Universidade da Beira Interior
Faculdade de Engenharia
Departamento de Informática

Programação de Dispositivos Móveis
Guia para Aula Laboratorial 8
Licenciatura em Engenharia Informática
Licenciatura em Informática Web

Programming of Mobile Devices
Guide for Laboratory Class 8
Degree in Computer Science and Engineering
Degree in Web Informatics

Sumário
Desenvolvimento de uma pequena aplicação para manipulação de uma base de dados local. Análise e exploração da *shell* SQLite3 fornecida com o *Software Development Kit* (SDK) Android™.

Summary
Development of a small application for manipulating a local database. Analysis and exploration of the SQLite3 shell provided with the Android™ Software Development Kit (SDK).

Pré-requisitos:
Algumas das tarefas enunciadas a seguir requerem o acesso a um sistema com o *Software Development Kit* (SDK) para Android™ e a ferramenta de linha de comandos para automatização do processo de compilação de aplicações Gradle™ instalados ou, alternativamente, com permissões para instalação e configuração do kit e da ferramenta. Serão suficientes permissões para criar diretorias e ficheiros num disco local e para configurar variáveis de sistema, nomeadamente a *path*. É igualmente necessário ter acesso a uma versão e imagem da plataforma Android™ ou, alternativamente, a um dispositivo físico com este sistema operativo e com a opção de *debug* ativa. Um compilador Java instalado também é necessário.

1 Preliminares
Preliminaries

O guia laboratorial 8 elabora nos passos necessários a criação e compilação (*build*) de projetos de aplicações para a plataforma Android™ via linha de comandos. Esta abordagem, apesar de não comportar algumas das facilidades oferecidas por ambientes de desenvolvimento integrados, nomeadamente ambientes de edição da interface de utilizador *What You See Is What You Get* (WYSIWYG), permite a prototipagem bastante expedita de uma aplicação e não requer mais do que a versão *Stand-Alone* (lobo solitário) do *Software Development Kit* (SDK) Android™[1]. Depois do sistema estar devidamente configurado, 4 passos são suficientes para criar um projeto Android™, gerar o ficheiro `.apk` e instalar a aplicação num dispositivo (virtual ou real):

1. Inicializar o dispositivo móvel virtual ou ligar um real ao computador[2];
2. Gerar o projeto através do Android ™
3. Compilar o projeto com a ferramenta Gradle™, emitindo o comando
   `$ gradlew assembleDebug lint`
   na raiz do projeto;
4. e instalando a aplicação com um comando semelhante a
   `$ adb install path\NomeApp-debug.apk`.

[1] Ver `https://developer.android.com/sdk/installing/index.html?pkg=tools`

[2] Se o dispositivo for real, tem de ter a opção de depuração ativada.

Tarefa 1 *Task 1*

Como já vem sendo habitual, a primeira tarefa consiste em iniciar um *Android Virtual Device* (AVD). Para isso, pode emitir o comando `$ emulator -avd Nome_AVD`, incluído na pasta `tools` do SDK, e aguardar que o emulador seja lançado. Caso não exista nenhum AVD configurado, crie um[3]. O ideal será um emulador de uma versão superior à 4.0 do Sistema Operativo (SO).

Tarefa 2 *Task 2*

Verifique se as variáveis `ANDROID_HOME` e `JAVA_HOME` estão devidamente definidas com comandos parecidos com:

```
$ echo %ANDROID_HOME%
$ echo %JAVA_HOME%
$ echo %PATH%
```

[3] O guia laboratorial 1 contém uma breve discussão acerca deste assunto.

