Miscellaneous Useful Interview Question For Android Kotlin - Rz Rasel

What is mean by data class in kotlin android?

Data class is a simple class which is used to hold data/state and contains standard functionality. A data keyword is used to declare a class as a data class.

Why use data class in Kotlin?

A Kotlin Data Class is used to hold the data only and it does not provide any other functionality apart from holding data.

What is the difference between data class and class Kotlin?

A data class is a class that only contains state and does not perform any operation. The advantage of using data classes instead of regular classes is that Kotlin gives us an immense amount of self-generated code.

What are the two classes of data?

Data can be classified as qualitative and quantitative.

What are getters and setters in Kotlin data class?

In Kotlin, setter is used to set the value of any variable and getter is used to get the value. Getters and Setters are auto-generated in the code. Let's define a property 'name', in a class, 'Company'. The data type of 'name' is String and we shall initialize it with some default value.

What is the difference between VAR and VAL in Kotlin data class?

val is a keyword in Kotlin that allows us to define properties as read-only. Because they are read-only, they cannot be modified. var , on the other hand, is a keyword that can be used to declare properties in Kotlin that are mutable. These properties are not read-only (as with val) and they can be modified at will.

What is the difference between data class and POJO in Kotlin?

A data class is specified by the keyword data when declaring the class definition in Kotlin, it is like defining a pojo class in Java. The difference is that Kotlin will take care of all these getter and setter as well as equals and hashCode method for you.

What is a data object vs class?

Class is a user-defined datatype that has its own data members and member functions whereas an object is an instance of class by which we can access the data members and member functions of the class.

Can data class be open Kotlin?

To ensure consistency and meaningful behavior of the generated code, data classes have to fulfill the following requirements: The primary constructor needs to have at least one parameter. All primary constructor parameters need to be marked as val or var . Data classes cannot be abstract, open, sealed, or inner.

What is the singleton class in Android?

The Singleton Pattern is a software design pattern that restricts the instantiation of a class to just “one” instance. It is used in Android Applications when an item needs to be created just once and used across the board.

What is singleton class in Kotlin with example?

A singleton class is a class that is defined in such a way that only one instance of the class can be created and used everywhere. It is used where we need only one instance of the class like NetworkService, DatabaseService, etc.

How do you define a singleton in Kotlin?

Using Singletons in Kotlin. By using the keyword object in your app, you're defining a singleton. A singleton is a design pattern in which a given class has only one single instance inside the entire app.

How to create a singleton class in Kotlin Android?

Rules for making Singleton Class:
1) A private constructor.
2) A static reference of its class.
3) One static method.
4) Globally accessible object reference.
5) Consistency across multiple threads.

Why use singleton class in Android?

A singleton is a design pattern that restricts the instantiation of a class to only one instance. Notable uses include controlling concurrency and creating a central point of access for an application to access its data store.

What is the purpose of singleton class?

The Singleton's purpose is to control object creation, limiting the number to one but allowing the flexibility to create more objects if the situation changes. Since there is only one Singleton instance, any instance fields of a Singleton will occur only once per class, just like static fields.

What is the difference between object and singleton in Kotlin?

In Kotlin, the singleton pattern is used as a replacement for static members and fields that don't exist in that programming language. A singleton is created by simply declaring an object . Contrary to a class , an object can't have any constructor, but init blocks are allowed if some initialization code is needed.

What is the difference between singleton and object Kotlin?

The singleton pattern restricts the instantiation of a class to a single object. It is useful in cases wherein you only need a single object to contain a global state. An object in Kotlin (among its many purposes) is both a class definition and an instantiation of single instance combined.

What are companion objects?

A companion object is an object that's declared in the same file as a class , and has the same name as the class. A companion object and its class can access each other's private members. A companion object's apply method lets you create new instances of a class without using the new keyword.

Why use companion objects?

Companion objects are beneficial for encapsulating things and they act as a bridge for writing functional and object oriented programming code. Using companion objects, the Scala programming code can be kept more concise as the static keyword need not be added to each and every attribute.

What is the difference between companion and object in Kotlin?

Companion objects are essentially the same as a standard object definition, only with a couple of additional features to make development easier. Companion objects allow their members to be accessed from inside the companion class without specifying the name.

What is the difference between object and companion object?

If we need to provide the Singleton behavior, then we are better off with Objects , else if we just want to add some static essence to our classes, we can use Companion objects .

What is the benefit of companion object Kotlin?

So why use companion objects? Because they provide a convenient shorthand for accessing “static” properties/functions. That's it. If, for some reason, you absolutely must have lazy initialization, then just use a regular object instead.

What is difference between singleton object and companion object?

In scala, when you have a class with same name as singleton object, it is called companion class and the singleton object is called companion object. The companion class and its companion object both must be defined in the same source file.

When should I use companion Kotlin?

Companion object is not a Singleton object or Pattern in Kotlin – It's primarily used to define class level variables and methods called static variables. This is common across all instances of the class.

What is Coroutines in Kotlin?

A coroutine is a concurrency design pattern that you can use on Android to simplify code that executes asynchronously. Coroutines were added to Kotlin in version 1.3 and are based on established concepts from other languages.

What's the difference between == and === operators in Kotlin?

Equality In Kotlin there are two types of equality: Structural equality (== - a check for equals()) Referential equality (=== - two references point to the same object)

What's the difference between lazy and Lateinit?

lateinit can only be used with a var property whereas lazy will always be used with val property. A lateinit property can be reinitialised again and again as per the use whereas the lazy property can only be initialised once.

How do I use companion object in Kotlin?

To create a companion object, you need to add the companion keyword in front of the object declaration. The output of the above code is " You are calling me :) " This is all about the companion object in Kotlin. Hope you liked the blog and will use the concept of companion in your Android application.

