

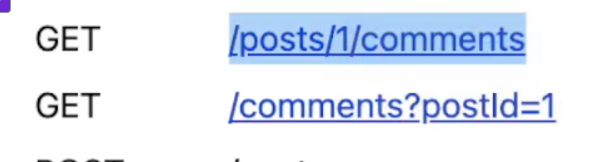
Se não passar os parâmetros vamos ter uma URL inválida.

Mas vai ter situações que nem sempre vamos passar os parâmetros, e ai vamos utilizar requisições GET com Query

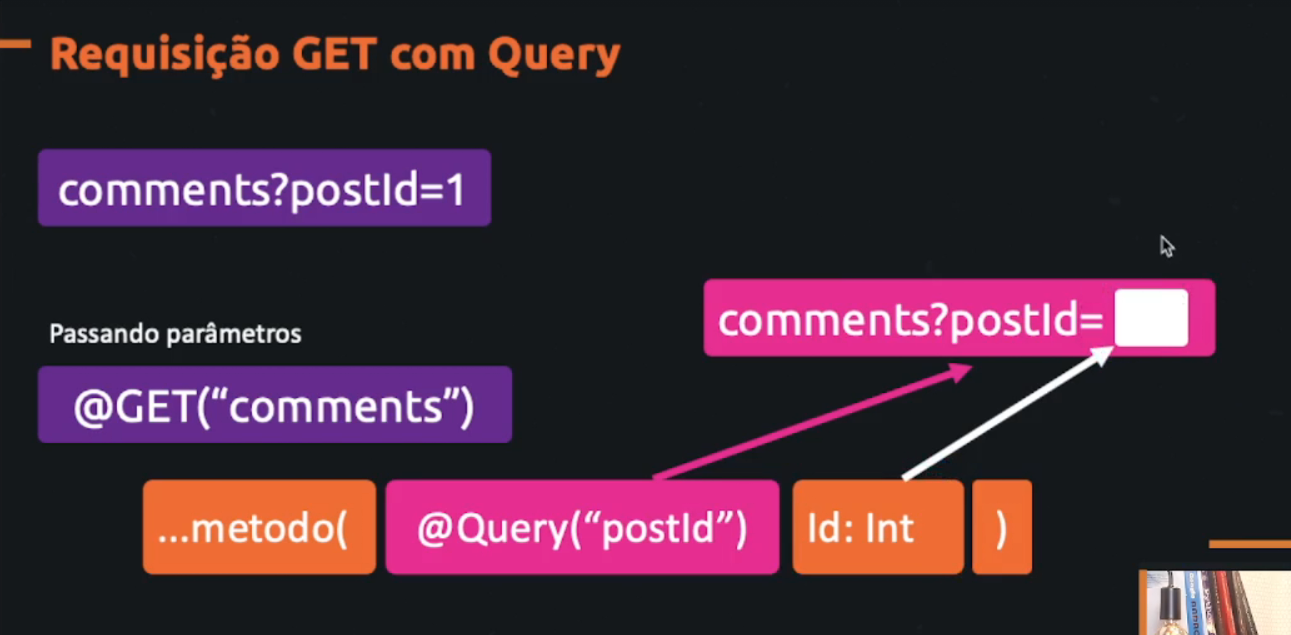
Ele vai utilizar uma URL de maneira diferente, repare na tradicional 1 e query 2

