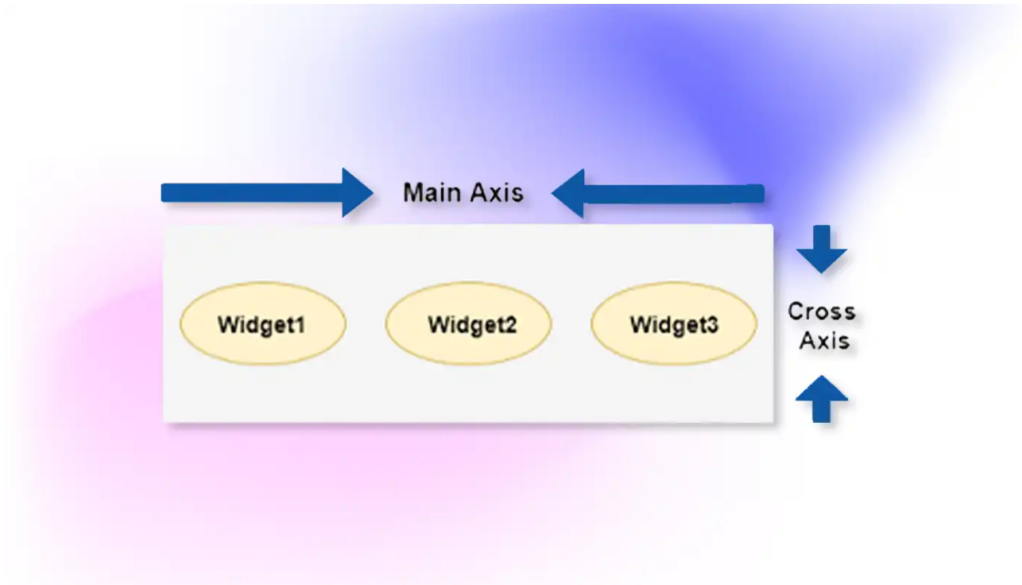
Answer: “async” is used to mark a function as asynchronous, whereas “await” is used to wait for the result of an asynchronous operation before continuing with the execution of the function.

52. What is the purpose of the “main.dart” file in a Flutter app?

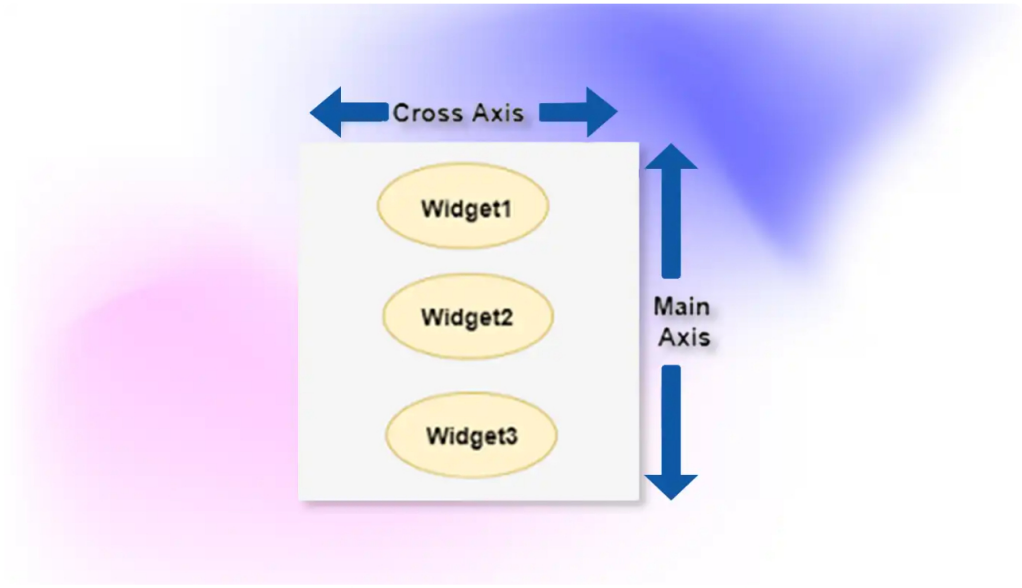
Answer: The “main.dart” file is the entry point of a Flutter app. It contains the main() function, which is the starting point of the app. It also sets up the app’s initial configuration, such as the root widget of the app.