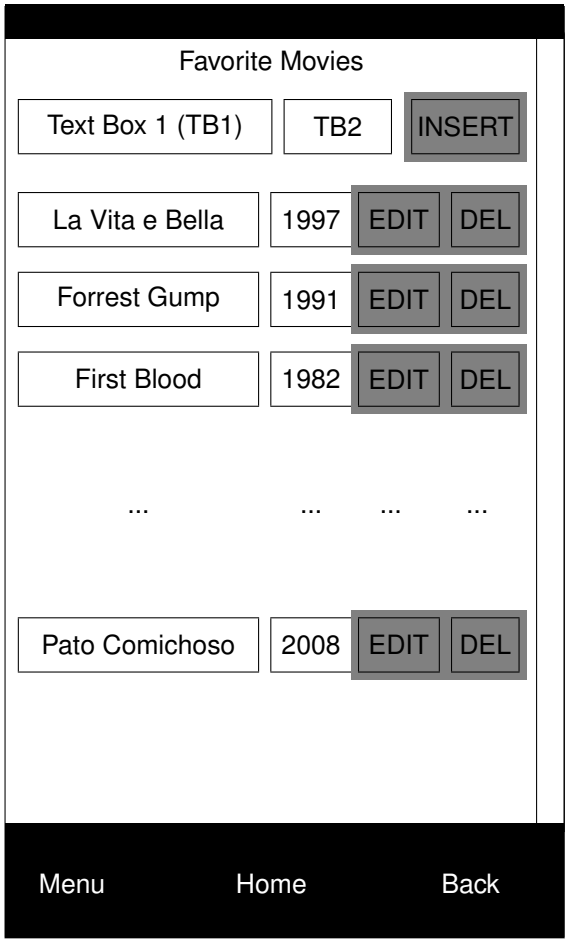
Caso as variáveis já estejam devidamente definidas, passe para a secção seguinte. Caso contrário, precisa de as definir antes de avançar, com comandos semelhantes a:

```
$ set ANDROID_HOME=RAIZ DO SDK
$ set JAVA_HOME=RAIZ do JAVA JDK
$ set PATH=%PATH%;%ANDROID_HOME%\tools;
%ANDROID_HOME%\platform-tools
```

2 Criação de Bases de Dados Locais SQLite

Creation of Local SQLite Databases

Os objetivos deste guia são construir uma aplicação simples para criação e manipulação de uma base de dados local, e ganhar alguma agilidade na depuração de problemas relacionados com bases de dados SQLite. A aplicação a construir deverá não só permitir visualizar os dados de uma base de dados relativa a filmes, como também permitir a inserção, edição e eliminação de registos. No final da execução deste guia, a aplicação desenvolvida deve ter um aspeto semelhante ao que se ilustra na figura seguinte:



Tarefa 3 *Task 3*

Inicialmente, a descrição centra-se na criação da base de dados. Comece por criar um novo projeto Android™ com nome `exStorage2` e com uma só atividade chamada `FavoriteMovies`. O nome do pacote deste projeto deve ser `pt.ubi.di.pmd.exstorage2` e a respetiva raiz deve ficar em `C:\Users\aluno\workspace\exStorage2`.

Tarefa 4 *Task 4*

Crie um novo ficheiro de código Java para implementar o ajudante de criação de bases de dados. Este ficheiro deve ser criado, por exemplo, na diretoria `src\pt\ubi\di\pmd\exstorage2`. O nome da classe que irá ajudar a criar a base de dados será `AjudanteParaAbrirBD`, pelo que o nome do ficheiro deve estar a condizer com a resolução tomada. O conteúdo desse ficheiro deve ser algo semelhante ao seguinte:

```
package pt.ubi.di.pmd.exstorage2;

import android.database.sqlite.SQLiteDatabase;
import android.content.Context;

public class AjudanteParaAbrirBD
  extends SQLiteOpenHelper {
  private static final int DB_VERSION = 1;
  private static final String
          DB_NAME = "FavoriteMovies";
  protected static final String
          TABLE_NAME = "Movie";
  protected static final String COL1 = "id";
  protected static final String COL2 = "name";
  protected static final String COL3 = "year";

  private static final String CREATE_MOVIE =
    "COMPLETAR COM A INSTRUCAO SQL CORRETA";

  public AjudanteParaAbrirBD(Context context) {
    super(context, DB_NAME, null, DB_VERSION);
  }

  @Override
  public void onCreate(SQLiteDatabase db) {
    db.execSQL(CREATE_MOVIE);
  }

  @Override
  public void onUpgrade(SQLiteDatabase db,
    int oldVersion, int newVersion) {
    db.execSQL("DROP TABLE "
              + TABLE_NAME + ";");
    db.execSQL(CREATE_MOVIE);
  }
}
```

Q1.: O que tem a dizer acerca da afirmação seguinte?
É necessário declarar a classe `AjudanteParaAbrirBD` no `AndroidManifest.xml` para que esta possa ser usada por uma componente da aplicação.
☑ Esta afirmação é estapafúrdia.
☐ Esta afirmação está correta.

