

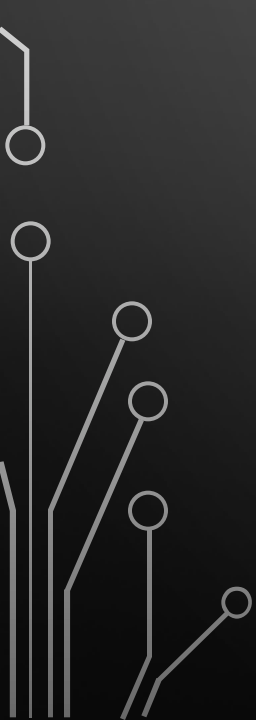

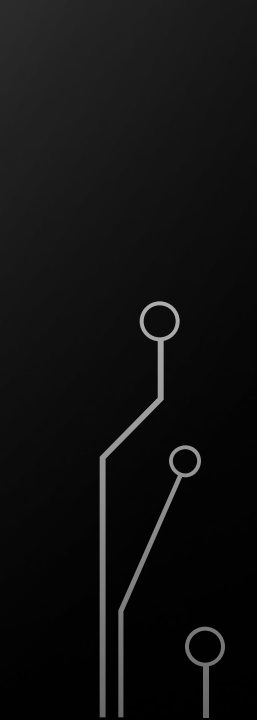
INTRODUCTION TO COROUTINES IN KOTLIN



Abhishek Bansal
Lead Android Developer
DoctorBox Health



AGENDA

- What are Coroutines
 - Coroutines and Threads
 - “suspend” modifier in Kotlin
 - Coroutine Builders
 - Structured Concurrency- Coroutine Scope
 - Demo
- 
- 
- 

WHAT ARE COROUTINES?


- Traditions first!

```
import kotlinx.coroutines.*  
  
fun main() {  
    GlobalScope.launch { // launch new coroutine in background and continue  
        delay(1000L) // non-blocking delay for 1 second (default time unit is ms)  
        println("World!") // print after delay  
    }  
    println("Hello,") // main thread continues while coroutine is delayed  
    Thread.sleep(2000L) // block main thread for 2 seconds to keep JVM alive  
}
```

Source: <https://kotlinlang.org/docs/reference/coroutines/basics.html>



WHAT ARE COROUTINES?

- First coined in 1958 by Melvin Conway
 - As per wikipedia
 - Coroutines are computer-program components that generalize *subroutines* for *non-preemptive multitasking*, by allowing multiple *entry points* for suspending and resuming execution at certain locations.
- 

WHAT ARE COROUTINES?

- A way of writing non-blocking asynchronous code
- Provide concurrency
- Allow passing control to each other while maintaining state
 - i.e. coroutine can return back to caller while maintaining its own state and allows caller to return back to it and resume its execution

COROUTINES AND THREADS

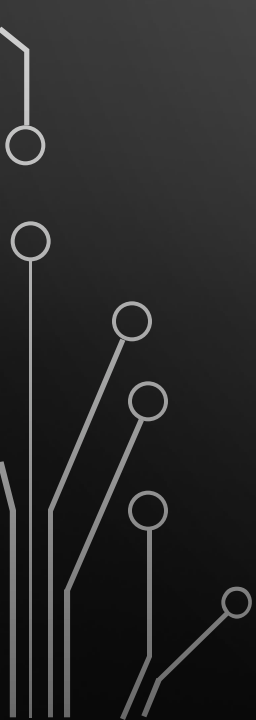
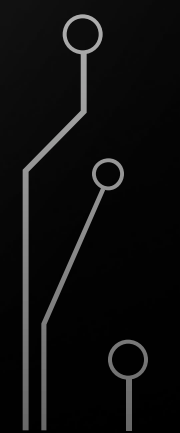
- Loosely speaking
 - Coroutines are essentially lightweight threads!

```
import kotlinx.coroutines.*  
fun main() = runBlocking { // blocks till all of the children coroutines finish execution  
    repeat(100_000) { // launch a lot of coroutines  
        launch {  
            delay(1000L)  
            print(".")  
        }  
    }  
}
```