53. When do you use main Axis alignment and cross Axis alignment?

Answer: Main Axis alignment and cross Axis alignment is used to control how row and column widgets align their children based on our choice. The row’s cross-axis will run vertically, and the main axis will run horizontally.



The column’s cross-axis will run horizontally and the main axis will run vertically. The following figure will clarify the concept more.



54. What is lazy loading?

Answer: Lazy Loading is a design pattern commonly used in computer programming and most commonly in web design and development to defer the initialization of an object until a point at which it is needed. It can contribute

to efficiency in the program's operation if properly and appropriately used.

55. How to hide the android status bar in Flutter?

Answer: You can use SystemChrome.setEnabledSystemUIOverlays([]) to hide and SystemChrome.setEnabledSystemUIOverlays(SystemUiOverlay.values) to bring it back in.

56. What are the similarities and differences between Future and Stream?

Answer: Similarities

01. Stream and Future both work asynchronously.

02. Both have the same potential

Differences:

01. A stream may be a mixture of Futures.

02. Future has only one response yet Stream can have any number of Responses.

57. Explain Threading in Dart.

Answer: Threads allow a program to operate more efficiently by doing multiple things at the same time. Threads can be used to perform complicated tasks in the background without interrupting the main program.

58. Why do we require Mixins?

Answer: Dart does not endorse multiple inheritances. Therefore for implementing multiple inheritances in Flutter. We require mixins. Mixins offer a way for providing a way for writing reusable class code in multiple class hierarchies.

59. What is Treeshaking in Flutter?

Answer: Tree Shaking is the optimization technique for removing the unutilized module in the bundle in the build process. It is also the code elimination technique used for optimizing our application.

60. Differentiate setState and Provider.

Answer: setState()

We use it for managing the local state in a similar stateful widget and its child. The downside is everything is in the same class like UI code, business logic, and mixin UI, which splits clean code principles.

Provider

In the provider pattern, we define everything in a separate class indicating that the UI presentation is defined in the different logics that specify different thus code appears high quality and clean. Moreover, we don't need to transmit the state from one screen to another through the constructor.

- The business Logic Component(BLoC) pattern is a common way to manage the state in Flutter apps. You may be asked questions about it in the interview.

61. What is BLoC and how does it differ from other reactive frameworks?

Answer: BLoC is a reactive programming framework that uses streams to emit data in response to events. This is different from other frameworks like ReactJS, which uses a more traditional event-based model. The advantage of using a stream-based approach is that it allows for a more declarative style of programming where the developer does not need to worry about explicitly handling events.

62. What kind of design patterns can be leveraged with BLoC?

Answer: BLoC can be used with a variety of design patterns but it is most commonly used with the Model-View-controller(MVC) and Model-View-Presenter (MVP) patterns.

63. Can you provide more details about error handling in Flutter?

Answer: There are a few different ways to handle errors in Flutter. The first is to use the try/catch method. This involves wrapping your code in a try block and then catching any errors that occur in the catch block. You can also use the assert method to check for errors. This will cause the program to stop if the condition is not met. Finally, you can use the onError callback. This will allow you to handle errors in a specific way.

64. What are some common BLoC patterns that are used when developing an application in Flutter?

Answer: Some common BLoC patterns that are used when developing an application in Flutter include the

following:

- The use of StreamController to emit events that will be processed by the BLoC.
- The use of StreamBuilder to listen for events emitted by StreamController and rebuild the UI in response to those events.
- The use of a Sink to provide input to the BLoC from outside sources
- The use of a Stateful Widget to manage the state of the BLoC.

65. Is there any difference between RxDart and vanilla dart Streams?

Answer: Rx dart streams are able to work with data types that are not just simple streams of data. This allows for more complex data structures and operations to be performed on the stream. Additionally, Rx dart streams can be used to perform operations such as filtering, mapping, and reducing which are not possible with Vanilla Dart Streams.

66. What are some advantages of using constructs like TextField, CheckBox, RadioButton, etc. over standard HTML elements like input, select, checkbox, radio button, etc.?

Answer: Some advantages of using these constructs over standard HTML elements are that they provide more flexibility and functionality. For example, the TextField construct can be used to create text input fields that can be used to validate data while the CheckBox construct can be used to create checkboxes that can be used to track selections.

67. What is the best practice for making network requests in Flutter?

Answer: The best practice for making network requests in Flutter is to use the built-in HttpClient class. This class provides a simple way to make HTTP requests and receive responses from a web server.

