1. Kotlin Basics

1. **What is Kotlin?**
   Kotlin is a statically-typed programming language that runs on the JVM, Android, and JavaScript. It is fully interoperable with Java.
2. **What are the key features of Kotlin?**
   - Concise syntax
   - Null safety
   - Extension functions
   - Coroutines for asynchronous programming
   - Data classes
   - Smart casts
3. **What is the difference between val and var?**
   - val is immutable (read-only).
   - var is mutable (can be reassigned).
4. **What is a nullable type in Kotlin?**
   A type that can hold either a value or null. Declared with ?, e.g., String?.
5. **How do you handle null safety in Kotlin?**
   - Use safe calls (?.).
   - Elvis operator (?:).
   - Non-null assertion (!!).
   - Safe casts (as?).
6. **What are data classes in Kotlin?**
   Classes specifically designed to hold data. They automatically generate equals(), hashCode(), toString(), and copy() methods.
7. **What are extension functions?**
   Functions that extend a class without modifying its source code. Example:
   kotlin

   Copy

   fun String.isPalindrome(): Boolean { return this == this.reversed() }
8. **What are higher-order functions?**
   Functions that take other functions as parameters or return them. Example:
   kotlin

   Copy

   fun operateOnNumbers(a: Int, b: Int, operation: (Int, Int) -> Int): Int { return operation(a, b) }
9. **What are lambdas in Kotlin?**
   Anonymous functions that can be passed as arguments. Example:
   Kotlin val sum = { a: Int, b: Int -> a + b }

10. **What are coroutines?**
    Lightweight threads for asynchronous programming. They allow non-blocking code execution.

---

2. Android Basics

11. **What is Android?**
    Android is an open-source operating system based on Linux, designed for mobile devices.
12. **What are the key components of Android?**
    - Activities
    - Services
    - Broadcast Receivers
    - Content Providers
13. **What is an Activity?**
    A single screen in an Android app with a user interface.
14. **What is a Fragment?**
    A reusable portion of the UI that can be embedded in an Activity.
15. **What is the Android Manifest file?**
    An XML file that describes essential information about the app, such as its components, permissions, and hardware requirements.
16. **What is Intent in Android?**
    A messaging object used to request an action from another app component.
17. **What is the difference between startActivity() and startActivityForResult()?**
    - startActivity() launches a new activity.
    - startActivityForResult() launches an activity and expects a result back.
18. **What is a Service in Android?**
    A component that runs in the background to perform long-running operations.
19. **What is a BroadcastReceiver?**
    A component that responds to system-wide broadcast announcements.
20. **What is SharedPreferences?**
    A lightweight storage mechanism to store key-value pairs persistently.

---

3. Kotlin Advanced

21. **What are sealed classes?**
    Classes that restrict hierarchies, allowing only a fixed set of subclasses.
22. **What is the difference between lateinit and by lazy?**
    - lateinit is used for mutable properties initialized later.
    - by lazy is used for immutable properties initialized lazily.

23. **What is a companion object?**
    A singleton object tied to a class, similar to static members in Java.
24. **What is inline function in Kotlin?**
    A function that reduces overhead by inlining the function code at the call site.
25. **What are reified type parameters?**
    A way to access type information at runtime in inline functions.
26. **What is delegation in Kotlin?**
    A design pattern where an object handles a request by delegating to a second object.
27. **What is the difference between List and MutableList?**
    - List is immutable.
    - MutableList is mutable.
28. **What is the purpose of the let function?**
    To execute a block of code on a non-null object.
29. **What is the purpose of the apply function?**
    To configure an object and return it.
30. **What is the purpose of the also function?**
    To perform additional actions on an object and return it.

---

4. Android Advanced

31. **What is ViewModel?**
    A class designed to store and manage UI-related data in a lifecycle-conscious way.
32. **What is LiveData?**
    An observable data holder that respects the lifecycle of Android components.
33. **What is Room in Android?**
    A persistence library that provides an abstraction layer over SQLite.
34. **What is Dependency Injection?**
    A design pattern where objects are passed their dependencies rather than creating them.
35. **What is Dagger/Hilt?**
    A dependency injection framework for Android.
36. **What is WorkManager?**
    A library for managing deferrable and guaranteed background tasks.
37. **What is Navigation Component?**
    A framework for navigating between destinations in an app.
38. **What is Data Binding?**
    A library that binds UI components to data sources directly in XML.
39. **What is ViewBinding?**
    A feature that allows you to interact with views more easily and safely.
40. **What is the difference between onCreate() and onStart()?**
    - onCreate() is called when the activity is first created.

