1. What is Flutter

Ans. Flutter is a framework developed by google, also we can say that it is sdk – Software Development Kit, flutter made in dart language, we can build not only mobile apps but also web and desktop application, it uses dart as intermediately language, also we can access native features as listed below, It is open source and free

1. For android – java/kotlin
2. For ios – swift/objective – c
3. For mac os – objective c
4. For windows – c++
5. For linux – c,

Flutter consist folders for android, ios, macos, linux and windows it has lib folder we can put dart file into it, to create flutter application from command line we can use flutter create project\_name,

Another method to create flutter project using view menu -> command palate and type flutter, there will available options for creating projects

To execute flutter code we can use flutter run command or if we want to run project from debug console we can use run without debugging from run menu

It has feature because of which we can see details of program execution in

A. Debug Console

B. Terminal

2.what are third party libraries

Ans. Example of third party libraries are

1. Firebase
2. Google Maps

Third party libraries are not part of flutter framework but it is used to extend functionality of an application,

This libraries are open-source and can be used to add feature and functionality to flutter application.

3. what are flutter Disadvantage or Limitation

Ans.

1. Limited third party libraries: In flutter lots of packages and plugins are available but framework is still relatively new and number of third party libraries are limited compared to react native

2. Steep learning curve: For some developer who are not familier with basic languages, the learning curve is challanging

3.Limited Corporate Adoption: As compared to react-native, flutter is still relatively new so Adoption by large number of organization is limited

4.App Size: Because of Flutter Engine and Framework included, flutter apps have large sizes compared to native apps,

However subsequent upgrades have made an attempt to optimize app Size

5.Package Compatibility: There are instances where packages are not compatible for each other or with specific platform versions, this creates integration issue and requires additional trouble shooting.

4. what are Flutter Advantages (8 points)

Ans.

1.cross platform development: Flutter not only support ios and android but also mac os,windows, web and linux.

2.Hot reload: According to this feature we can see changes in UI without restarting whole app, it preserve state and loads code changes into VM.

3. Access to native Feature: we are able to write native code in flutter for example

Kotlin/Java on Android

Swift/Objective C on ios

C on linux

C++ on windows

Objective c on Mac Os.

4. Rich UI Experience : With the help of lots of Flutter Widget, we can create User Interface which looks beautiful and neat clean

5. Ecosystem and Community: With the help of large community of plugin – packages, flutter offers wide range of functions which are easy to integrate with flutter app

6. The vibrant and helpful flutter community for tools , tutorials, support, making it simple to learn and easy to understand

7. Flutter is open source and free to use on it which developed by google.

8. Faster compilation, it consist of dart VM, it has JIT AND AOT compiler

5. What is use of Ticker in flutter

Ans: Ticker is a mechanism which control the timing of animation by providing callback at regular interval, To use ticker we can use “TickerProviderStateMixin”.

6. What are different build modes available in flutter

Ans.

1. Debug Mode: This mode enables debugging of an App on physical device/emulator/ or simulator

2. Profile Mode: On emulators and simulator, this mode is disabled because it does not provide real-world performance, enough to analyze performance while testing

flutter run --profile

3. Release Mode: when we want to deploy app this mode is useful, Faster start-up, Faster execution and less size are key factors of this mode

flutter run --release

7.Diffrence types of testing

1.unit testing:Using this kind of method, we can test class or method.

2.widget testing: Using this kind of testing you can test single widget

3.integration testing:Using this kind of testing you can test flow of the app, it is important to check that all services and widget work together as expected

8.Explain Stream in detail

Ans. Stream is like a pipe where you put value at one end and receive value at another end. It is Sequence of Asynchronous events, It is like Asynchronous iterable, instead of getting next event when you ask for it, it tells when next event is ready

Advantages:

The owner of Stream can emit value when it is available and it doesn't need to know who is listening or why listening

Mainly two types:

1. Single Subscription Stream: This kind of stream can listened to only once, it doesn’t generate events until it has listeners and stops sending events when listener stops

Ex: This kind of stream is used when download file or any single use operation

2.Broadcast Stream: A broadcast stream allow any number of listeners, it fires when it’s events are ready whether there are listeners or not, If you need that multiple parts of your app access the same stream you can use broadcast stream.