68. What is the purpose of FutureBuilders in flutter?

Answer: FutureBuilders are used to build widgets that depend on data from a Future. When the Future completes, the widget is rebuilt with new data. This is useful when you need to perform an asynchronous operation to obtain data before building a widget.

69. What is the easiest way to search for data in a ListView?

Answer: The easiest way to search for data in a listView is to use the built-in search functionality provided by the ListView class. This can be accessed by calling the search() method on a ListView instance.

70. How do you use the StreamProvider to manage the state within a Flutter application?

Answer: The StreamProvider is a widget that provides a stream of data to its children. In order to use it, you will need to create a stream of data and then pass that stream to the StreamProvider. The StreamProvider then handles listening to the stream and updating its children accordingly.

71. Is multiple inheritances possible in Flutter?

Answer: Unlike Java, dart also doesn't support multiple inheritances.

72. What is dependency injection in Flutter?

Answer: Dependency injection in Flutter is a technique in which one object supplies the dependencies of another object. A dependency is an object that can be used in a class. It can be a Network service, Database Service, Location service, etc.

73. What is the Factory method in flutter?

Answer: The factory method is referred to as a creational design pattern that provides an interface for creating objects in a superclass but allows subclasses to alter the types of objects that will be created.Also known as virtual constructors.

74. Can you use multiple scaffolds in Flutter?

Answer: Yes but that is precisely what we want when we make two separate screens each one with its own scaffold.

75. Define setState().

Answer: We use it for managing the local state in a similar stateful widget and its child. The downside is everything is in the same class like UI code, business logic, and mixin UI, which splits clean code principles.

76. What is the difference between Inkwell and GestureDetector?

Answer: The main difference between InkWell and GestureDetector lies in the material widget. InkWell must have a material widget as an ancestor. However, there is no such compulsion for the GestureDetector Widget. Otherwise, in many ways, these two share many common features.

77. Why did the first Flutter app take so long?

Answer: When we build the Flutter application the first time, it will take more time. It is because Flutter developed a device-specific APK or IPA file. So, the XCode and Gradle are used for building the file, taking more time.

78. What is a Spacer widget?

Answer: Spacer manages the empty space between the widgets with a flex container. Even with the row and Column MainAxis alignment, we can manage the space as well.

79. How is Flutter native?

Answer: Flutter uses only the canvas of the native platform and draws the UI and all other components from scratch. All the UI elements look the same as the native ones. This mainly reduces the burden of the time for converting some languages to native ones and speeds up the UI rendering time. As a result, the UI performance is remarkably high.

80. How can you update a listview dynamically?

Answer: By using setState to update the listview item source and rebuild the UI.

81. When can you use double.INFINITY?

Answer: When you want the widget to be big as the parent widget allows.

82. How would you access StatefulWidget properties from its state?

Answer: Using the widget property

83. Mention two or more operations that would require or turn you to the Future?

Answer: 1. Calling API using HTTP

2. Getting results from the Geolocator package

3. With FutureBuilder widget

84. What is the purpose of a SafeArea?

Answer: SafeArea is basically a glorifying padding widget. If you wrap another widget with SafeArea, it adds any necessary padding needed to keep your widget from being blocked by the system status bar, notches, holes, rounded corners, and other creative features by manufacturers.

85. When to use a mainAxisSize?

Answer: When you use mainAxisSize on your column or row, they will determine the size of the Column or Row, along the main axis i.e height for the column and width for the row.

86. What is the difference between these operators '??' and '?'

Answer: ?? expr1 ?? expr2 If expr1 is non-null, returns its value; otherwise, evaluates and returns the value of expr2.

?. Like. but the leftmost operand can be null; for example: foo?.bar selects property bar from expression foo unless foo is null (in which case the value of foo?.bar is null)

87. What is a vsync?

Answer: Vsync basically keeps the track of screen so that Flutter doesn't render the animation when the screen is not being displayed

88. How is an inherited widget different from a provider?

Answer: Provider basically takes the logic of InheritedWidgets but reduces the BoilerPlate to the strict minimum.

89. Why do we need mixins?

Answer: Mixins are very helpful when we want to share behavior across multiple classes that don't share the same class hierarchy or when it does not make sense to implant such behavior in a superclass

90. When do you use the WidgetsBindingObserver?

Answer: To check when the system puts up the app in the background or returns the app to the foreground

91. What are Global Keys?

Answer: Global keys are like global variables. Try not to overuse them. There are almost always better ways to preserve the state globally using the right state management solution.