Note que o código anterior tem uma instrução SQL em falta. Complete-a no seu ficheiro de código (recorra aos valores das *Strings* estáticas) e use o espaço seguinte para a anotar, para referência futura. Quando estiver a construir esta instrução, considere que:

- os nomes dos filmes não terão mais do que 50 ca-

racteres, mas que alguns nomes se podem repetir, inclusive no mesmo ano;

- por causa do que foi dito anteriormente, vai precisar de um `id` para cada filme, que será chave primária da tabela.

Q2.: Já agora, o que significa o S do acrónimo SQL?

Significa Structured.

Q3.: Que nome se dá às instruções SQL usadas assim no contexto de outra linguagem de programação?

☑ *Embedded SQL* ☐ *Included SQL*
☐ *Host Language* ☐ *Ghost SQL*
☐ *SQLlite* ☐ NoSQL

Q4.: Ainda no contexto da questão anterior, que qualificação se dá à linguagem Java?

☐ Linguagem pai. ☐ Linguagem embutida.
☐ Linguagem anfitriã. ☐ Linguagem hotel.
☑ Linguagem hospedeira. ☐ Linguagem convidada.

Q5.: De acordo com o código incluído antes, o que é que acontece à base de dados durante um *upgrade*?

☐ Nada.
☐ A base de dados é destruída e recriada.
☐ A única tabela da base de dados é eliminada e depois recriada. Todos os registos que lá estavam são mantidos.
☑ A única tabela da base de dados é eliminada e depois recriada. Quaisquer registos anteriores presentes na base de dados são perdidos.
☐ A base de dados é reescrita (`@Override`).
☐ A base de dados ganha um bilhete só de ida para o `stderr`.

Tarefa 5 *Task 5*

Depois de compilar de novo o código e de se certificar de que não há erros a tratar, procure entender o seguinte excerto de código. Compare a sua implementação da atividade principal com o código seguinte, e procure completá-la com as instruções que efetivamente criam e abrem a ligação à base de dados:

```
package pt.di.ubi.pmd.exstorage2;

import android.database.sqlite.SQLiteDatabase;
...

public class FavoriteMovies extends Activity {
  private SQLiteDatabase oSQLiteDB;
  private AjudanteParaAbrirBD oAPABD;

  @Override
  protected void onCreate(Bundle state){
    super.onCreate(state);
    setContentView(R.layout.main);
    oAPABD = new AjudanteParaAbrirBD(this);
    oSQLiteDB = oAPABD.getWritableDatabase();
  }
  @Override
  protected void onResume(){
    oSQLiteDB = oAPABD.getWritableDatabase();
  }
  @Override
  protected void onPause(){
    oAPABD.close();
  }
}
```

Q6.: Falta algum *import* em alguma das classes implementadas antes?

☐ Não falta absolutamente nada.
☐ Falta importar `android.database.sqlite.SQLiteOpenHelper` no ficheiro `AjudanteParaAbrirBD.java`
☐ Falta importar `android.database.sqlite.SQLiteOpenHelper` no ficheiro `FavoriteMovies.java`.
☐ Falta importar a classe `AjudanteParaAbrirBD` no ficheiro `FavoriteMovies.java`.

Tarefa 6 *Task 6*

Prepare, instale e execute a aplicação. Use o *Android™ monitor*, nomeadamente o seu explorador de ficheiros, para se assegurar de que a base de dados foi realmente criada. **Q7.: Qual a diretoria onde as bases de dados são criadas?**

data/data/“nome_da_app”/databases/

Q8.: A tipo de armazenamento é que a diretoria que indicou antes pertence?

☐ Armazenamento Android™.
☐ Armazenamento em série.
☐ Armazenamento de bolachas.
☐ Armazenamento para o inverno.
☑ Armazenamento interno.
☐ Armazenamento externo.
☐ Recursos do projeto.
☐ Preferências partilhadas.

