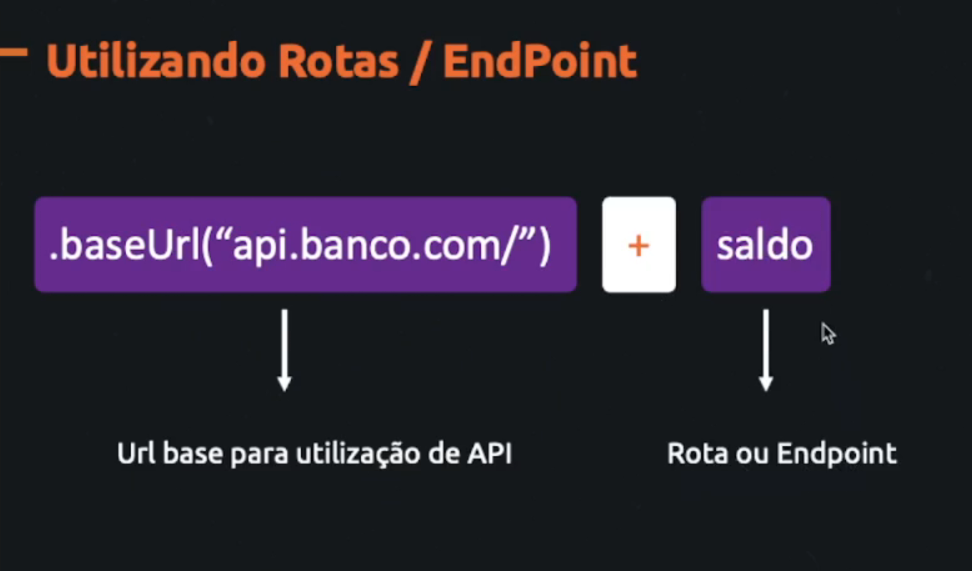
# END POINT / ROTAS



End point seria a informação

Imagina na API do banco, temos o site padrão e temos também os end POINTS como, saldo, transferência, limite de credito...





# Passando endereço BASE

class RetrofitHelper {  
 // Queremos acessar os recursos sem precisar instanciar a classe  
 companion object {  
 val retrofit = Retrofit.Builder()  
 **.baseUrl("https://viacep.com.br/")**  
 .addConverterFactory(GsonConverterFactory.create()) // Conversor para JSON || XML  
 .build()  
 }

}

# Fazendo a requisição com uma coroutina

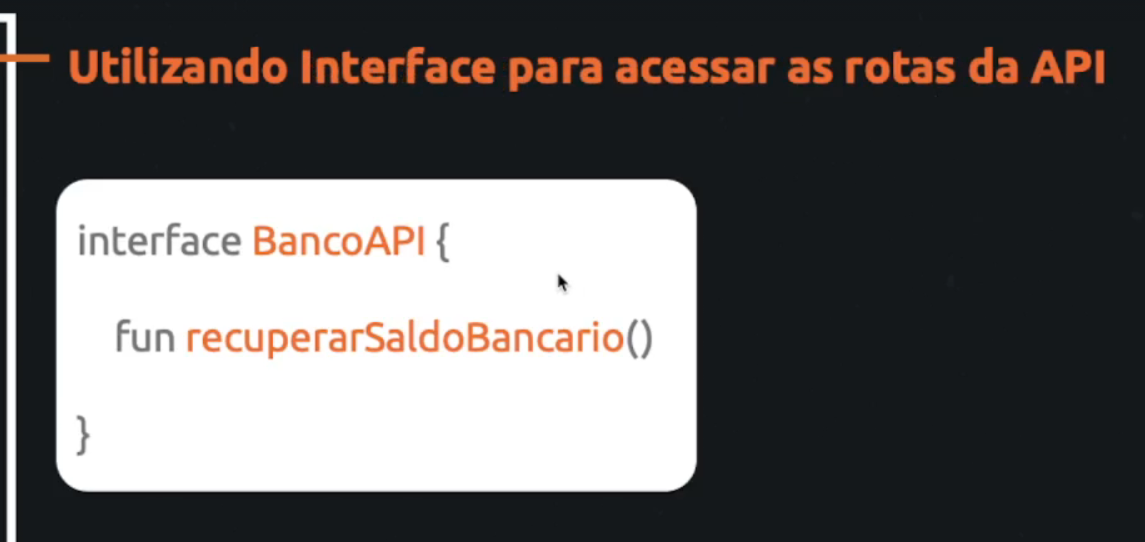
Usamos a IO para buscar do servidor, assim não deixamos a aplicação travada.

binding.btnIniciar.setOnClickListener **{** *CoroutineScope*(Dispatchers.IO).*launch* **{** recuperarEndereco()  
 **}  
}**

# Metodo para recuperar endereço

Criamos esse método suspend para evitar erros, ele só vai exibir outra coisa assim que ele terminar de recuperar os dados

Para recuperar o endereço precisamos de uma interface para configurar as rotas da API

Pq fazemos isso? Pq a própria Retrofit vai usar essa interface para configurar tudo por baixo dos panos.

## Criando interface

interface EnderecoAPI {  
 suspend fun recuperarEndereco()  
}

## Usando dentro do método

private suspend fun recuperarEndereco() {  
 */\*\* Explicando o codigo  
 \* O metodo create ele faz assim:  
 \* Ele recebe a interface e ele mesmo cria e retorna para voce o objeto  
 \* Sem voce precisar de uma classe para usar o metodo  
 \* ai esse objeto permite voce acessa o metodo recuperarEndereco  
 \*/* val enderecoAPI = retrofit.create( EnderecoAPI::class.*java* )  
 enderecoAPI.recuperarEndereco()  
}

Resumindo o método create

Temos apenas uma interface e não um método ainda de recuperarEndereco()

Então o método create a partir de uma interface ele cria o método e retorna para você no objeto.

\*\*\* O método que esta dentro da interface \*\*\*

Agora o método create vai criar um objeto e esse objeto vai permitir que você acesse o método

Por baixo dos panos a Retrofit vai, pegar a interface e criar uma classe e te retornar o objeto.

Ai sim podemos utilizar o método.