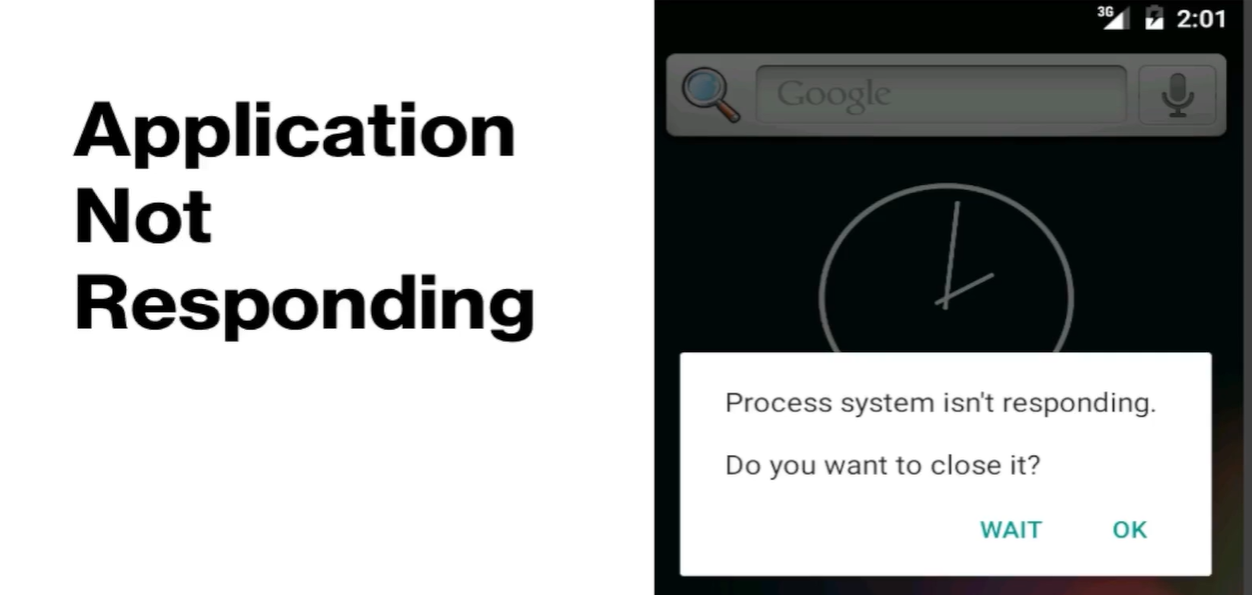
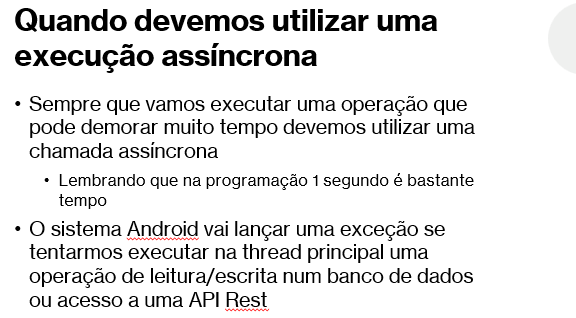
Placa com letras pretas

Descrição gerada automaticamente com confiança média

**• Aula 1: Programação assíncrona**



 Uma imagem contendo Linha do tempo

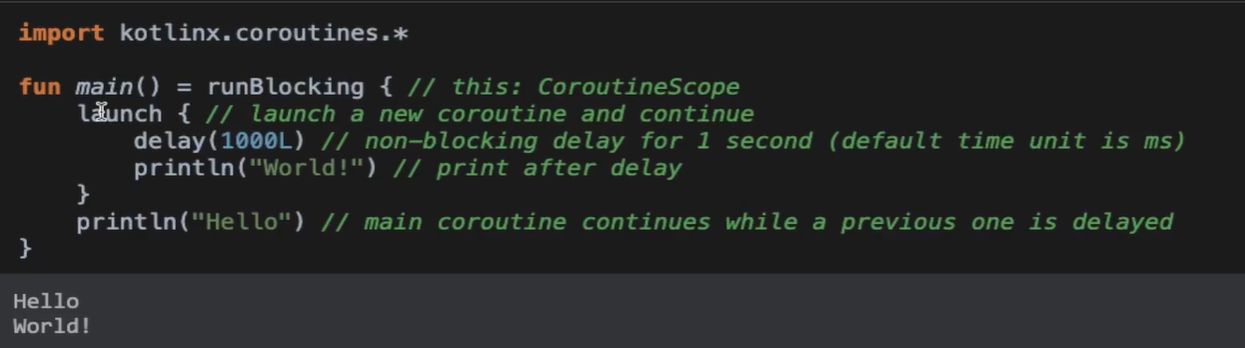
Descrição gerada automaticamente

**• Aula 2: Corroutine**

* Uma das grandes vantagens de utilizar Kotlin é a possibilidade de utilizar as coroutines
* Elas proporcionam um jeito simples e elegante de criar código assíncrono
* Utilizando esta biblioteca podemos escrever código de forma muito similar ao que faríamos de forma síncrona