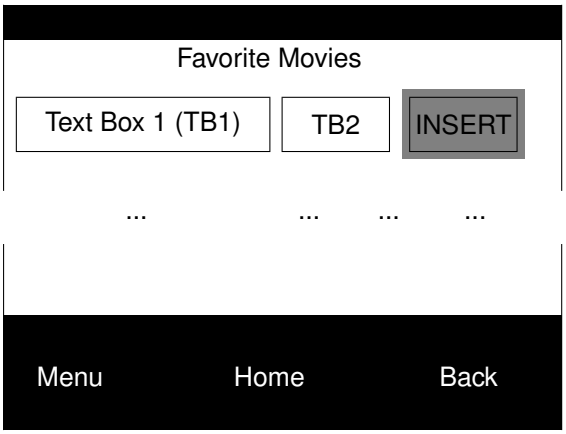
3 Inserção de Dados em Bases de Dados SQLite

Inserting Data in SQLite Databases

A aplicação a desenvolver é minimalista, mas com bastante funcionalidade, porque terá de lidar não só com a exibição dos registos da base de dados, como também com a inserção edição e eliminação de valores. Comece por se focar na secção superior da interface de utilizador, que é a que irá permitir introduzir valores na base de dados.

Tarefa 7 *Task 7*

A parte superior da interface de utilizador da única atividade da aplicação é composta por uma etiqueta de texto centrada numa linha, e por duas caixas de texto e um botão na outra linha, conforme se representa a seguir:



Esta tarefa consiste em editar o ficheiro XML que define o *layout* da atividade de modo a que se pareça com a que foi ilustrada antes. Note que a segunda linha, que mostra as caixas de texto e o botão, deve ocupar toda a largura. Neste caso, terá de utilizar um `ScrollView` para obter o resultado pretendido. Vai também precisar de definir um `ConstraintLayout`:

```
<?xml version="1.0" encoding="utf-8"?>
<ScrollView xmlns:android="http://schemas.android.
    com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-
        auto"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <android.support.constraint.ConstraintLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content">

        <TextView
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:id="@+id/text"
            android:text="Favorite Movies"
            app:layout_constraintLeft_toLeftOf="
                parent"
            app:layout_constraintRight_toRightOf="
                parent"
            android:gravity="center"
            app:layout_constraintBottom_toTopOf="@+
                id/INSERT"
            />

        <EditText
            android:id="@+id/name"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:text="Movie Name"
            app:layout_constraintRight_toLeftOf="@+
                id/year"
            android:layout_constraintTop_toBottomOf=
                "@id/text"
            app:layout_constraintTop_toBottomOf="@id
                /text"
            app:layout_constraintHorizontal_chainStyle
                ="spread"
            app:layout_constraintStart_toStartOf="
                parent"
            app:layout_constraintHorizontal_weight="
                3"
            />

        <EditText
            android:id="@+id/year"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:text="year"
            app:layout_constraintHorizontal_weight="
                1"
            app:layout_constraintLeft_toRightOf="@id
                /name"
            app:layout_constraintRight_toLeftOf="@+
                id/INSERT"
            app:layout_constraintTop_toBottomOf="@id
                /text"
            app:layout_constraintEnd_toStartOf="@+id
                /INSERT"
            app:layout_constraintStart_toEndOf="@id/
                name"/>

        <Button
            android:id="@+id/INSERT"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:onClick="onINSERTclick"
            android:text="ADD"
            app:layout_constraintHorizontal_weight="
                1"
            app:layout_constraintLeft_toRightOf="@id
                /year"
            app:layout_constraintRight_toRightOf="
                parent"
            app:layout_constraintTop_toBottomOf="@id
                /text"
            app:layout_constraintEnd_toEndOf="parent
                "

            />

    </android.support.constraint.ConstraintLayout>

</ScrollView>
```

Q9.: O que pode dizer acerca do tamanho da primeira caixa de texto, em relação aos outros dois objetos interativos da mesma linha, depois de definir o *layout* seguindo as indicações do guia?

- ☐ Que ficou com o mesmo tamanho que as outras duas.
- ☐ Que ficou com o dobro do tamanho das outras duas.
- ☐ Que ficou com o triplo do tamanho das outras duas.
- ☐ Que o largura da segunda linha ficou dividida em 5, sendo 3 desses pedaços são ocupados pela primeira caixa de texto.

Tarefa 8 *Task 8*

Implemente o método `onINSERTclick(View)`, que é despoletado pelo clique no botão `INSERT`. Siga a sugestão providenciada a seguir:

```java
public void onINSERTclick( View v ){
  ContentValues oCV = new ContentValues();
  EditText oED1 = (EditText)findViewById(R.id.name);
  EditText oED2 = (EditText)findViewById(R.id.year);
  oCV.put(oAPABD.COL2, oED1.getText().toString() );
  // FALTA INSTRUCAO
  oSQLiteDB.insert(oAPABD.TABLE_NAME, null ,oCV);
}
```

Q10.: Que classes (adicionais) vai precisar de importar para que o código compile corretamente?