To get value from stream is termed as subscribing or listening to a stream

1. Sink B. Source.

9.what is variable in dart

Ans. variable is a container which used to store value in program, there are different types of variables types :

1.int -> to store value like 1,2,3,0,-1,-5.

2.String-> to store values like ‘A’ ,’b’.

3.double -> to store vakues like 4.0, 5.5,-1.9.

4.num ->to store int and double values like 10.9,45,-1,-0.9.

5.bool->to store either true or false.

6.var-> to store any value like ‘a’, 23,44.4,true,false etc, var is dynamic until we assign a value to variable suppose if we write var a = 10, the data type of a will be an integer

7.dynamic->If a variable is dynamic its type can changed at anytime.

Once variable is declared with var it’s type can’t be changed after once assigned

10. Data types in flutter/dart

Ans. Datatypes specify which kind of data you want to store in variable, below specified different datatypes:

1.List -> ordered group of items

2.map -> store data in key – value pair

3.set -> unordered collection of values , we can’t get values by index as they are unordered

5.Runes -> with runes you can find Unicode value of String

ex: void main() {

String value = "a";

print(value.runes);

}

o/p: (97)

6.num -> it contains positive/negative of int or double, it represent numeric value

7.string -> it represent sequence of character

8.bool -> it represent either true or false

9.int -> represent integer value either negative or positive

10.double-> represent integer value with decimal point

11.null -> it represent null value

11. Comments In dart

A.single Line :

Ex: //

B.multiple Line:

Ex: /\* \*/

C. document Line:

Ex: For generating documentation or reference for project or software package

///

12.Operators In dart

Ans: Operators in dart is used to perform mathematical and logical operation on variables, each operation in dart uses a symbol called operator to denote the type of operation it perform

Operands : It represent the data

Operators : It represent how operands will processed to produce a value

13.Diffrence between React native and Flutter

Ans:

1. React native is based on java script which is more popular and Flutter is based on dart, dart is less famalier than javacsript, dart may hard or take more time to learn than javascript for new developers
2. Community and Eco-System: React native has larger and more community than flutter,flutter’s community is growing rapidly faster, react native has huge 3rd party libraries
3. User Interface: Flutter has its own rendering engine named skia, means it can provide highly customizable user interface, on the other end react native use native component which can be customized to some extent but it is not powerful as flutter

Development Speed: Flutter has a feature named hot reload according to it we can view changes in ui without restarting the app which is fast then react – native

1. Performance: Flutter is known for it’s performance,single code-base which compiles directly to native code, on the other end react native uses a bridge to communicate between java script and native code
2. React-Native use native component and flutter uses widgets which are not native

14. Explain Stateful widget lifecycle:

Ans.

1.createState: when flutter is instructed to build stateful widget, this method is called, it immediately calls this method

2.initState: this method called once when page loads, we can initialize variables, subscribe or listen to stream,

3.didChangeDependency:

Dependency means external data or services that a widget relies for it’s functionality

called immediately after initState when dependency of a widget change

4.build : it build ui tree, it has parameter named buildContext

5.didupdateWidget: This method is called at the time of widget at rebuild and parent widget changes it’s configuration

6.setState:state is an information ,if we want to update information we can use this method, whenever there is a change in internal information it rebuilds ui.

7.de-activate : it is called when state object is removed from tree but it might be re – inserted before current frame is finished.

8.dispose : when state Is completely removed from widget tree this method can be used, we can unsubscribe/cancel/dispose controllers,streams animations

15.Ephermal State and App state (State managment)

Ans.

App State: known as Global State, this type of state represent data and state which can be shared across multiple screen or widgets within the app,

The app state can include data such as user authentication status , app preferences, theme settings and it can modify by different parts of program

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Local State: known as Ephemeral State , it refers to the state or information which used and managed by single part of the app, this type of state usally store information which changes frequently and not needed by any other part of an application.