- onStart() is called when the activity becomes visible.

---

5. Kotlin Coroutines

41. **What are coroutine scopes?**
    Define the lifecycle of coroutines, such as GlobalScope, lifecycleScope, and viewModelScope.
42. **What is a coroutine dispatcher?**
    Determines the thread on which a coroutine runs.
    Examples: Dispatchers.Main, Dispatchers.IO.
43. **What is suspend in Kotlin?**
    A keyword used to mark a function as a coroutine.
44. **What is the difference between launch and async?**
    - launch is used for fire-and-forget tasks.
    - async is used for tasks that return a result.
45. **What is withContext?**
    A function to switch the context of a coroutine.

---

6. Android Architecture

46. **What is MVVM?**
    Model-View-ViewModel, an architectural pattern for separating UI logic from business logic.
47. **What is MVP?**
    Model-View-Presenter, an architectural pattern where the presenter acts as a middleman between the view and the model.
48. **What is Clean Architecture?**
    A design philosophy that separates concerns into layers (e.g., presentation, domain, data).
49. **What is Repository Pattern?**
    A design pattern that abstracts data sources and provides a clean API for data access.
50. **What is Single Source of Truth (SSOT)?**
    A principle where data is stored in one place and accessed consistently.

---

7. Networking in Android

51. **What is Retrofit?**
    A type-safe HTTP client for Android.
52. **What is OkHttp?**
    An HTTP client for making network requests.
53. **What is Gson?**
    A library for serializing and deserializing JSON.

54. **What is Moshi?**
    A modern JSON library for Kotlin.
55. **What is Glide?**
    An image loading library for Android.

---

8. Testing in Android

56. **What is JUnit?**
    A unit testing framework for Java and Kotlin.
57. **What is Mockito?**
    A mocking framework for unit tests.
58. **What is Espresso?**
    A UI testing framework for Android.
59. **What is Robolectric?**
    A framework for running Android tests on the JVM.
60. **What is the difference between unit tests and instrumented tests?**
    - Unit tests run on the JVM.
    - Instrumented tests run on an Android device or emulator.

---

9. Performance Optimization

61. **What is ProGuard?**
    A tool for code shrinking and obfuscation.
62. **What is R8?**
    A replacement for ProGuard with better performance.
63. **What is memory leak in Android?**
    When an object is no longer needed but still referenced, preventing garbage collection.
64. **What is LeakCanary?**
    A library for detecting memory leaks in Android apps.
65. **What is ANR?**
    Application Not Responding, a dialog shown when the app is unresponsive.

---

10. Security in Android

66. **What is SSL/TLS?**
    Protocols for securing network communication.
67. **What is HTTPS?**
    HTTP over SSL/TLS for secure communication.
68. **What is Keystore?**
    A system for storing cryptographic keys.
69. **What is biometric authentication?**
    Authentication using fingerprints or facial recognition.

70. **What is ProGuard/R8 obfuscation?**
    A technique to make reverse engineering harder.

---

11. Miscellaneous

71. **What is Gradle?**
    A build automation tool for Android.
72. **What is KAPT?**
    Kotlin Annotation Processing Tool.
73. **What is KTS?**
    Kotlin Script, a way to write Gradle build scripts in Kotlin.
74. **What is Jetpack Compose?**
    A modern UI toolkit for building native Android UIs.
75. **What is Flutter?**
    A cross-platform UI toolkit by Google.