OPCAO 1 usando PATH = Caminho

OPCAO 2 Usando Query



# Criando um get do tipo Query



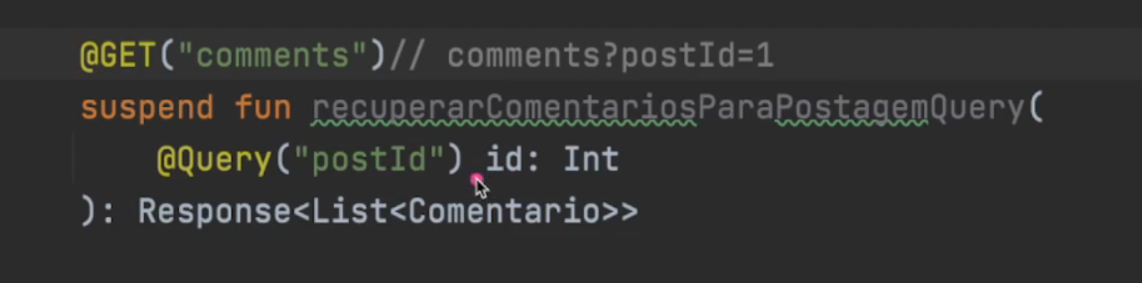
Vai ficar dessa forma, o valor do parâmetro que vai ser passado depois.

No caso o ID.

Atenção o nome que colocamos dentro do Query precisa ser o mesmo nome que esta dentro da api.



Por baixo dos panos quando a Retrofit for criar a interação vai ficar dessa forma



O ID vai ser o id que você vai passar.

Ou seja oque você colocar no parâmetro vai ser usado la no local do 1

Imagina que queremos utilizar mais parâmetros.

Então fica dessa forma ele divide os parâmetros com o &

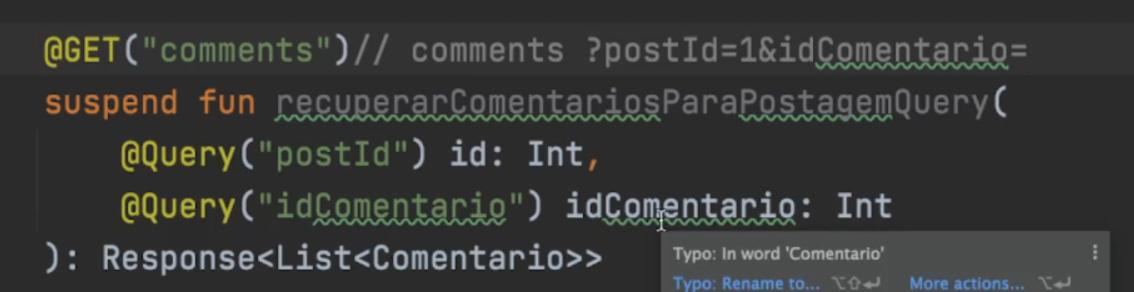
Ficando assim

Comments?postID=1 & idComentario= 1

Nesse exemplo iriamos acessar o post 1

E o comentário 1 desse post...

Agora se quiser colocar mais parâmetros só usar o & colocar mais parâmetros.