Local state is managed by stateful widget class, setState method is used to rebuild widget tree when state is changed

Local state can be useful for managing UI elements that depend on user interaction or input ex: Form input, animation etc.

16.Flutter Inspector

17. Flutter Architecture

Ans. Flutter architecture divided into 3 parts:

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

1. Framework: it consist of below parts

A. Material And Cupertino : It presents widgets usally material means android type and Cupertino means ios type,

B. widgets: it provides collection of pre-built widget, stateful or stateless widget and custom widgets

C.Rendering : convert Code into pixel-perfect widget UI

D.Animation / Painting / Gestures.

E.Foundation:

2. Engine:

- written in c++,

-It has dart vm which executes code

-It supports hot reload according to that we can see changes in screen without restart the app and app state will preserved

-It convert code into beautiful and responsive design even on low-level devices,

-It handles entire flutter application lifecycle

It consist of below Parts:

1.Frame Scheduling

2.Frame Pipelining

3.Dart Isolate Setup

4.Dart Runtime management

5.Text Layout

6.Asset Resolution

7.composition

8.Rendering

9.Platform Channels

10.Service Protocol

11. System Events.

3. Embedder: It acts as bridge between host platform and flutter frame work, it consist of following parts:

1. Native Plugins

2. Event Loop Interop

3. Thread Setup

4. Render Surface Setup

5. App Packaging

18.what is dart and write it’s importance

19.what is use of Animation Controller

Ans. Animation controller is used to control and manage animation, it providers way to start stop and pause animation along with direction, duration and speed of animation

20. how you can test single widget

Ans. Using the widget test method you can test the widget, you can test that various portions of the user-interface work as intended without need of physical device or simulator

21.can you use widgetsApp for basic navigation

Ans.yes a material app builds a navigator which manages stack of widgets identified by string, also known as root objects and give you two ways to manage the stack.

22. what is “Tree Shaking”:

Ans. Removing dead code and unused module from the app which helpful to improving performance and reduces code size

23.Diffrence between Synchronous and Asynchronous task:

Ans:

Synchronous: In the task tree previous task must be completed in order to execute next task, it block whole task tree, for example if we have four task named A,B,C,D and to complete task B it takes 5 seconds, the execution of task c done after 5 seconds, it is done by one after one

Asynchronous: it executes task simultaneously suppose we have four task in task tree named A, B, C, D, suppose task B takes 3 seconds, it will execute in order such as it don’t wait other task B to complete first it may execute Task A, after that it executes task C and D and after 3 seconds it will executes task B

Ex:

1. reading file

2. downloading file

3. Fetch and send data to internet

4. write something to database

24. what is Future:

Ans. Future tell that value of function will be available at some time in future,

It has two states:

1. uncompleted and 2.completed

future may be completed with value or error

uncompleted: it waits for function to wait for finish or throw an error

completed:it can be completed with value or completed with error, Future<int> produce integer value, Future<String> produce string value

ex:

void main(){

print(‘First’);

getData();

print(‘Third’);

}

Future<String> printString() {

return Future.delayed(Duration(seconds:5), ()=> “HELLO”);

}

Void getData() async {

String data = printString();

Print(data);

}

O/P: Example of above function will be

First

Third

HELLO

25.layout Builder

Ans: Layout builder Widget depend on parent widget for width and height, and it provides constraint according to that

layout builder has builder method, it has parameters like 1.BuildContext 2.BoxConstraints, according to BoxConstraint it has properties like maxWidth/minWidth/maxHeight/minHeight, this constraint are specified on child widget

26. How to achive internationalization in flutter App,

Ans.

To achieve internationalization, we can use intl package in flutter app it provides international language and format number and date and time etc

27. what are difference between packages and plugins

Ans.

Packages and plugins are both we can add as dependency in pubspec.yaml file, but it has following difference

Packages:

Packages are set of classes interface and other packages, it is used to create modular code that can be shared easily.

