Inserindo no banco

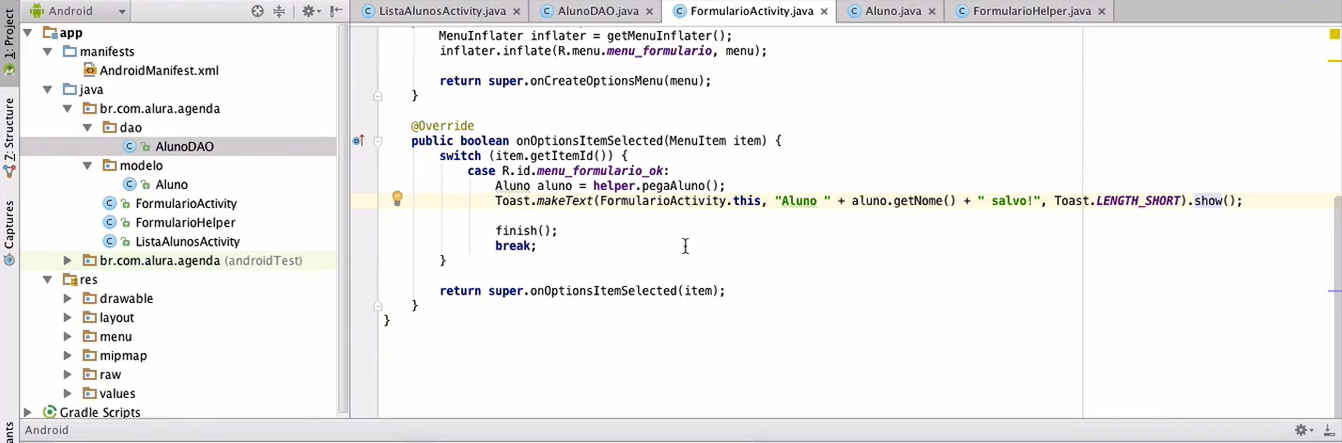
Nós já preparamos parte do caminho para um banco de dados. Vamos continuar nesse percurso para fazer com que o aluno seja de fato selecionado quando clicarmos no *check mark*.



Vamos jogar nosso aluno de fato no banco de dados?

Para alterar o comportamento, quando clicarmos no nosso *check mark* no formulário, vamos voltar a FormularioActivity.

Encontraremos a seguinte tela:



Bom, o que queremos fazer é simplesmente dizer para o código inserir o aluno, sem termos que nos preocupar com o banco de dados. Vamos na onOptionsItemSelected e temos o seguinte:

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

Depois do Aluno aluno = helper.pegaAluno, damos um "Enter" e na linha de baixo vamos instanciar o objeto DAO responsável por fazer essa abstração. Vamos criar uma referência para o DAO e quando acrescentarmos o AlunoDAO dao = new AlunoDAO ele vai pedir como parâmetro o contexto, aquele que temos na AlunoDAO.java, no construtor, isto é, nossa identificação para o *Android*. O contexto vai ser acrescentado entre os parênteses, digitaremos this e ficaremos com, AlunoDAO dao = new AlunoDAO(this).

Vamos inserir algo que fale para o dao que queremos que os novos nomes sejam adicionados. Digitaremos dao.insere e como parâmetro escreveremos o aluno dao.insere(aluno). Só que esse método ainda não existe, portanto, daremos um "Alt+Enter" e "Create method insere". Ele irá criar o método na aba AlunoDAO.java onde teremos o seguinte:

public class AlunoDAO extends SQLiteOpenHelper {  
 public class AlunoDAO (Context context) {  
 super(context, "Agenda", null, 1);   
 }  
  
 @Override  
 public void onCreate (SQLiteDatabase db) {  
 String sql = "CREATE TABLE Alunos (id INTEGER PRIMARY KEY, nome TEXT NOT NULL, endereco TEXT, telefone TEXT, site TEXT, nota REAL);";  
 db.execSQL(sql);   
 }  
  
 @Override  
 public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {  
 String sql = "DROP TABLE IF EXISTS Alunos";  
 db.execSQL(sql);  
 onCreate(db);   
 }  
  
 public void insere(Aluno aluno) {  
 }  
}

Para inserirmos esse aluno poderíamos escrever uma instrução String e utilizar um INSERT INTOe discriminar as colunas e o respectivos valores. Teríamos o seguinte:

public void insere(Aluno aluno) {  
 String sql = "INSERT INTO Alunos (nome, endereco, telefone, site, nota) VALUES (" + aluno.getNome())  
 }

O que aconteceria se alguém fosse no nosso formulário e preenchesse com o nome de Paulo seguido de "'); DROP TABLE Alunos;"?