92. When to use double. infinity ()?

Answer: I want to be as big as my parent allows(double. infinity)

I want to be as big as the screen(MediaQuery)

93. How to bind the list in reverse?

Answer: ListView.builder(

shrinkWrap: true,

reverse: true,

itemCount: 2,

itemBuilder: (context, index){

return listDesign(index);

},

),

94. What is the package?

Answer: Packages are readymade tools or libraries which we can use instead of reinventing and developed by someone else.

95. Can we use color and Decoration properties simultaneously in a container?

Answer: No, the Color property is a shorthand for creating a BoxDecoration with a color field. If you are adding a box decoration, simply place the color on the box decoration.

96. What is a text editing controller?

Answer: When the user modifies a text field with an associated texteditedController, the text field updates the value and the controller notifies its listeners.

97. What is the difference between flutter and another cross-platform tool?

Answer: Flutter allows you to develop apps much more quickly. Not only it is a cross-platform SDK but it also includes the builder tool as well as numerous user interface components. As a result, building applications quickly become possible.

98. Explain Flutter Stateful life cycle?

Answer: In Flutter, a Stateful widget has a state object that stores mutable state data. The state object is created when the widget is first inserted into the widget tree and persists throughout the lifetime of the widget. The **stateful widget's lifecycle** consists of several methods that are called at different points in the widget's lifecycle:

01. **createState()**: This method is called when a Stateful widget is instantiated, and it creates the associated State object for the widget. This method is called only once in the lifecycle of the widget.
02. **initState()**: This method is called after the state object is created and inserted into the widget tree. This method is typically used to perform initialization tasks that are required for the widget to function correctly, such as setting up event listeners, initializing variables, or loading data from a database.
03. **didChangeDependencies()**: This method is called whenever the widget's dependencies change. Dependencies can include data from inherited widgets or changes to the app's global state. This method is typically used to update the state of the widget based on changes to its dependencies.
04. **build()**: This method is called whenever the widget needs to be rebuilt, such as when the state of the widget changes or when the widget is first inserted into the widget tree. This method returns the widget tree that represents the current state of the widget.
05. **didUpdateWidget()**: This method is called whenever the widget is rebuilt with a new configuration, such as when the parent widget changes or when the widget is being updated with new data. This method is typically used to update the state of the widget based on changes to its configuration.

06. **setState()**: This method is used to update the state of the widget and trigger a rebuild of the widget tree. When this method is called, the build() method is called again to create a new widget tree that reflects the updated state.
07. **deactivate():** This method is called when the widget is removed from the widget tree. This method is typically used to perform cleanup tasks, such as canceling event listeners or releasing resources.
08. **dispose()**: This method is called when the state object is no longer needed and can be safely disposed of. This method is typically used to release resources or perform final cleanup tasks.

99. What are the two types of streams available in Flutter?

Answer: There are two types of streams available in a flutter:

Single-Subscription Streams and Broadcast streams.

100. How do you handle l10n in your apps?

Answer:

In Flutter, localization (l10n) refers to the process of adapting an app's user interface and content to different languages, regions, and cultures. Flutter provides built-in support for localization, making it easy to create apps that can be localized for different audiences.

To handle l10n in your Flutter apps, follow these steps:

01. Add the `flutter_localizations` package to your project's dependencies in the `pubspec.yaml` file. This package provides support for localization in Flutter.
02. Create a folder named `l10n` in your project's root directory. This folder will contain the localization files for different languages and regions.
03. Create a localization file for each language and region that you want to support. The file should be named in the format `<locale>.arb`, where `<locale>` is the language and region code (e.g. `en_US` for English in the United States). The file should contain a JSON object with the translations for the app's content.
04. Use the `flutter_localizations` package to set the app's supported locales and delegate the localization work to the `GlobalMaterialLocalizations` and `GlobalWidgetsLocalizations` classes.
05. Use the `Localizations` widget to wrap the app's root widget and specify the app's default locale.
06. Use the `Localizations` widget to retrieve the localized text and other resources from the `AppLocalizations` class.
07. Use the `Intl` package to format dates, times, numbers, and other data according to the user's locale.

By following these steps, you can create a localized Flutter app that can be adapted to different languages, regions, and cultures. This can help you reach a wider audience and provide a better user experience for your app's users.

