Editando alunos

Já conseguimos recuperar os dados de um aluno clicado no formulário. O nosso problema é que quando alteramos os dados desse aluno no formulário e salvamos, nossa aplicação adiciona um novo aluno, em vez de simplesmente salvar as edições. Na nossa lista teremos nomes duplicados por causa disso.

Vamos acrescentar uma opção de alterar os dados e salvar as alterações em vez de gerar um novo aluno como está acontecendo.

Se estamos falando do *checkmark*, estamos falando de um clique de um botão do menu. Então, vamos no FormularioActivity.java, no métodoonOptionsItemSelected, que fala que um item do menu foi selecionado. Encontramos o seguinte:

@Override   
 public boolean onOptionsItemSelected(MenuItem item) { switch (item.getItemId()) {   
 case R.id.menu\_formulario\_ok;  
 Aluno aluno = helper.pegaAluno(); AlunoDAO dao = new AlunoDAO(this);   
 dao.insere(aluno);  
 dao.close();   
 Toast.makeText(FormularioActivity.this, "Aluno " + aluno.getNome() + " salvo!", Toast.LENGTH\_SHORT).show();  
  
 finish();   
 break;   
 }   
  
 return super.onOptionsItemSelected(item);   
 }

Repare que dentro dele fizemos uma verificação para saber qual item do menu foi clicado. Então, nós falamos que caso clicássemos no botão menu\_formulario\_ok, teríamos um comportamento que pegava o aluno do formulário e com esse aluno instanciávamos o dao e chamávamos o insere. Agora, não é sempre que vamos querer chamar o insere, pois, podemos estar simplesmente fazendo uma alteração e não querendo inserir um novo aluno.

Para resolver esse problema podemos fazer um outro tipo de verificação. Se o aluno tiver um id e se não for nulo, significa que estamos simplesmente alterando esse aluno. Acrescentaremos duas linhas abaixo de Aluno aluno = helper.pegaAluno() uma verficação if (aluno.getId() !=null). E com isso dizemos que se é diferente de nulo, deve ser alterado, assim, digitamos dao.altera(aluno). Importante que o Aluno dao = new AlunoDAO(this)fique acima da verificação.

Caso o id do aluno for nulo, acrescentamos, else e embaixo dele copiamos e colamos o dao.insere(aluno). Isso que acabamos de modificar será na linha de baixo do dao.altera.

Teremos:

@Override   
 public boolean onOptionsItemSelected(MenuItem item) { switch (item.getItemId()) {   
 case R.id.menu\_formulario\_ok;  
 Aluno aluno = helper.pegaAluno();  
  
 AlunoDAO dao = new AlunoDAO(this);  
 if (aluno.getId() != null) {  
 dao.altera(aluno);  
 } else {  
 dao.insere(aluno);  
 }  
 dao.close();   
 Toast.makeText(FormularioActivity.this, "Aluno " + aluno.getNome() + " salvo!", Toast.LENGTH\_SHORT).show();  
  
 finish();   
 break;   
 }   
  
 return super.onOptionsItemSelected(item);   
 }

Agora, só falta implementar o método altera. Para isso damos um "Alt+Enter" em cima dele e nosso método será criado no *AlunoDAO.java*, no final da página. Ficaremos com:

public void altera(Aluno aluno) {  
}

Agora, podemos inserir o comportamento desse método. Primeira coisa que precisamos é um banco de dados que possa ser alterado, digitaremos um SQLiteDatabase db = getWritableDatabase().Uma vez que temos o WritableDatabase precisamos de um método que faça por nós a atualização, uma vez que, fazer a mão seria extremamente complicado. Usaremos o db.update e para completá-lo é bem parecido com o que já fizemos em outros momentos. Completaremos com a tabela, "Alunos".

public void altera(Aluno aluno) {  
 SQLiteDatabase db = getWritableDatabase();   
  
 db.update("Alunos");   
 }

Para seguir completando o db.update também teremos que inserir os dados que usaremos, um ContentValue. Mas, teremos que criar esse ContentValue, então, na linha de cima digitaremos ContentValues dados = new ContentValues. Na linha de baixo disso poderíamos acrescentar um dados.put("nome", aluno.getNome()), mas lembra que já fizemos isso antes? Se você rolar sua barrinha para cima vai ver já temos tudo isso feito no insere(Aluno aluno). Uma coisa que pode ser feita é extrair o método, já que fizemos isso anteriormente.