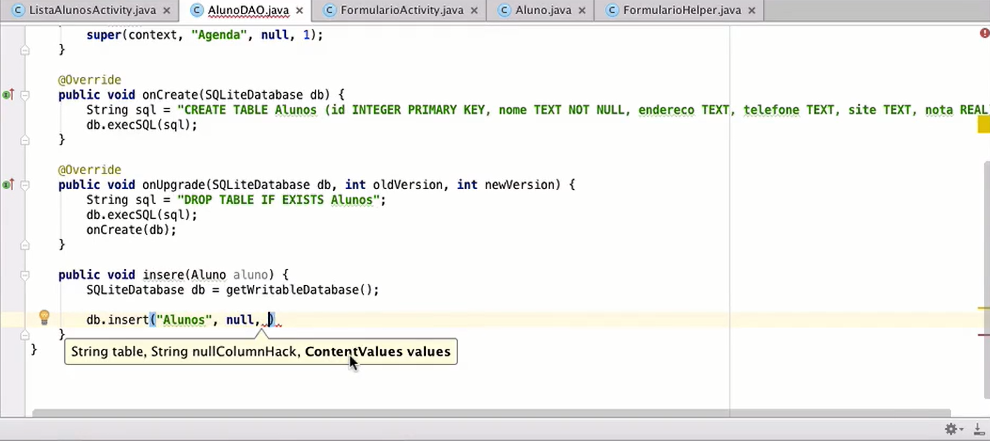
Se o java concatenar o que foi escrito no campo nome, que é "Paulo'); DROP TABLE" teremos mais ou menos o seguinte:

public void insere(Aluno aluno) {  
 String sql = "INSERT INTO Alunos (nome, endereco, telefone, site, nota) VALUES ('Paulo'); + DROP TABLE Alunos;";  
 }

Se executarmos a sql vai executar a primeira instrução e depois, a segunda, que é DROP TABLE Alunos que vai jogar fora a tabela inteira que construímos, isto é, ele manipula o nosso banco de dados. Essa prática é bem comum para quem gosta de destruir aplicações por aí...

Existem formas de cuidar para que isso não ocorra, em vez de buscarmos a mão essas ações vamos pedir para que a 'SQLiteOpenHelper' faça isso por nós, que ele faça a operação de inserção de novos alunos.

Para isso vamos inserir no insere(Aluno aluno) um método getWritableDatabase cuja referência é o Database para que justamente possamos escrever nele. Vamos dar um "Alt+Enter" nele e "Introduce local variable" e vamos chamar um db. Agora, temos a referência e falta inserir um db.insert para poder inserir algo, vamos colocar o primeiro parâmetro "Aluno", o segundo deixaremos como "null" , pois não utilizamos ele normalmente, a não ser que queiramos linha em branco, e não queremos fazer isso aqui. Por fim, ele vai pedir alguns valores. Ele vai pedir um "Content Value".



Vamos criar ele em cima do db.insert("Alunos", "null", ). Vamos criar um ContenValues, vou chamar de dados e instanciar ele , teremos ContentValues dados = new ContentValues(). agora, ele vai funcionar como uma espécie de *mapping* do *java*. Isto é, a ideia de chave e valor, na chave nome, guardaremos o valor nome do aluno, na chave endereço o endereço e assim por diante. Por isso, acrescentaremos abaixo de ContentValues o que queremos guardar dados, put para colocar algo e o que queremos colocar, seguido de onde vêm essas informações aluno e getNome. Teremos, dados.put("nome", aluno.getNome()). Faremos o mesmo com o resto dos dados que queremos guardar, como o endereço, site, telefone e nota alterando apenas o que queremos colocar e o get. E agora, passamos esse ContentValues como parâmetro no insert.

Temos o seguinte:

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

Vamos voltar no FormularioActivity.java para ver como ficou, instanciamos o dao e pedimos para ele inserir logo na linha abaixo. O dao não precisa saber nada do banco de dados. Ele simplesmente pede para inserir o "aluno". Quando fazemos uma operação com o dao é bom lembrar de fechar a conexão com o banco de dados. Para isso vamos utilizar um método close. Ficamos com, dao.close(). Lembrando que conseguimos chamar esse método por que ele já está na AlunoDAO na SQLiteOpenHelper.

Temos 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);  
 }

Por enquanto ainda não conseguimos visualizar o que fizemos através do emulador. Ficaremos com esse início de caminho traçado.