- ☐ `android.app.Activity;`
- ☐ `android.os.Bundle;`
- ☐ `android.database.sqlite.SQLiteDatabase;`
- ☐ `android.view.View;`
- ☐ `android.content.ContentValues;`
- ☐ `android.widget.EditText;`

Tarefa 9 *Task 9*

Compile, instale e teste a aplicação (na medida do possível). Aproveite para inserir um ou dois filmes da sua preferência usando as caixas de texto disponíveis.

Experimente, já agora, colocar uma palavra na caixa de texto referente ao ano, e verifique se a aplicação se queixa ou não desse facto.

4 Depuração da Base de Dados

Debugging the Database

Tarefa 10 *Task 10*

Nota: esta tarefa requer a utilização de um emulador Android™ ou de um dispositivo móvel onde possua privilégios de administração (i.e., acesso `root`).
Como anda não foi implementada forma de mostrar registos da base de dados na atividade (apenas de os inserir), o ideal será recorrer a outros meios para verificar se a aplicação está a funcionar corretamente. Para isso, sugere-se que use a ferramenta `sqlite3`, fornecida com o SDK e nos emuladores Android™. Considere as seguintes questões antes de evoluir para a tarefa em si.

Q11.: Em que diretoria do SDK está o executável que lhe permite abrir uma *shell* SQLite3 na sua máquina?

- ☑ `platform-tools`
- ☐ `tools`
- ☐ `system-images`
- ☐ `build-tools`
- ☐ `sqlite-tools`
- ☐ `ferramentas-back-and-decker`

Q12.: Qual o comando que lhe permite ver quais os dispositivos que estão disponíveis via `adb`?

- ☐ `$ adb list devices`
- ☐ `$ adb please show devices`
- ☑ `$ adb devices`
- ☐ `$ adb mobile phones`
- ☐ `$ abc defg hijklmnop qrstuv wxyz`

Tarefa 11 *Task 11*

Construa o comando `adb` que lhe permite aceder à *shell* do emulador, e emita-o na sua consola (considere que tinha dois dispositivos ligados):

`> adb` -s "nome do dispositivo" shell

Tarefa 12 *Task 12*

Uma vez na *shell*, navegue até à diretoria onde a base de dados deve estar guardada e emita o comando:

```
$ sqlite3 FavoriteMovies
```

Tarefa 13 *Task 13*

A *shell* disponibilizada pelo comando `sqlite3` integra uma série de comandos interessantes. **Q13.: Qual é o comando que permite ver a descrição de todos os outros?**

- ☐ `.aid`
- ☐ `HELP`
- ☐ `SOS`
- ☐ `.description`
- ☑ `.help`
- ☐ `rainbow dash`

Q14.: Agora que já tem forma de ver todos os comandos disponíveis, qual é o que permite ver todas as tabelas contidas na base de dados?

- ☐ `.databases`
- ☐ `SELECT TABLES FROM DATABASE`
- ☑ `.tables`
- ☐ `SHOW TABLES`
- ☐ `.show`

Q15.: Verificou se a tabela `Movies` foi criada, de facto, na base de dados em análise?

- ☑ Verifiquei e sim, existe!
- ☐ Verifiquei e não, não existe!
- ☐ Não verifiquei e sim, existe!
- ☐ Não verifiquei e sim, não existe!
- ☐ Não verifiquei e não, existe!

Engendre e emita a instrução que lhe permite verificar se existe registos na base de dados, nomeadamente aqueles que inseriu via aplicação. Use o espaço seguinte para guardar a instrução, para referência futura:

SELECT * FROM Movies;

Tarefa 14 *Task 14*

O objetivo desta tarefa é o de ficar a conhecer um pouco melhor o SQLite. Para isso, considere os seguintes desafios e questões.

Os autores do SQLite definem-no da seguinte forma: *SQLite is a software library that implements a **self-contained**, **serverless**, **zero-configuration**, **transactional** SQL database engine. SQLite is the most widely deployed SQL database engine in the world. The source code for SQLite is in the public domain.*

Procure saber os significados das palavras realçadas a negrito no excerto anterior.

self-contained: não tem dependências externas

serverless: não tem servidor (não é uma arquitetura cliente-servidor)

zero-configuration: não precisa de ser configurado

transactional: usa transações

Antes de continuar, responda às seguintes questões:

Q16.: Em que linguagem foi escrito o SQLite?

