Kotlin Coroutines Cheat Sheet

What Are Coroutines?

- Lightweight threads managed by Kotlin runtime.
- Designed for asynchronous, non-blocking code.
- Suspendable using `suspend` functions.

Coroutine Builders

- launch: fire-and-forget (returns Job)
- async: returns Deferred<T> for results
- runBlocking: blocks the current thread (used in tests)

Dispatchers

- Main: For UI (Android)
- IO: For disk and network IO
- Default: For CPU-intensive work
- Unconfined: Executes in caller's context

Structured Concurrency

- coroutineScope: Cancels all children if one fails
- supervisorScope: Children are independent

Exception Handling

- Use try-catch inside coroutine.
- CoroutineExceptionHandler for uncaught exceptions (only in launch).

Cancellation

- Use isActive, yield(), ensureActive() for cooperative cancellation.

Coroutines in Android

Kotlin Coroutines Cheat Sheet

- viewModelScope: For launching in ViewModel.
- lifecycleScope: For collecting flows in Activity/Fragment.

Flow Basics

- Cold asynchronous data stream.
- Use flow { emit(...) } and collect { ... }

Side Effects in Compose

- SideEffect { ... } for calling non-Composable functions.

Common Interview Questions

- Difference between launch and async?
- How does structured concurrency work?
- How do you cancel a coroutine?
- How do you handle exceptions?

Sample Coding Qs

1. Parallel calls: Use async inside coroutineScope.
2. Retry with timeout: Combine retry logic with withTimeout.