Plugins:

Plugins are written is native language, suppose you want to access camera of a mobile phone you can use camera as plugin

28.can you describe how to use flutter animation api to create custom animation

Ans.

Flutter animation api provides set of classes and widgets that allow you to create custom animation, such as tween and physics based animation, flutter provides animation controller which is used to create and manage animation and we can start /stop/ pause animation with duration/direction.

Below Steps To Use Animation Api in Flutter Project:

1.import necessary package: for example material and math package

2.create animation controller

3.Define an Animation

4.Implement an Animation With Animated Builder

5.Start And Stop The Animation

29. benifits of Dart

Ans.

1. Dart is declarative and programmatic so developer can understand read and visualize dart code very easily
2. It support majority of basic programming language concepts like below:
3. class, B. interface, C. Function, D. Constructor , E. mixin F. abstract class and methods , G. Generics, H. inheritance

Also it provides additional data type such as 1.list 2.map 3.set

1. For better performance and fast execution dart support Dart VM uses

1. AOT – Ahead Of Time and,

2. JIT – Just In Time Compilers

4. Dart code executes several times faster than java script

5. Dart is Object Oriented Programming Language which makes it scalable and stable for creating complex application

30. what is null assignment operator

Ans. Null Assignment operator is denoted by =??

Ex. Int a;

a?? = 10;

as per above example value of a is null and it assign 10 to a

31. What is overlay widget in flutter

Ans. Overlay widget perform floating on another widget as a separate layer, suppose we have drop down, we can create overlay widget which display menu on another widget means floating menu,

Ex: dropdown menu or pop up menu

32. How do you handle complex Gestures and touch interaction in flutter

Ans.

33. How to achieve Localization and Internationalization

Ans. To achieve internationalization we can use intl package which comes with various international language and various date formats, Internationalization means design app that have various languages and make it adapt for various regions

Localization means accept translating and adept app for particular region and languages

34. what is Flex widget

Ans. As per flex widget, it provides flexible space in which child widgets are laid out linearly along the main Axis either horizontally or vertically

We can use flex widget in conjuction with expanded or flexible widget

35. What are flutter’s best practice for handling state management for large and complex application

Ans.

1. Lazy Loading: Only load that data which is necessary to improve performance
2. MobX : Reactive State Management, it Efficiently updates UI, using observable, data computation and action
3. Bloc/Cubit : Bloc stands for business logic component, it has separate layer and code from business layer, means presentation and logic layers are different
4. Avoid using setState : use of setState can lead to performance issue in large apps, so it is better to use State-Management package
5. Redux : It ensures that every state change is predictable by centralizing the state and defining actions

36. what is “shallow” and “deep” widget tree, when you use it

Ans. Shallow widget tree consist fewer nested widgets, leading to a simplified structure that is easier to manage and debug,

It is used for simple layouts where complexity is minimal

\*\*\*\*\*\*\*

Deep Widget Tree: Multiple layers of nested widgets, complex and rich user interface

37. Explain BuildContext:

Ans.

BuildContext is a parameter of build method this method found in Stateful or Stateless widget,

In flutter every widget has its own build context

Buildcontext parameter is used to identify widget in widget tree, also it determine relationship with parent and child widget ,

It represent location of widget

=============================================================

38. usage scenario for keys:

Ans.

Keys are used to uniquely identify widget in a widget tree,

Ex:

Suppose we have listview and we can uniquely identify each item of listview using key and can do any operation whatever we want

Forms: if we want that data should not be loose of widget that take input we can use keys

State-Preservation :

State remains attached to its widget regardless of its position.

Keys preserve state when widget move around widget tree

39. Global Key :

Ans. used to identify unique widget globally

40. what is difference between ink well and gesture detector

Ans. Inkwell provide splash effect while gesture detector does not provide it, Gesture Detector is used when you need to handle variety of gesture and want more flexibility in defining response to that gesture

41. Diffrence between material app and widget app

Ans.

