





Imagina que estamos no Facebook e queremos fazer login e ver nossas postagens, então podemos usar uma função suspensa

Imagina que para recuperar os dados do usuário que esta logado, podemos demorar 2 segundos, mas existe também pessoas com conexões ruins que podem levar ate 10 segundos.

E não podemos ver as postagens sem antes pegar os dados do usuário.

Numa função suspensa ela vai suspender a função ate que recuperamos os dados do usuário, após recuperar os dados do usuário vamos avançar para a segunda linha de código, que é: ver as postagens do usuário.

Sem a função suspensa eu teria um problema, porque enquanto esta recuperando os dados do usuário ela tentaria já executar a segunda linha de código, sem saber de qual usuário ela estaria exibindo os dados.

RESUMINDO: Ele vai aguarda a primeira linha ser executada antes de executar a próxima, mesmo que demore 10 segundos.

# PRATICA:

binding.btnIniciar.setOnClickListener **{** *CoroutineScope*(Dispatchers.IO).*launch* **{** executar()  
 */\*\* O metodo executar ele vai pegar todos os dados e vai executar aqui  
 \* Ele vai pegar os dados de outros metodos  
 \* Vai recuperar o usuario que esta logado  
 \* Vai recuperar as postagens do usuario  
 \*/* **}  
 }** }  
  
  
 private suspend fun recuperarUsuarioLogado() : Usuario {  
 */\*\* Toda funçao suspensa ela só pode ser usada dentro de uma corrotina  
 \* Ou só pode ser usada dentro de outra funçao suspensa.  
 \*  
 \* Delay = Se você reparar ele também é uma suspend fun, repare o icone ao lado.  
 \* Se voce remover o suspend voce nao pode usar o metodo.  
 \*  
 \* Atençao aqui estamos simulando que vamos retornar um usuario.  
 \* Usuario = ID e Nome  
 \*  
 \* Aqui vamos retornar o nome e id do usuario.  
 \*/* delay(2000) //2s  
  
 return Usuario(1020, "Alleph")  
 }  
  
  
 private suspend fun recuperarPostagensPeloId( idUsuario : Int) : List<String>{  
 delay(2000)  
  
 return *listOf*(  
 "Viagem para Cabo Frio",  
 "Comprei um carro novo",  
 "Me formei...",  
 "Estudando Android",  
 "Indo dormir."  
 )  
 }  
  
  
 private suspend fun executar() {  
 */\*\* Nesse log vamos pegar o usuario que esta vindo da outra funçao e vamos exibir o nome do usuario  
 \*  
 \* Atençao, repara que para recuperar o usuarioLogado simulamos que ele iria demorar 2 segundo para acontecer isso.  
 \* Então ele vai aguardar recuperar esse usuario e apos isso que ele vai executar a proxima linha  
 \*/* val usuario = recuperarUsuarioLogado()  
 Log.i("info\_coroutine", "${usuario.usuario}: ${Thread.currentThread().*name*}")  
  
 */\*\* Atençao aqui também, o metodo recuperar postagens também demora 2 segundos  
 \* Reparar que se não usar o suspend fun ele não vai aguardar as postagens ser recuperada e vamos ter problemas  
 \* Imagina oque vamos exibir na tela se as postagens não for configuradas  
 \*  
 \* Imagina que recuperamos os dados e salvamos na nossa data class  
 \* Entao vamos pegar esse ID recuperado e vamos passar para recuperar também as postagens.  
 \*/* val postagens = recuperarPostagensPeloId(usuario.id)  
 Log.i("info\_coroutine", "Usuario: ${usuario.id} - T: ${Thread.currentThread().*name*}")  
 Log.i("info\_coroutine", "Postagens: ${postagens.size} - T: ${Thread.currentThread().*name*}")  
 }  
}