12. Kotlin Advanced (Continued)

76. **What is the difference between const and val?**
    - const is used for compile-time constants.
    - val is used for runtime constants.
77. **What is a destructuring declaration?**
    A way to unpack values from objects into variables. Example:
    kotlin

    Copy

    ```
    val (name, age) = Person("John", 25)
    ```
78. **What is the when expression?**
    A replacement for switch in Java, used for conditional branching.
79. **What is the difference between == and ===?**
    - == checks for structural equality.
    - === checks for referential equality.
80. **What is a type alias?**
    A way to give an existing type a new name. Example:
    kotlin

    Copy

    ```
    typealias Name = String
    ```
81. **What is the @JvmStatic annotation?**
    Used to expose a Kotlin function as a static method in Java.
82. **What is the @JvmOverloads annotation?**
    Generates overloaded methods for Kotlin functions with default parameters.

83. **What is the @JvmField annotation?**
    Exposes a Kotlin property as a public field in Java.
84. **What is the @Throws annotation?**
    Used to specify checked exceptions that a Kotlin function can throw.
85. **What is the @file:JvmName annotation?**
    Specifies the name of the generated Java class for a Kotlin file.
86. **What is the @Experimental annotation?**
    Marks APIs that are experimental and may change in the future.
87. **What is the @Deprecated annotation?**
    Marks a function or class as deprecated.
88. **What is the @Repeatable annotation?**
    Allows an annotation to be applied multiple times to the same target.
89. **What is the @Target annotation?**
    Specifies the possible targets for an annotation.
90. **What is the @Retention annotation?**
    Specifies how long an annotation is retained (e.g., source, binary, runtime).
91. **What is the @MustBeDocumented annotation?**
    Indicates that an annotation should be included in the generated documentation.
92. **What is the @Suppress annotation?**
    Suppresses specific compiler warnings.
93. **What is the @Transient annotation?**
    Marks a property as non-persistent in serialization.
94. **What is the @Volatile annotation?**
    Marks a property as volatile, ensuring visibility across threads.
95. **What is the @Synchronized annotation?**
    Ensures that only one thread can execute a method at a time.
96. **What is the @Strictfp annotation?**
    Ensures floating-point calculations are platform-independent.
97. **What is the @Native annotation?**
    Marks a function as native, implemented in platform-specific code.
98. **What is the @InlineOnly annotation?**
    Ensures that a function is inlined at the call site.
99. **What is the @HiddenDeclaration annotation?**
    Hides a declaration from the public API.
100. **What is the @DslMarker annotation?**
    Restricts the scope of DSL (Domain-Specific Language) functions.

---

13. Android Advanced (Continued)

101. **What is the difference between Activity and Fragment?**
    - Activity represents a single screen.
    - Fragment represents a portion of the UI.

102. **What is the FragmentManager?**
A class for managing fragments in an activity.
103. **What is the FragmentTransaction?**
A set of changes to be performed on fragments.
104. **What is the Lifecycle class?**
A class that holds the lifecycle state of an Android component.
105. **What is the LifecycleObserver?**
An interface for observing lifecycle events.
106. **What is the LifecycleOwner?**
An interface for components that have a lifecycle (e.g., Activity, Fragment).
107. **What is the ViewModelProvider?**
A utility class for creating ViewModel instances.
108. **What is the SavedStateHandle?**
A wrapper for saving and restoring state in a ViewModel.
109. **What is the Navigation Component?**
A framework for navigating between destinations in an app.
110. **What is the NavController?**
A class for managing navigation within a NavHost.
111. **What is the NavGraph?**
A collection of destinations and actions in a navigation flow.
112. **What is the NavHost?**
A container for displaying destinations in a navigation graph.
113. **What is the Paging Library?**
A library for loading and displaying large datasets in chunks.
114. **What is the PagingSource?**
A class for loading data in the Paging Library.
115. **What is the PagingData?**
A container for paginated data in the Paging Library.
116. **What is the RemoteMediator?**
A class for handling pagination with remote data sources.
117. **What is the Room Database?**
A persistence library for SQLite databases.
118. **What is the DAO?**
Data Access Object, an interface for accessing data in Room.
119. **What is the Entity?**
A class representing a table in Room.
120. **What is the Database?**
A class representing the database in Room.
121. **What is the Migration?**
A class for handling database schema changes in Room.
122. **What is the TypeConverter?**
A class for converting custom types in Room.
123. **What is the WorkManager?**
A library for managing background tasks.