Material App follow material guideline which we can say that android style widget

While widgets app does not follow either material or Cupertino means ios or android type

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Material app comes with theming, styles,

While widget app demonstrate functionality but without style or theming

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Material App Provides high level material design widgets

While widgetapp provides general purpose widgets

42. what are routes:

Ans. routes is the property of material app, the data types of routes are map, map with string as key and widget function as value

Ex:

InitialRoute: ‘/’,

Routes : {

‘/’ : (context) => const FirstScreen();

‘/second’ : (context) => const SecondScreen();

}

Ex:

onPressed: () {

Navigator.pushNamed(context,’/second’);

}

}

=============================================================

43. purpose of material app widget

Ans. it provides theming and styling and many more,

It has also property named home , in this parameter we have to specify page name whatever we want to load

It also provides navigation based on routes

44. purpose of mediaquery:

Ans.

The purpose of media query widget is to get dimension and orientation of the screen, in other words we can get device specific height and width of particular devices

45.explain safeArea widget:

Ans. The purpose of Safe Area widget is to prevent the UI element from being obscured by certain parts of the display

46. purpose of listview:

Provide scrollable list,

Direction : either vertical or horizontal

Ex: mobile phone contact list,

Message in chat

shrinkWrap Property : when set to true listview only occupies space it needs rather than taking available entire height, suppose if I have height of listview by using sizedbox is 350, and it has more than 100 items, then it occupies height only 350 and it is scrollable within height of 350,

============================================================

Stack follow LIFO -> Last In First Out

47. explain SafeArea widget.

Ans. Safearea is a widget which gives facility that your content is displayed within safe area of the screen and not obscured by system elements such as notches or status bar

=============================================================

48. Explain concept of Flutter Theme and how to create custom theme

Ans. Theme in flutter is defines that how screen of an app looks like, it may same for single or more pages or may be different pages design

Flutter also support dynamic theming, means we can change apps theme at run time, ex: switch between dark mode and light mode

To create custom theme you can use ThemeData object,

Theme is a way to define and customize visual appearance of an App,

49. Differences between expanded and flexible

Ans. Expanded takes all available space to occupy child widget while Flexible will take space which are defined

50. how to open Flutter Inspector  
Ans. To open flutter inspector first run your project and then in visual studio code press shift + ctrl + p 🡺 A window will appear and select open dart dev tools 🡺 then select open dev tools in web browser 🡺 it will display flutter inspector page.

51. what are Mixin

Ans. To create mixin, you can use mixin keyword, mixin have capability to use it by keyword “with”, we can use properties and method of mixin, without using inheritance. We can use multiple mixin that extend the functionality of a class, it solves problem of “Multiple Inheritance”,

Mixin in flutter are commonly used for sharing code related themes,logging animations or any other functionality that can be re-used across multiple widgets or classes.

Limitation Of Mixin:

1. We can not create CONSTRUCTOR in mixin
2. We can not create object of mixin directly
3. Order of mixin is important when we are going to use them using with keyword

Explanation : suppose we have two mixin named FirstMixin And SecondMixin both mixin have same method named “doSomething” and we want to use both mixin in a class named “MyClass” For example:

mixin FirstMixin{

void doSomething(){

print(‘Do Something first mixin’);

}

}

Mixin SecondMixin{

void doSomething(){

print(‘Do Something Second mixin’);

}

}

Class MyClass with FirstMixin, SecondMixin{

}

void main(){

MyClass myClass = MyClass();

myClass.doSomething();

}

As per above example when we want to initiate doSomething() method it will print 🡺 “Do Something Second mixin” because method of FirstMixin method overrides/overreturn by SecondMixin

52: what is flutter inspector

Ans. Flutter inspector is a tool for visualizing, inspect and interact widget tree of flutter application during development time, we can do following with Flutter Inspector:

1. Interact with widget tree :
2. Inspect specific widget with widget tree :
3. Know properties and state of each widget:
4. Debugging UI Issues:
5. Debugging Layout Issues:
6. Performance Profiling :
7. Widget Customization
8. Hot reload and Hot restart: Flutter inspector seamlessly work with hot reload and hot restart.