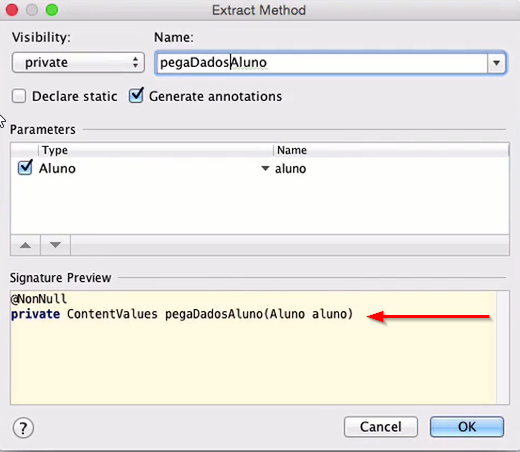
Lembrando o que temos no insere(Aluno aluno):

public void insere(Aluno aluno) {   
 SOLiteDatabase db = getWritableDatabase();   
  
 ContentValues dados = new ContentValues();  
 dados.put("nome", aluno.getNome()); dados.put("endereco", aluno.getEndereco());  
 dados.put("telefone", a1uno.getTelefone());  
 dados.put("site", aluno.getSite());  
 dados.put("nota", aluno.getNota());   
  
 db.insert("A1unos", null, dados);  
 }

Selecionamos com o mouse lá no insere(Aluno aluno) o seguinte:

ContentValues dados = new ContentValues();  
 dados.put("nome", aluno.getNome()); dados.put("endereco", aluno.getEndereco());  
 dados.put("telefone", a1uno.getTelefone());  
 dados.put("site", aluno.getSite());  
 dados.put("nota", aluno.getNota());

E clicamos com o botão direito e seguimos por Refactor> Extract> Method. Vai abrir uma janela pedindo para nomear o método, no campo *Name*, chamaremos de pegaDadosDoAluno. Perceba que na *signature preview* conseguimos ver que ele criará um método que recebe um (Aluno aluno) e devolve um ContentValues.



Damos um "OK" e ele cria o seguinte método:

@NonNull  
 private ContentValues pegaDadosDoAluno(Aluno aluno) {  
 ContentValues dados = new ContentValues();  
 dados.put("nome", aluno.getNome()); dados.put("endereco", aluno.getEndereco());  
 dados.put("telefone", aluno.getTe1efone());  
 dados.put("site", aluno.getSite());  
 dados.put("nota", aluno.getNota()); return dados;  
}

Agora, vamos voltar no nosso altera(Aluno aluno) e completamos o ContentValuescom o aluno. Teremos o ContentValues dados = pegaDadosDoAluno(aluno).

