Opcional: Protegendo contra falhas ao ouvir mensagens

Para ouvir as mensagens usando Retrofit, sabemos que é necessário passarmos um *callback* para o método enqueue. No nosso projeto, criamos a classe OuvirMensagensCallback com esse propósito. E no método onResponse, disparamos um evento com a mensagem caso a resposta seja uma resposta de sucesso. Vamos refrescar nossa memória?

// OuvirMensagensCallback.java  
  
public class OuvirMensagensCallback implements Callback<Mensagem> {  
 private EventBus eventBus;  
  
 public OuvirMensagensCallback(EventBus eventBus, MainActivity activity) {  
 this.activity = activity;  
 this.eventBus = eventBus;  
 }  
  
 @Override  
 public void onResponse(Call<Mensagem> call, Response<Mensagem> response) {  
 if(response.isSuccessful()) {  
 Mensagem mensagem = response.body();  
  
 eventBus.post(new MensagemEvent(mensagem));   
 }  
  
 }  
  
 @Override  
 public void onFailure(Call<Void> call, Throwable t) {  
  
 }  
  
}

Mas o que fazer em caso de falha na requisição? Ou caso a resposta não seja uma resposta de sucesso (ou seja, o código da resposta deve estar entre 200 e 300)?

Podemos apresentar uma mensagem de erro pro usuário e pedir pra ele tentar novamente. Mas será que faz sentido? O ideal é caso haja algum erro ao tentar ouvir as mensagens, imediatamente se recuperar da falha e tentar ouvir mensagens novamente. Dessa forma a gente consegue deixar o aplicativo *Fail-safe*.

Nessa estratégia estamos deixando de lado o contador de tentativas e o [tempo entre elas](https://en.wikipedia.org/wiki/Exponential_backoff). Que é interessante para evitarmos que o servidor, em caso de indisponibilidade, seja sobrecarregado com várias requisições ao se tornar de novo disponível.

Como podemos chamar o ouvirMensagens novamente em caso de falha? Sabemos que o *EventBus* chamará para gente o método ouvirMensagem da classe MainActivity ao chegar um MensagemEvent. Uma das formas de implementar, seria aproveitar deste fato e lançar um FailureEvent.

// FailureEvent.java  
  
public class FailureEvent {  
  
  
}

// OuvirMensagensCallback.java  
  
public class OuvirMensagensCallback implements Callback<Mensagem> {  
 private EventBus eventBus;  
  
 public OuvirMensagensCallback(EventBus eventBus, MainActivity activity) {  
 this.activity = activity;  
 this.eventBus = eventBus;  
 }  
  
 @Override  
 public void onResponse(Call<Mensagem> call, Response<Mensagem> response) {  
 if(response.isSuccessful()) {  
 Mensagem mensagem = response.body();  
  
 eventBus.post(new MensagemEvent(mensagem));   
 }  
  
 }  
  
 @Override  
 public void onFailure(Call<Void> call, Throwable t) {  
 eventBus.post(new FailureEvent());   
 }  
  
}

Vamos precisar de um método anotado @Subscribe na classe MainActivity, que receba FailureEvent, para tratar esse evento:

// MainActivity.java  
  
@Subscribe  
public void lidarCom(FailureEvent event) {  
  
}

Dentro desse método precisamos chamar o ouvirMensagens. Nesse caso, não houve um evento de mensagens, apenas queremos chamar o método. Portanto vamos chamá-lo passando null:

// MainActivity.java  
  
@Subscribe  
public void lidarCom(FailureEvent event) {  
 ouvirMensagens(null);  
}