Source: <https://kotlinlang.org/docs/reference/coroutines/basics.html>



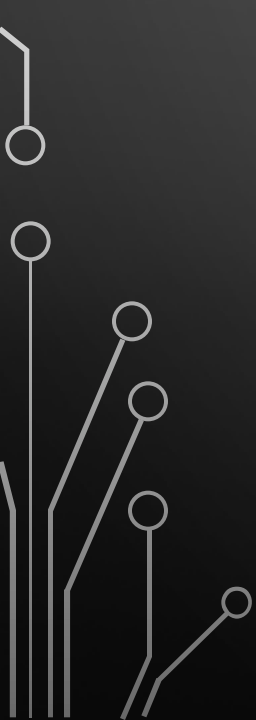
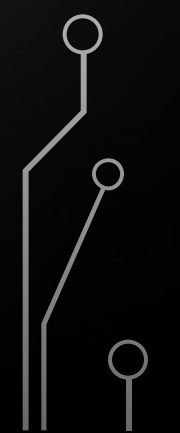
COROUTINES AND THREADS

- But Conceptually
 - Coroutines are not threads! (Not at all)
- 
- 



COROUTINES AND THREADS



- Coroutines provide concurrency but not parallelism
 - Scheduling is non-preemptive and is done by language/programmer
 - Super efficient
 - No context switching
 - No extra stack space
 - No synchronization needed
 - Threads are always global, coroutines are scoped
- 
- 

suspend MODIFIER

- suspend modifier allows a method to be launched in a coroutine
- Methods marked with suspend can only be called from a coroutine or a method marked as suspend
- Example
 - <https://medium.com/@elye.project/understanding-suspend-function-of-coroutines-de26b070c5ed>

COROUTINE BUILDERS

- Coroutine builders are functions that take suspending lambda as arguments create coroutines and then optionally return their result.
- Can be invoked from regular non suspending functions
- Available in **kotlinx.coroutines** by jetbrains (<https://github.com/kotlin/kotlinx.coroutines>)
- Example:
 - `launch{}` – executes a coroutine and returns a Job object
 - `Async{}`- executes a coroutine and returns `Deferred<T>` object, call `await()` to get T
 - `runBlocking{}`- blocks current thread till execution of wrapped block completes

COROUTINE SCOPE

- Each coroutine is scoped, unlike threads
- Can be global or user defined
- Each coroutine builder is an extension function on *CoroutineScope*
- *GlobalScope* can be used to launch coroutines in Application Scope
- Having well defined scope and lifecycle brings structure to concurrency provided by coroutines which is not possible for threads.

The background is a dark gray gradient. In the corners, there are white line-art illustrations of circuit boards or neural networks. These lines connect to small white circles, resembling nodes or components. The patterns are symmetrical, with more complex branching in the top-left and bottom-left corners, and simpler, more linear patterns in the top-right and bottom-right corners.

DEMO

<https://github.com/abhishekBansal/coroutine-mvvm-architecture-demo>

FURTHER READINGS/REFERENCES

- <https://kotlinlang.org/docs/reference/coroutines/basics.html>
- <https://en.wikipedia.org/wiki/Coroutine>
- <https://android.jlelse.eu/coroutines-basic-terminologies-and-usage-b4242bd1b2a4>
- <https://www.youtube.com/watch?v=a3agLJQ6vt8>
- https://www.youtube.com/watch?v=_hfBv0a09Jc
- Useful stackoverflow discussions
 - <https://stackoverflow.com/questions/1050222/what-is-the-difference-between-concurrency-and-parallelism>
 - <https://stackoverflow.com/questions/1934715/difference-between-a-coroutine-and-a-thread>
 - <https://stackoverflow.com/questions/553704/what-is-a-coroutine>
- <https://www.geeksforgeeks.org/coroutines-in-c-cpp/>



THANK YOU

QUESTIONS?

Any Feedback?

discover.ab@gmail.com