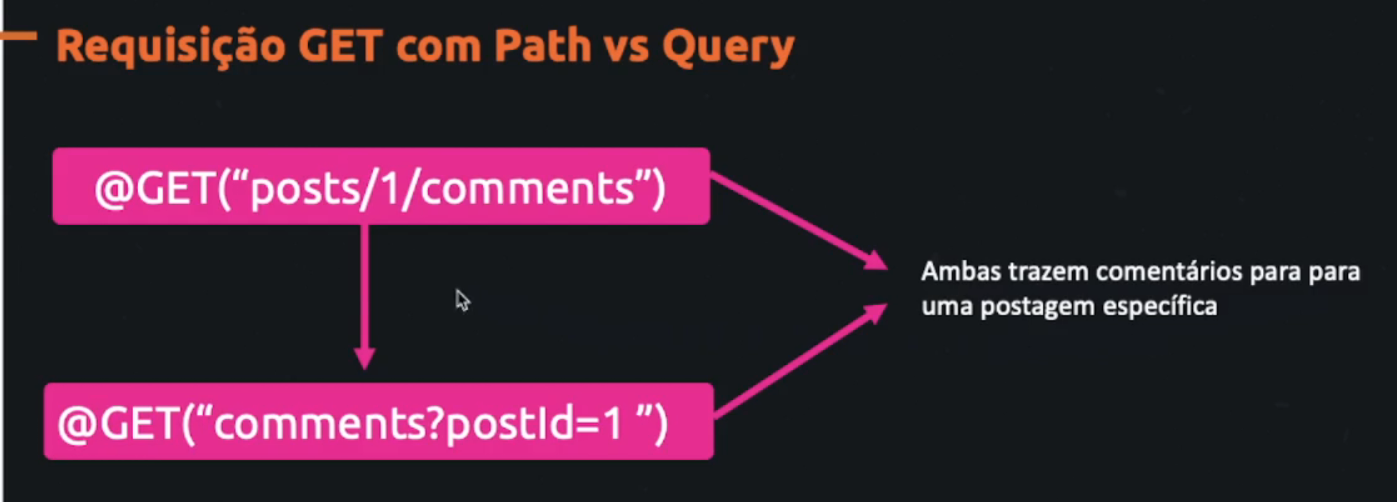


# Criando método e botão para exibir as postagens.

binding.btnPostagensComQuery.setOnClickListener **{** // Recuperando comentarios com Query  
 // Vamos recuperar os comentarios de uma postagem, no caso vai ficar como padrao postagem 1  
 *CoroutineScope*(Dispatchers.IO).*launch* **{** recuperarComentariosDaPostagemComQuery()  
 **}  
  
 }**}  
  
private suspend fun recuperarComentariosDaPostagemComQuery() {  
 var retorno : Response<List<Comentario>>? = null  
  
 try {  
 val ComentarioAPI = retrofit.create(ComentarioAPI::class.*java*)  
 retorno = ComentarioAPI.recuperarComentariosParaPostagemQuery(1)  
 }catch (e : Exception) {  
 e.printStackTrace()  
 Log.i("info\_comentario", "ERRO: Comentario não localizado!")  
 }  
  
 if (retorno != null && retorno.*isSuccessful*) {  
 val listaDeComentario = retorno.body()  
 var comentariosJuntos = ""  
 listaDeComentario?.*forEach***{** val id = **it**.id  
 val nome = **it**.name  
 val email = **it**.email  
 val comentario = **it**.comentario  
  
  
 comentariosJuntos += "\nID: $id\n" +  
 "Nome: $nome\n" +  
 "E-mail: $email\n" +  
 "$comentario"  
  
 withContext(Dispatchers.Main) **{** binding.textComentario.*text* = comentariosJuntos  
 **}** Log.i("info\_comentarios", "$id - $comentario")  
  
 **}** }  
}

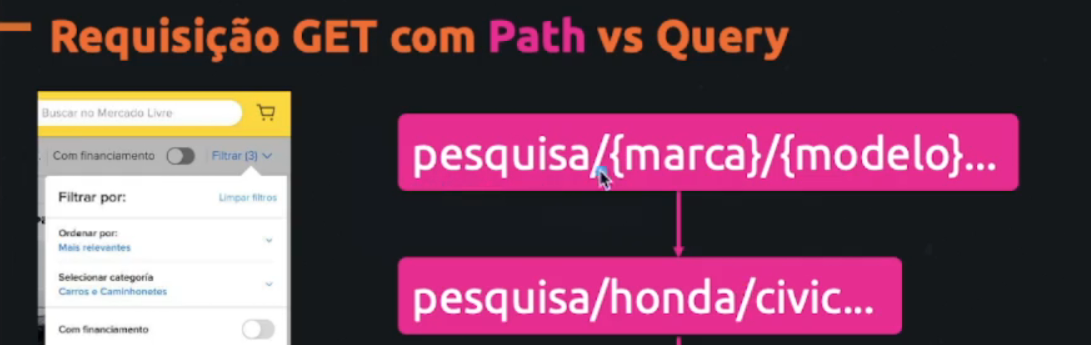
# Diferença entre PATH ou QUERY

## Path

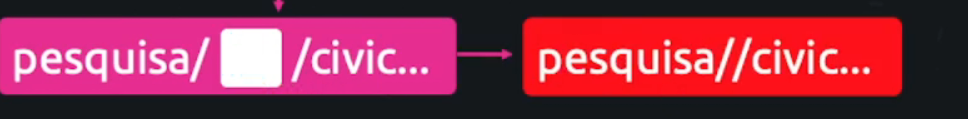


Imagina o mercado livre e a gente fazer uma requisição usando PATH

Seria uma requisição para carros.