Why Kotlin removed static?

Be advised: Kotlin has removed Java-style statics to encourage more maintainable (dare I say 'better') coding practices. Static globals are generally against the OOP-paradigm but they can be quite convenient. Hence, Kotlin has provided us with companions, a more OOP-friendly way of having statics.

Can we create 2 object of singleton class?

The Singleton's purpose is to control object creation, limiting the number to one but allowing the flexibility to create more objects if the situation changes. Since there is only one Singleton instance, any instance fields of a Singleton will occur only once per class, just like static fields.

Is Encapsulation a singleton?

A Singleton doesn't bring any encapsulation to the table that a class doesn't already have. If a class has poor encapsulation, it will still have poor encapsulation after it is turned into a Singleton. A Singleton restricts a class to a single instance, and that's all that it does.

What is lazy in Kotlin?

lazy in Kotlin is useful in a scenario when we want to create an object inside a class, but that object creation is expensive and that might lead to a delay in the creation of the object that is dependent on that expensive object.

Companion Object in Interface?

The companion object provides a common constant. It defines the minimum amount of wheels in a double-track vehicle. Additionally, we provide a helper method isDoubleTrack(). Common constants and helper methods are common use cases for using companion objects in interfaces.

How many types of coroutines are in Kotlin?

Dispatchers: Dispatchers help coroutines in deciding the thread on which the work has to be done. There are majorly three types of Dispatchers which are IO, Default, and Main.

What is the difference between coroutines and threads?

Coroutines are lighter than threads . Since they stack less . I.e coroutines don't have a dedicated stack . It means coroutine suspend execution by returning to the caller and the data that is required to resume execution is stored separately from the stack.

What is mutable and immutable in Kotlin?

Mutable – The contents of the list can be freely changed. Read-Only – The contents of the collection are not meant to be changed. However, the underlying data can be changed. Immutable – Nothing can change the contents of the collection.

Which is thread safe late init or lazy?

The Lazy initialization is thread-safe. Lateinit can only be used with var. Lazy initialization is used with the val property.

What is difference between Val and const in Kotlin?

The variables declared as const are initialized at the runtime. val deals with the immutable property of a class, that is, only read-only variables can be declared using val. val is initialized at the runtime. For val, the contents can be muted, whereas for

const val, the contents cannot be muted.

What is scope function in Kotlin?

In Kotlin, there are five types of scope functions: let, with, run, apply, and also. As an Android mobile developer, you must understand this critical concept. The beauty of Kotlin emerges from its unique features, which make it suitable for both frontend and backend development.

What is reflection in Kotlin?

In Kotlin, Reflection is a combination of language and library capabilities that allow you to introspect a program while it's running. Kotlin reflection is used at runtime to utilize a class and its members, such as properties, methods, and constructors.

Is Kotlin static or dynamic?

Statically typed language.
Being a statically typed language, Kotlin still has to interoperate with untyped or loosely typed environments, such as the JavaScript ecosystem.

Why all classes are final in Kotlin?

All Kotlin classes are final, so they cannot be inherited. To make a class inheritable, the keyword open needs to be present at the beginning of the class signature, which also makes them non-final.

Why singleton is stateless?

In these terms, singleton - scoped beans are usually stateless, because they are used by multiple clients simultaneosly and their states are not client-specific.

What are the three types of encapsulation?

There are three basic techniques to encapsulate data in Object Programming.
Types of Encapsulation in OOPs
Member Variable Encapsulation.
Function Encapsulation.
Class Encapsulation.

What is zip in Kotlin?

Returns a sequence of values built from the elements of this sequence and the other sequence with the same index. The resulting sequence ends as soon as the shortest input sequence ends. The operation is intermediate and stateless.

Are coroutines faster than threads?

You can compare the time taken to create many threads to the time taken to create many coroutines. This provides a practical way to compare coroutines to threads in the context of the often-quoted statement that coroutines are faster to create than threads.

Do coroutines run every frame?

Coroutines are updated every frame. Your loop is a counter loop. If each frame takes more time, then your loop will take longer to complete.

What is the purpose of coroutines?

A coroutine is a concurrency design pattern that you can use on Android to simplify code that executes asynchronously. Coroutines were added to Kotlin in version 1.3 and are based on established concepts from other languages.

Which thread is faster user or kernel?

User-level threads are easier and faster to create than kernel-level threads. They can also be more easily managed. User-level threads can be run on any operating system. There are no kernel mode privileges required for thread switching in user-level threads.

How many coroutines a thread can have?

A thread can only execute one coroutine at a time, so the framework is in charge of moving coroutines between threads as necessary.

What is a sealed class in Kotlin?

Sealed classes are used for representing restricted class hierarchies, when a value can have one of the types from a limited set, but cannot have any other type.

How do I use synchronized in Kotlin?

There are two types of synchronization available in Java (and Kotlin). Synchronized methods and synchronized statements. To use synchronized methods, the method needs to be denoted by the synchronized function modifier keyword in Java or @ Synchronized annotation in Kotlin.

Is String immutable in Kotlin?

Strings are immutable which means the length and elements cannot be changed after their creation. Unlike Java, Kotlin does not require a new keyword to instantiate an object of a String class.

What is null safety Kotlin?

Kotlin null safety is a procedure to eliminate the risk of null reference from the code.

Kotlin compiler throws NullPointerException immediately if it found any null argument is passed without executing any other statements. Kotlin's type system is aimed to eliminate NullPointerException form the code.

What is the difference between list and Mutablelist in Kotlin?

In Kotlin, a list is an ordered collection of elements. Immutable lists and mutable lists are the two types of lists in Kotlin. Immutable Lists can not be modified, but mutable lists are editable. Immutable Lists are read-only lists and are created with the listOf() method.