124. **What is the Worker?**
A class for performing background work in WorkManager.
125. **What is the WorkRequest?**
A request for performing work in WorkManager.
126. **What is the Constraints?**
Conditions under which a WorkManager task should run.
127. **What is the Data?**
A class for passing input/output data in WorkManager.
128. **What is the Result?**
The outcome of a WorkManager task.
129. **What is the PeriodicWorkRequest?**
A WorkManager request that repeats at intervals.
130. **What is the OneTimeWorkRequest?**
A WorkManager request that runs once.
131. **What is the Chaining?**
Linking multiple WorkManager tasks in sequence.
132. **What is the BroadcastReceiver?**
A component for receiving system-wide broadcasts.
133. **What is the LocalBroadcastManager?**
A class for sending and receiving broadcasts within an app.
134. **What is the IntentFilter?**
A filter for specifying which intents a component can handle.
135. **What is the PendingIntent?**
A token for delegating an intent to another app.
136. **What is the ContentProvider?**
A component for sharing data between apps.
137. **What is the ContentResolver?**
A class for accessing data from a ContentProvider.
138. **What is the Cursor?**
A class for accessing query results from a ContentProvider.
139. **What is the Loader?**
A class for loading data asynchronously.
140. **What is the AsyncTaskLoader?**
A Loader that performs tasks on a background thread.
141. **What is the CursorLoader?**
A Loader for querying ContentProvider data.
142. **What is the JobScheduler?**
A system service for scheduling background tasks.
143. **What is the AlarmManager?**
A system service for scheduling alarms.
144. **What is the NotificationManager?**
A system service for managing notifications.
145. **What is the NotificationChannel?**
A channel for grouping notifications in Android 8.0+.
146. **What is the NotificationCompat?**
A compatibility library for creating notifications.

147. **What is the MediaPlayer?**
A class for playing audio and video.
148. **What is the ExoPlayer?**
A modern media player library.
149. **What is the CameraX?**
A library for working with the camera in Android.
150. **What is the Camera2?**
A low-level API for working with the camera in Android.

---

14. Kotlin Coroutines (Continued)

151. **What is GlobalScope?**
A coroutine scope that lives for the entire application lifecycle.
152. **What is lifecycleScope?**
A coroutine scope tied to the lifecycle of an Android component (e.g., Activity, Fragment).
153. **What is viewModelScope?**
A coroutine scope tied to the lifecycle of a ViewModel.
154. **What is Dispatchers.Main?**
A coroutine dispatcher for running tasks on the main/UI thread.
155. **What is Dispatchers.IO?**
A coroutine dispatcher for running I/O-bound tasks.
156. **What is Dispatchers.Default?**
A coroutine dispatcher for CPU-intensive tasks.
157. **What is Dispatchers.Unconfined?**
A coroutine dispatcher that does not confine execution to a specific thread.
158. **What is Job in coroutines?**
A cancellable task that represents a coroutine.
159. **What is Deferred in coroutines?**
A Job with a result that can be awaited.
160. **What is SupervisorJob?**
A Job that does not cancel child coroutines when one fails.
161. **What is CoroutineExceptionHandler?**
A handler for catching uncaught exceptions in coroutines.
162. **What is runBlocking?**
A coroutine builder that blocks the current thread until the coroutine completes.
163. **What is coroutineScope?**
A coroutine builder that creates a new scope without blocking the current thread.
164. **What is supervisorScope?**
A coroutine builder that creates a new scope with a SupervisorJob.
165. **What is async?**
A coroutine builder that returns a Deferred result.

166. **What is launch?**
A coroutine builder that starts a new coroutine without returning a result.
167. **What is withTimeout?**
A function that cancels a coroutine if it exceeds a specified time limit.
168. **What is withTimeoutOrNull?**
A function that returns null if a coroutine exceeds a specified time limit.
169. **What is flow in Kotlin?**
A cold stream of values that can be collected asynchronously.
170. **What is StateFlow?**
A Flow that emits the current and new state updates.
171. **What is SharedFlow?**
A Flow that emits values to multiple collectors.
172. **What is channel in coroutines?**
A hot stream of values that can be sent and received asynchronously.
173. **What is produce?**
A coroutine builder that creates a ReceiveChannel.
174. **What is consumeEach?**
A function for consuming values from a ReceiveChannel.
175. **What is select?**
A function for waiting on multiple suspending functions simultaneously.

