

Convertendo json para aluno

Caso não tenha o projeto com as alterações realizadas na aula passada, você pode baixá-lo por meio [deste link \(https://github.com/alura-cursos/android-sync/archive/3582f56b979d05cc22878e103b2ab18f1f34785a.zip\)](https://github.com/alura-cursos/android-sync/archive/3582f56b979d05cc22878e103b2ab18f1f34785a.zip).

Embora estejamos recebendo as informações dos alunos que estão sendo salvos/editados no servidor, ainda não estamos atualizando a lista! Para isso, o nosso primeiro passo será de fato converter a mensagem que estamos recebendo do servidor para um aluno da App. Para isso vá até o método `onMessageReceived()` faça a chamada ao método `converteParaAluno()` enviando o `Map mensagem` como parâmetro. Em seguida, use o atalho **Alt + Enter** para criar o método já com o parâmetro.

Pegando valor do Map

Dentro do método `converteParaAluno()` o nosso primeiro passo será justamente tentar pegar o valor contido no `Map`, para isso precisamos de uma chave! Portanto, extraia a chave `"alunoSync"` para a variável `chaveDeAcesso` do tipo `String`. Em seguida, antes de tentar pegar o valor do `Map` faça um `if` verificando se existe a chave utilizando o método `contains()`. Dentro do `if` utilize o método `get()` do `Map` enviando a `chaveDeAcesso` como parâmetro. Então atribua o retorno para a variável `json` do tipo `String`.

Convertendo String para objeto

Com o `json` em mãos o que precisamos agora é de fato realizar a conversão para o objeto da nossa App. Atualmente o Retrofit faz manualmente esse processo de conversão, ou seja, o único método que conhecemos até o momento seria por meio da classe `JSONStringer` que era aquele método um pouco trabalhoso... Entretanto, já que estamos usando o Jackson como conversor do Retrofit, podemos também utilizá-lo para converter essa `String`! Para isso faça instância da classe `ObjectMapper` e devolva para o objeto `mapper`.

Em seguida, a partir do `mapper` use o método `readValue()` enviando como parâmetro o `json` e a classe que espera que ele converta. Nesse caso, desejamos converter para a classe `AlunoSync` que é justamente a forma na qual o servidor se comunica, isto é, manda as informações para nós, portanto, envie o segundo parâmetro com o valor `AlunoSync.class`.

Note que o Android Studio estará reclamando dessa linha de código, pois ela lança uma `Exception` que precisa ser tratada, para resolver isso, basta usar o atalho **Alt + Enter** e escolha a opção **Surround with try/catch** para que o código seja tratado. Por fim atribua o retorno desse método para a variável do tipo `AlunoSync`

