07

Configurando o Retrofit

Agora que temos o Retrofit no nosso projeto precisamos configurá-lo! Para isso, inicialmente, vamos criar uma classe que ficará responsável em inicilizá-lo para nós, portanto, crie a classe `RetrofitInicializador` no pacote **br.com.alura.agenda.retrofit**: Em seguida crie um construtor que faça a instância da classe `Retrofit.Builder` que é justamente a classe que realiza a configuração do Retrofit:

```
package br.com.alura.agenda.retrofit;

import retrofit2.Retrofit;

public class RetrofitInicializador {
    public RetrofitInicializador() {
        new Retrofit.Builder();
    }
}
```

Configurando URL base

Dessa forma garantimos que todas as vezes que essa classe for instanciada o retrofit será configurado! Agora vamos realizar a primeira configuração que é justamente a URL base conforme vismo em aula. Para isso adicione uma URL base a partir do método `baseURL()` enviando como parâmetro o endereço `http://seu_ip:8080/api/` . Vale lembrar que é necessário adicionar a **/** no final do endereço, pois se não o Retrofit vai lançar uma exception!

Adicionando o conversor ao projeto

Após a inserção da URL base, precisamos agora adicionar o conversor que ficará responsável em serializar e deserializar tanto os objetos quantos os documentos JSON. Para isso, no mesmo código, adicione também o método `addConverterFactory()` . Observe que agora esse método espera um `Factory` que é justamente uma fábrica de conversores! Atualmente não temos nenhuma, ou seja, precisamos agora ou implementar uma (método mais trabalhoso) ou então podemos usar algumas existentes que o próprio Retrofit nos oferece por meio de "plugins" que são dependências que podemos adicionar no projeto. Veja abaixo a lista de conversores que temos disponíveis:

- **Gson** - com.squareup.retrofit2:converter-gson
- **Jackson** - com.squareup.retrofit2:converter-jackson
- **Moshi** - com.squareup.retrofit2:converter-moshi
- **Protobuf** - com.squareup.retrofit2:converter-protobuf
- **Wire** - com.squareup.retrofit2:converter-wire
- **Simple Framework** - com.squareup.retrofit2:converter-simpleframework
- **Scalars** - com.squareup.retrofit2:converter-scalars

Em aula vimos o conversor do Jackson que é o que tenho mais familiariedade, ou seja, que estou mais acostumado, portanto, recomendo que o utilize! Porém, se preferir algum outro diferente dele, fique a vontade. Para adicionar a lib basta apenas fazer o mesmo procedimento realizado para adicionar o Retrofit, a única diferença é que ao invés de escrever **retrofit** no input para buscar as libs, basta apenas copiar e colar o nome completo da dependência conforme a lista mencionada acima, ou seja, se for utilizar a do Jackson, copie e cole o valor **com.squareup.retrofit2:converter-jackson**. Em seguida aparecerá a dependência com a versão, escolha a mesma versão que foi escolhida para o Retrofit, ou seja, a versão 2.1.0, então clique em **OK** e veja se a dependência aparece na lista, caso aparecer clique em **OK** novamente.

Adicionando o conversor no Retrofit

Agora que temos o conversor no nosso projeto basta apenas adicionar a sua fabrica, no caso do Jackson, basta apenas chamar o método estático `create()` da classe `JacksonConverterFactory` .

Construíndo o objeto do Retrofit

Com a URL base e fábrica de conversores configurada precisamos apenas finalizar esse builder, para isso chame o método `build()` e atribua para um atributo da classe do tipo `Retrofit`. Lembre-se que esse atributo não é modificado, portanto, por boa prática, deixo-o como `final`.