**Vantagens das coroutines**Ícone

Descrição gerada automaticamente



[[https://kotlinlang.org/docs/coroutines-basics.html#your-first-coroutine](https://kotlinlang.org/docs/coroutines-basics.html)](https://kotlinlang.org/docs/coroutines-basics.html#your-first-coroutine)

**Funções suspensas**

* Quando utilizamos uma função suspensa, ela permite que suspendamos a execução daquele trecho do código enquanto a operação custosa está sendo executada **mas não bloquei a thread principal**
* Quando a operação custosa termina, ele volta a executar o código que estava suspenso do ponto aonde ele tinha parado
* Funções suspensas só podem ser chamadas por outras funções suspensas ou coroutines

Texto

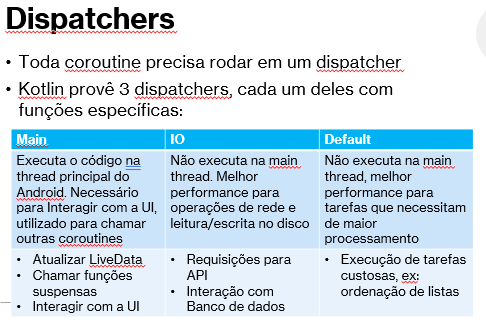
Descrição gerada automaticamente

**Criando uma coroutine**

* Para criar uma coroutine utilizamos duas palavras reservadas, **launch** e **async**. As coroutines criadas desta forma **não bloqueiam a thread principal**
* Utilizamos o **launch** em operações que **não tem retorno**
  + O tipo do retorno da função launch é **Job,** com ele podemos cancelar a execução da coroutine
* Paraoperações **aonde precisamos de um retorno** utilizamos **o async**
  + O tipo do retorno do async é o **Deferred,** através do método **await** conseguimos obter o retorno

**Texto

Descrição gerada automaticamente**  
**(.await)** -> Aguarda retorno  
Neste caso irá executar noReturn sem aguardar o println

****

[<https://medium.com/androiddevelopers/coroutines-on-android-part-i-getting-the-background-3e0e54d20bb>](https://medium.com/androiddevelopers/coroutines-on-android-part-i-getting-the-background-3e0e54d20bb)

**Texto

Descrição gerada automaticamente**

**Escopos**

Os escopos ajudam a definir o ciclo de vida de uma coroutine

* Existem vários escopos disponíveis, os principais são:
* GlobalScope: Escopo global vinculado ao ciclo de vida da aplicação
* ViewModelScope: Escopo vinculado a uma viewModel
* LiveDataScope: Vinculado ao ciclo de vida de um livedata
* MainScope: Vinculado ao ciclo de vida da UI

**Escopo x dispatcher**

* O escopo define **qual o ciclo de vida** que uma coroutine vai estar vinculada
* O dispatcher **define aonde** a coroutine vai rodar

**Retrofit + coroutine**

* Para utilizar coroutines com o retrofit basta transformar as funções na interface em **suspended** e alterar o tipo de retorno para **Response**

**Adicionando as dependências**

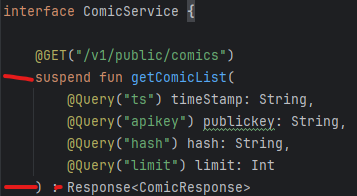
****

[<https://medium.com/android-dev-br/utilizando-kotlin-coroutines-no-android-c73fcda71e27>](https://medium.com/android-dev-br/utilizando-kotlin-coroutines-no-android-c73fcda71e27)[<https://medium.com/androiddevelopers/coroutines-on-android-part-i-getting-the-background-3e0e54d20bb>](https://medium.com/androiddevelopers/coroutines-on-android-part-i-getting-the-background-3e0e54d20bb)[<https://github.com/android/architecture-samples/blob/todo-mvvm-live-kotlin/todoapp/app/src/main/java/com/example/android/architecture/blueprints/todoapp/Event.kt>](https://github.com/android/architecture-samples/blob/todo-mvvm-live-kotlin/todoapp/app/src/main/java/com/example/android/architecture/blueprints/todoapp/Event.kt)

**Aula 3: Hands On - Parte 1**

**Modulo 20.3**

• Incluir a dependência:  


- [ComicService.kt] **corrotinas e Retrofit**

**SUSPENDE ->** Marcação de função como uma função de suspensão. Isso significa que essa função pode ser pausada e retomada posteriormente sem bloquear a thread principal da aplicação.

Execução em corrotinas: A função getComicList só pode ser chamada de dentro de outra função de suspensão ou de uma corrotina.

**RESPONSE ->** encapsula a resposta da requisição HTTP, permitindo que você acesse o código de status, cabeçalhos e corpo da resposta.