Mas imagina que não queremos pesquisar a marca e somente o modelo, teríamos um problema



Ficaria dessa forma e a rota seria invalida.

Ou seja o path tem esse problema pq ele obriga você a passar os valores para criar o caminho.

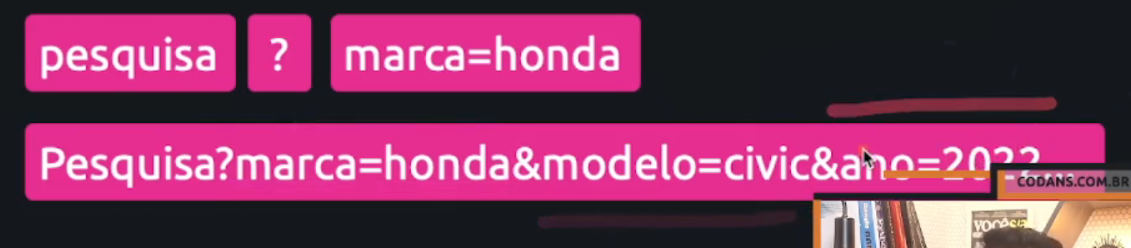
## Query

Já aqui podemos passar quantos parâmetros quiser, podemos passar somente a marca ou somente o modelo



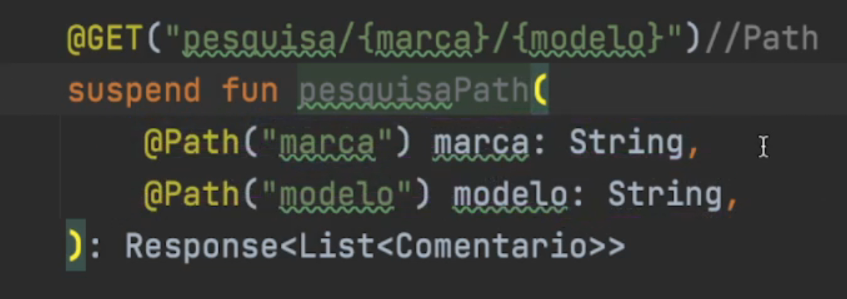
Nesse exemplo a cima passamos somente o modelo.

Também poderíamos fazer outro tipo de pesquisa passando outros dados.

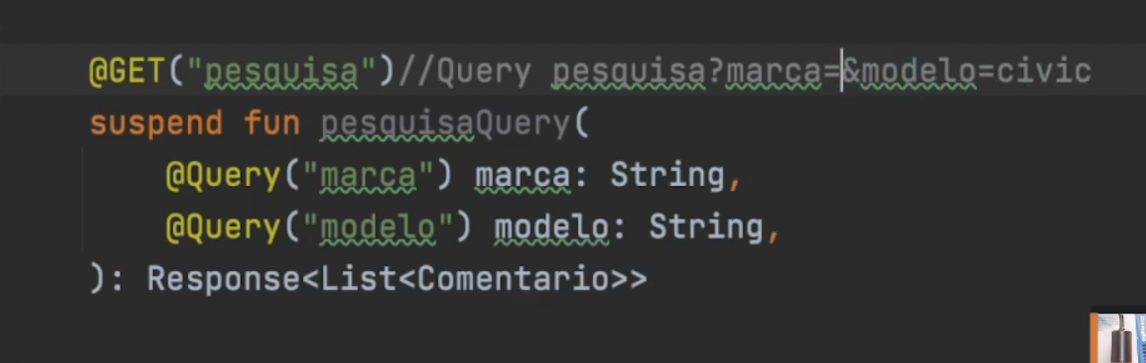


Resumindo quando você utiliza QUERY você tem mais flexibilidade.

# Exemplos práticos de path e query



Nesse exemplo estamos utilizando o path, para ele funcionar preciso passar a marca e modelo se não o caminho fica invalido.



Nesse exemplo podemos passar a marca

Podemos passar somente o modelo

Repara que o marca esta vazio, somente com o =

Estamos passando nesse exemplo somente o modelo.