53.OOPS concepts :

1.class: class is a blue print or template for creating objects, class consist of A. Properties and

B. Methods, in other words it consists of attributes and behaviour, we can say that class has 1.variables 2. Getter/setter 3.constructors 4.abstract methods 5. Interface 6. Method/functions etc.

To declare class we can use class Keyword

2.object: variable of class type is known as object, with the help of object we can access properties and behaviour of class, whenever we create object it occupies memory in ram

3.polymorphism:

Poly -> many and Morphism -> forms , ability of an object to take on multiple forms

Two types:

1. Compile Time 🡪 Method Overloading
2. Run Time 🡪 Method Overriding,

1.Method Overloading: according to traditional concept it means that method with same name but different parameters, it may have different return type also, but dart does not support this kind of Overloading,

In dart it can achieved using optional and named parameters, To create named parameter, we can use ‘Curly Braces – {}’ and for optional we can use ‘[]’ this parameter named as Square Brackets

1. Method Overriding : Run time Polymorphism known as method overriding,

It means that method which already present in super class, same method can be found in subclass we can use @override as annotation, to access method of super class you can use ‘super’ keyword, in short sub class and super class have same method name and same method parameters

4.Constructor :

Constructor is a Special Member function which have same name as class name, the use of constructor is to initialize class properties, dart support multiple constructor, dart can have default/parameterised/named constructors. Constructor can not return value,

Factory Constructor : it is a special type of constructor which can be declared with factory keyword it can return another constructor, we can initialize variables in factory constructor also,

5.Inheritance: it’s a capability of dart that one class can use properties and method of another class, the sub class is the class which use properties and method of another class, and super class is the class whose properties and method used by child class, to use inheritance we can use ‘extend’ keyword

A. Single Inheritance: according to it one base class derives properties and methods of single super class

B.Multi – Level Inheritance : according to it there are situation that, suppose there are 3 classes A, B and C, class B extends A and class C extends B, means class c have permission to use properties of class A and B

C. Multiple Inheritance : suppose one class extends by 2 or more super class known as Multiple Inheritance, it is not supported by dart, to achieve multiple inheritance we can use concept of mixin or interface

D. Hierarchical Inheritance: suppose we have one parent class and this class extended by two different child class this situation is known as hierarchical inheritance

1. Abstract Class:

To create abstract class we can use ‘abstract’ keyword, abstract class can have both methods abstract (means without implementation) and non-abstract (concrete methods – methods with implementation), we cannot create object of abstract class directly, the class which we extends as abstract class we have to implement all methods which are abstract in child class

Abstract class can extends Another abstract class,

Abstract class can implement multiple Another abstract class,

1. Interface : To declare an interface, we can not have keyword like “interface” as like other languages, interface solves problem of multiple inheritance, one class can implements any number of interfaces, we can use abstract class which can be interface and we can implements it, the main condition is that class which claims to implement as interface must provide concrete implementation of all methods.
2. Mixin: It’s a concept that solves problem of multiple inheritance, to declare mixin we can use mixin keyword, to use it we can do it by “with” keyword one can use any number of mixin, by using mixin you can use properties and methods of mixin into particular class

Limitation Of Mixin:

1. Mixin does not have constructor
2. Order of mixin is important:

Suppose we have 2 mixin named firstMixin and secondMixin and both have method with same name draw(), and we call it in particular class then method of first mixin is overwritten by method of secondMixin

1. we can not create object of mixin directly

54. which skills require to use flutter

Ans. 1.Knowledge of dart

2. Knowledge of widgets

3. API Integration

4. State Managment

55. What is meaning of “await”

Ans. “await” it’s a keyword used in Asynchronous Operation, await means it pause the execution of function until awaited operation is complete

56.what is difference between var and dynamic

Ans: when you use var you can not change it’s type after assigning, you Can change its value,

But when you use dynamic you can change it’s type and value also