FLUTTER /DART QUESTIONS AND ANSWER

What is flutter?

Originally developed by Google and now managed by ECMA, Flutter offers free and open-source UI development support. With one programming language and one codebase, users can create beautiful, natively compiled mobile applications with this UI toolkit. Flutter is not a language; it is an SDK. Flutter apps use the Dart programming language for creating an app. The first alpha version of Flutter was released in May 2017.

**What are the Flutter widgets?**

Flutter widgets are built using a modern framework that takes inspiration from [React](https://reactjs.org/). The central idea is that you build your UI out of widgets. Widgets describe what their view should look like given their current configuration and state. When a widget’s state changes, the widget rebuilds its description, which the framework diffs against the previous description to determine the minimal changes needed in the underlying render tree to transition from one state to the next.

**What is the Difference Between Stateless and Stateful Widget in Flutter?**

A **stateless** widget is useful when the part of the user interface you are describing does not depend on anything other than the configuration information and the BuildContext whereas a **Stateful** widget is useful when the part of the user interface you are describing can change dynamically.

A **Stateful** Widget is a mutable widget that is the reason it can be drawn multiple times within its lifetime.  
It is referred to as dynamic because it can change the inner data during the widget's lifetime. A widget that allows us to refresh the screen is called a Stateful widget. This widget does not have a build() method. It has the createState() method, which returns a class that extends the Flutters State Class. The examples of the Stateful widget are Checkbox, Radio, Slider, InkWell, Form, and TextField.

A **Stateless** widget will never rebuild by itself but can from external events.  
The Stateless widget does not have any state information. It remains static throughout its lifecycle. Examples of the Stateless widget are Text, Row, Column, Container, etc. If the screen or widget contains static content, it should be a Stateless widget, but if you want to change the content, it needs to be a Stateful widget.

**What is StatefulWidget LifeCycle?**

The lifecycle has the following simplified steps:

* **createState()**: When we build a new StatefulWidget, this one calls createState() right away and this override method must exist.
* **initState()**: It is the first method called after the Widget is created.This is our equivalent to onCreate() and viewDidLoad()
* **didChangeDependencies()** : This method is called immediately after initState() on the first time the widget is built.
* **build()** : This iscalled right after didChangeDependencies(). All the GUI is rendered here and will be called every single time the UI needs to be rendered.
* **didUpdateWidget()**: It’ll be called once the parent Widget did a change and needs to redraw the UI.
* **deactivate()**: Framework calls this method whenever it removes this State object from the tree
* **dispose()**: It is called when this object and its State are removed from the tree permanently and will never build again.

### Write the advantages of using flutter.

* **Reduce Code Development**: Flutter's hot reload feature allows it to offer faster performance. With it, the application gets compiled using the arm C/C++ library, making it closer to machine code and enabling it to run more quickly. The Flutter team has put lots of effort into providing a wide variety of ready-to-use widgets. Most of them are incredibly customizable, saving your time like no other framework before.
* **Cross-platform Development**: Using Flutter, you can write code, manage, and run it across multiple platforms. For the developers, this saves time, money, and effort.
* **Live and Hot Reloading**: This makes the app development process simpler and faster. Additionally, it also allows us to modify or update the code once a change is made.
* **Similar to Native App performance**: In contrast to most cross-platform frameworks, Flutter does not rely on intermediate code representations or interpretations. The Flutter application is built directly into the machine code, which eliminates any performance issues associated with the interpretation process. With Flutter, you get a fully compiled release application ahead of time.
* **Good Community Support**: Developers can ask questions about issues and get answers quickly.
* **Little/Minimal Code**: Each Flutter app is built using Dart programming language, which uses JIT and AOT compilation for faster startup time, faster performance, and smoother functionality. With the JIT feature, you can increase the speed of development and refresh the UI.
* **Documentation:**  Flutter's documentation is well-organized and informative. It serves as a central repository for all written documents.
* **Expressive and Flexible UI**: Flutter offers a customizable layered architecture that allows for highly customizable designs, expressive UIs, and fast rendering.