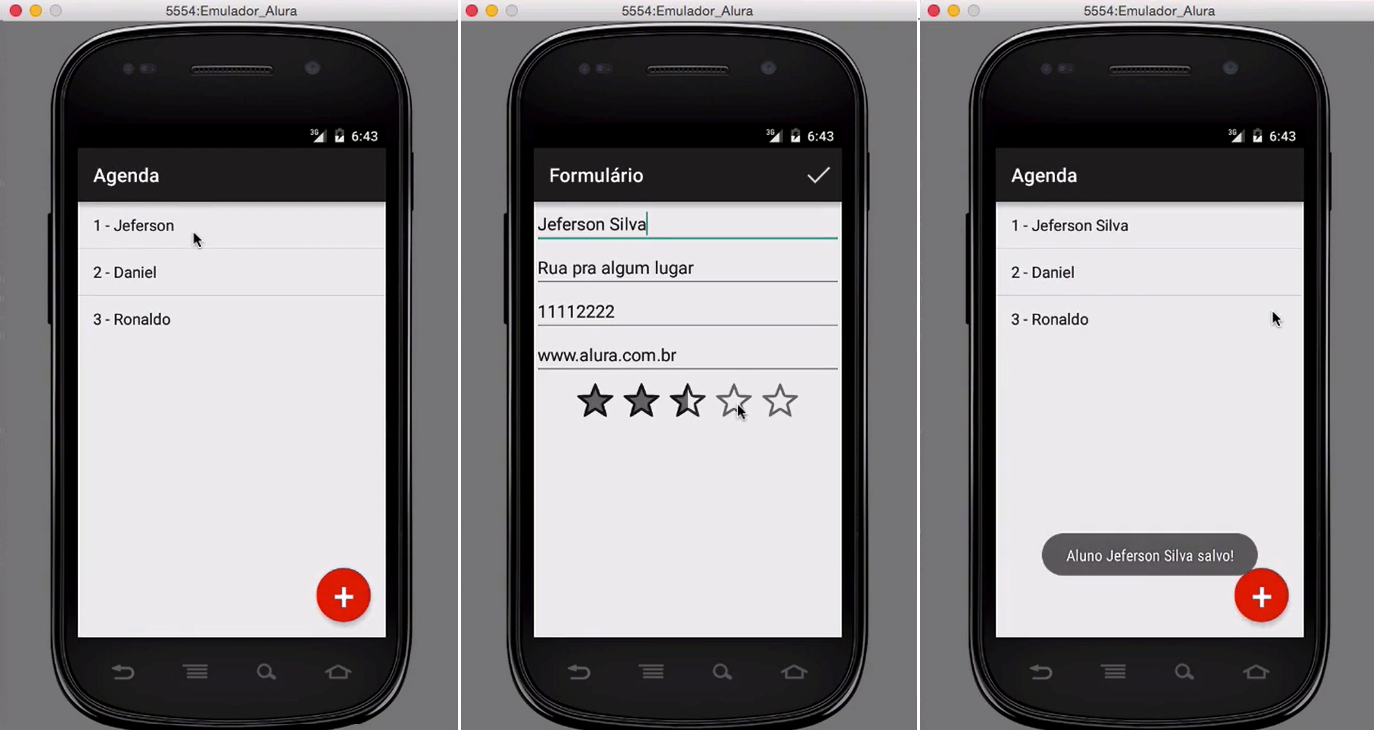
Vamos agora no db.update e completar o resto dos parâmetros do parênteses. Ao em vez de concatenar o parâmetro, colocaremos um parêmtro que é uma ? e esse será substituído pelo parâmetro seguinte, que é uma *Array* de *String*, um params. Teremos um update("Aluno", dados, "id=?", params).

Na linha de cima do updatetemos que construir também uma *Array* de *String*, uma String[] params = {} e dentro das chaves colocaremos os valores que entrarão no lugar da interrogação, no caso, o id do aluno. Acrescentaremos no parenteses aluno.getId, lembrando que como é um Long temos que converter, usaremos uma .toString. Digitaremos String[] params = {aluno.getId().toString()}. Ficaremos com o seguinte:

public void altera(Aluno aluno) {  
 SQLiteDatabase db = getWritableDatabase();  
  
 ContentValues dados = pegaDadosDoAluno(aluno);  
  
 String[] params ={aluno.getId().toString()};  
 db.update("Aluno", dados, "id = ?", params);   
 }

Agora, só falta testar. Vamos salvar e dar um *play*.

Vamos editar o aluno "Jeferson", vamos acrescentar nele o sobrenome "Silva" e veremos que agora ele altera o aluno.



E também conseguimos inserir um novo aluno sem problemas! E, agora, estamos com os passos básicos da nossa Agenda prontos.