05

Verificando se a data é mais recente

Realizamos o primeiro passo da padronização de sincronização dos nosso alunos centralizando a sincronização para o método `sincroniza()` da classe `AlunoService`. Entretanto, ainda precisamos fazer com que esse método seja esperto o suficiente e verifique se a versão que veio do servidor é uma versão mais recente ou não, para então tomar a decisão cabível.

Em outras palavras, dentro do método `sincroniza()` adicione um `if` logo depois do trecho de código que pega a versão do `AlunoSync`.

O `if` vai verificar se a partir da versão que foi extraída se é uma versão nova ou não. Para isso, envie para o teste do `if` o método `temVersaoNova()` com o parâmetro `versao`. Em seguida, crie o método `temVersaoNova()`.

> **DICA:** Para criar o métodos de uma maneira fácil utilize o atalho da IDE `Ctrl + Alt + M`, dessa forma, ela já vai criar o método com o retorno esperado.

Por padrão deixe o retorno do método `temVersaoNova()` como `false`, justamente porque a ideia do método é justificar que a versão que veio do servidor é nova, ou seja, todos os testes terão o objetivo de retornar um valor `true` caso isso for verdade.

O primeiro teste para indicar que a versão vinda do servidor é nova, é verificar se no Shared Preferences existe alguma versão salva, portanto, adicione um `if` que verifica isso, caso não tiver uma versão salva, retorne `true`.

Caso o primeiro teste falhar significa que temos uma versão salva, portanto precisamos verificar se a versão do servidor é mais recente que a do cliente. Para isso, precisamos converter a `String` da versão na API `Date` do Java que nos fornece métodos capazes de avaliar se uma data é mais recente do que outra.

Para converter a `String` para `Date`, utilize o formatador `SimpleDateFormat` utilizando o seguinte *pattern*: `"yyyy-MM-dd'T'HH:mm:ss.SSS"`. Agora que temos o formatador utilize o método `parse` do formatador enviando a `versao` e atribua para a variável `dataExterna` que representa a versão do servidor. Faça o mesmo com a versão salva no Shared Preferences, porém, nomeie com o nome `dataInterna`.

> Quando lidamos com conversões, geralmente os métodos lançam exceptions do tipo checked, ou seja, ou precisamos lançá-las ou tratá-las. Nesse caso trate a exception com o try catch, pois caso falhar a conversão e a exception for lançada, o retorno padrão será `false`, portanto, não iremos salvar uma versão que não deu certo na conversão.

Agora que temos dois objetos do tipo `Date`, somos capazes de compará-los. Então utilize o objeto `dataExterna` e chame o método `after()` enviando o objeto `dataInterna`, esse método é responsável em saber se uma data é mais recente do que outra, nesse caso, se a data do servidor é mais recente que a data interna, caso isso for verdade ele devolve `true` caso contrário `false`, em outras palavras, retorne o retorno desse método.

Por fim, dentro do `if` do método `sincroniza()`, adicione todo o código que salva a versão nova no Shared Preferences e sincroniza as informações do servidor com o banco de dados interno.