So, these were the **flutter interview questions** for Experienced/Advanced/ Senior Level Developers. Now we have a bonus **Flutter MCQ Section** included which includes the 10 most commonly asked **Flutter Multiple Choice Questions** along with their Answers.

Best Flutter Interview Multiple Choice Questions (MCQs)

1. Which widget is used to create a button in Flutter?

01. Text
02. Image
03. Icon
04. **ElevatedButton**

Answer: 4

2. Which of the following widget is used for repeating content in Flutter?

01. **ListView**
02. ArrayView
03. Expanded View
04. None of the above

Answer: 1

Answer: 1

3. What does SDK stands for?

01. Software Data Kit

02. **Software Development Kit**

03. Software Database Kit

04. None of the above

Answer: 2

4. Flutter is developed by:

01. IBM

02. Microsoft

03. **Google**

04. All of the above

Answer: 3

5. The first alpha version of Flutter was released in:

01. **2017**

02. 2016

03. 2018

04. 2019

Answer: 1

6. The example of Stateless widgets are

01. Row

02. Column

03. Text

04. **All of the above**

Answer: 4

7. Which programming language is used to build flutter applications?

01. Kotlin

02. Java

03. **Dart**

04. Go

Answer: 3

8. Which function is responsible for starting the program?

01. runapp()

02. **mainapp()**

03. run()

04. flutter()

Answer: 2

9. What type of test can examine your code as a complete system?

01. Unit tests

02. Widget tests

03. **Integration Tests**

04. All of the above

Answer: 3

10. What type of Flutter animation allows you to represent real-world behavior?

01. **Physics-based**

01. Physics-based

02. Maths-based

03. Graph-based

04. Sim-based

Answer: 1

Conclusion – Flutter Interview Questions (Beginner to Advanced)

In conclusion, we have covered a wide range of **Flutter interview questions and answers** in this blog, including topics such as Flutter widgets, state management, layout, navigation, and more. These questions are designed to help you prepare for your next Flutter interview and demonstrate your proficiency in Flutter development.

We hope that this blog has been helpful to you and that you feel more confident in your Flutter skills and knowledge.

Remember, preparation is key to success in any interview, and practicing with these questions can help you showcase your expertise and stand out as a strong candidate for any Flutter development position. Good luck in your Flutter career!

About author

Gourav Khanna

Gourav Khanna is co-founder and CEO of APPWRK IT SOLUTIONS PVT LIMITED, a web & mobile app development company. He is a technophile who is always eager to learn and share his views on new technologies and future advancements. Gourav's knowledge and experience have made him one of the industry's most respected and referenced leaders in the IT industry. His passion for writing and a high spirit of learning new things is reflected in his write ups. He has inspired many organizations to leverage digital platforms with his top-notch writing strategy skills that cut through the noise, backed by sharp thinking. Gourav believes that - "Words are the way to know ecstasy, without them life is barren ".

Related Post

Our Premium Clientele

and many more...

APPWRK Clients' Success Stories

OUR LOCATIONS

18723 Via Princessa #204 Santa Clarita CA 91387, United States.
+1 (718) 313-0706 (US)

1798 Fox Grape Loop, LUTZ FL 33558, United States
+1 (718) 313-0706 (US)

107, 30 Campbell st. Blacktown NSW -2148
+61 478 495 863 (AU)

142 Onstein, Amsterdam 1082 KM.
+31.687.944.230 (NL)

101 First Floor, WORLD TECH 67 ITC 10, Near Municipal Building, Sector 67, SAS Nagar, Mohali Punjab 160062
+91.986.500.0760 (IN)

1242, 447 Broadway, 2nd Floor, New York, NY, New York, US, 10013

HR Department & Job Inquiry
01724120318, 8288070950, 9041176950, 8699765760, 6239648958

SERVICES

Quality Assurance
Digital Marketing
Developer Referral
iOS APP Development Agency

QUICK LINKS

Careers
Privacy Policy
Sitemap

DEDICATED DEVELOPERS

Hire React JS Developers
Hire Angular Developers
Hire Webflow Developer
Hire Remote Developers
Hire Laravel Developers
Hire Android Developers
Hire Blockchain Developers
Hire Shopify Developer

SOLUTIONS

IT Outsourcing
Portfolio

HAVE QUESTIONS?

Call Us at
+1.718.313.0706

Skype: live:appwrk.global
Email: contact@appwrk.com

4.7 star rating by 1.2K+ APPWRK Clients on over 2.1K+ Projects

Email Address Subscribe