15. Android Architecture (Continued)

176. **What is the Repository pattern?**
A design pattern that abstracts data sources and provides a clean API for data access.
177. **What is the UseCase pattern?**
A design pattern that encapsulates business logic in a reusable component.
178. **What is the DataBinding library?**
A library that binds UI components to data sources directly in XML.
179. **What is the ViewBinding library?**
A feature that allows you to interact with views more easily and safely.
180. **What is the LiveData transformation?**
A way to transform LiveData values using map or switchMap.
181. **What is the MediatorLiveData?**
A LiveData subclass that can observe other LiveData sources.
182. **What is the SavedStateHandle?**
A wrapper for saving and restoring state in a ViewModel.
183. **What is the Navigation component?**
A framework for navigating between destinations in an app.
184. **What is the NavController?**
A class for managing navigation within a NavHost.
185. **What is the NavGraph?**
A collection of destinations and actions in a navigation flow.

186. **What is the NavHost?**
A container for displaying destinations in a navigation graph.
187. **What is the Paging library?**
A library for loading and displaying large datasets in chunks.
188. **What is the PagingSource?**
A class for loading data in the Paging library.
189. **What is the PagingData?**
A container for paginated data in the Paging library.
190. **What is the RemoteMediator?**
A class for handling pagination with remote data sources.
191. **What is the Room database?**
A persistence library for SQLite databases.
192. **What is the DAO?**
Data Access Object, an interface for accessing data in Room.
193. **What is the Entity?**
A class representing a table in Room.
194. **What is the Database?**
A class representing the database in Room.
195. **What is the Migration?**
A class for handling database schema changes in Room.
196. **What is the TypeConverter?**
A class for converting custom types in Room.
197. **What is the WorkManager?**
A library for managing background tasks.
198. **What is the Worker?**
A class for performing background work in WorkManager.
199. **What is the WorkRequest?**
A request for performing work in WorkManager.
200. **What is the Constraints?**
Conditions under which a WorkManager task should run.

---

16. Networking in Android (Continued)

201. **What is Retrofit?**
A type-safe HTTP client for Android.
202. **What is OkHttp?**
An HTTP client for making network requests.
203. **What is Gson?**
A library for serializing and deserializing JSON.
204. **What is Moshi?**
A modern JSON library for Kotlin.
205. **What is Glide?**
An image loading library for Android.
206. **What is Picasso?**
An image loading library for Android.

207. **What is Coil?**
A modern image loading library for Android.
208. **What is Volley?**
A networking library for Android.
209. **What is HttpURLConnection?**
A low-level API for making HTTP requests.
210. **What is WebSocket?**
A protocol for real-time communication between a client and server.
211. **What is Socket.IO?**
A library for real-time, bidirectional communication.
212. **What is Firebase Realtime Database?**
A cloud-hosted NoSQL database for real-time data syncing.
213. **What is Firebase Firestore?**
A cloud-hosted NoSQL database with advanced querying capabilities.
214. **What is Firebase Cloud Messaging?**
A service for sending push notifications to Android devices.
215. **What is Firebase Authentication?**
A service for authenticating users using email, phone, or third-party providers.
216. **What is Firebase Crashlytics?**
A service for tracking and analyzing app crashes.
217. **What is Firebase Performance Monitoring?**
A service for monitoring app performance.
218. **What is Firebase Remote Config?**
A service for customizing app behavior without deploying a new version.
219. **What is Firebase A/B Testing?**
A service for running experiments on app features.
220. **What is Firebase Test Lab?**
A service for testing apps on real devices in the cloud.

---

17. Testing in Android (Continued)

221. **What is JUnit?**
A unit testing framework for Java and Kotlin.
222. **What is Mockito?**
A mocking framework for unit tests.
223. **What is Espresso?**
A UI testing framework for Android.
224. **What is Robolectric?**
A framework for running Android tests on the JVM.
225. **What is the difference between unit tests and instrumented tests?**
    - Unit tests run on the JVM.
    - Instrumented tests run on an Android device or emulator.

226. **What is UI Automator?**

A testing framework for cross-app UI testing.

227. **What is Barista?**

A library for simplifying UI tests in Android.