Em linguagem C

Q17.: Em quantos ficheiros distintos se encontra a implementação do SQLite, e qual a razão principal que levou os seus programadores a fazer a implementação desta forma?

Está implementado em 1 ficheiros, e foi assim implementado para que pudesse ser embutido noutros projetos

Q18.: O SQLite permite acesso concorrente?

- ☐ Essa é boa! Então depois de discutirmos a simplicidade do sistema e o fato de ser *serverless*, ainda queremos que permita acesso concorrente à mesma base de dados por várias aplicações?
- ☑ Permite.

Q19.: Qual das seguintes opções concretiza uma situação para a qual o SQLite não garante as propriedades ACID de uma transação?

- ☐ Uma falha do programa;
- ☐ Uma falha do sistema operativo;
- ☐ Uma falha de energia.

Q20.: Como se sai do SQLite?

.exit ou .quit

Q21.: É possível obter as instruções que definem a criação das tabelas?

- ☐ Não, não é.
- ☐ Sim é, nomeadamente através do comando `.tables`.
- ☑ Sim é, nomeadamente através do comando `.schema`.
- ☐ Sim é, nomeadamente através do comando `.creates`.
- ☐ Sim é, nomeadamente através do comando `.description`.

5 Listagem e Remoção de Registos

Listing and Deleting Records

Tal como ilustrado na primeira figura que define o *layout* da aplicação, a parte central da atividade é composta por um conjunto de *widgets* que se repete tanta vezes quantas necessário para mostrar toda a informação contida na tabela `Movie`. Este conjunto vai repetir-se um número de vezes igual ao número de linhas na tabela, eventualmente ultrapassando o limite vertical do ecrã para mostrar informação. Essa possibilidade vai ter de ser, assim, acomodada.

Tarefa 15 *Task 15*

A próxima tarefa consiste em adicionar uma nova *View* ao ficheiro de *layout* com o conteúdo sugerido a seguir (adicione o excerto de código seguinte na última linha do `ConstraintLayout` que já tem definido no XML, i.e., imediatamente antes do fecho da *tag* `</ConstraintLayout>` existente):

```
<ScrollView
  android:layout_width="fill_parent"
  android:layout_height="fill_parent"
  >
  <android.support.constraint.ConstraintLayout xmlns
      :android="http://schemas.android.com/apk/res/
      android"
    xmlns:app="http://schemas.android.com/apk/res-
        auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:orientation="vertical"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:id="@+id/llsv"
  >
  </android.support.constraint.ConstraintLayout>
</ScrollView>
```

Q22.: O que é que faz uma `ScrollView`?

- ☐ Uma `ScrollView` é um contentor especial, que oferece a funcionalidade de *scrolling* sempre que o seu conteúdo ultrapassa os limites do ecrã sejam ultrapassados.
- ☐ Uma `ScrollView` é um contentor igual aos outros, que pode ser usado para fazer a separação de resíduos.

☑ Uma `ScrollView` é um contentor como os outros, que oferece a funcionalidade de *scrolling* sempre que o seu conteúdo ultrapassa os limites do ecrã sejam ultrapassados.

Q23.: Nota algum detalhe estranho no excerto de código adicionado?

☐ Não.

☑ Sim, o `ConstraintLayout` interno não contém qualquer objeto.

☐ Sim, o `ScrollView` não pode conter um `ConstraintLayout`.

Q24.: Pode colocar objetos interativos, i.e., *widgets*, (e.g., um botão) diretamente dentro de um contentor do tipo `ScrollView`?

☐ Sim, sem problemas.

☑ De facto, a documentação parece sugerir que não, que dentro de um `ScrollView` só pode ser colocado outro objeto do tipo *contentor*.

Tarefa 16 *Task 16*

Vai ser preciso criar um *layout* para cada uma das linhas que apresenta a informação para os filmes na base de dados. Cada uma destas linhas é composta por duas caixas de texto e dois botões (EDIT e DEL), dispostos na horizontal. A primeira caixa de texto, por se referir ao nome do filme, deve ocupar o dobro do que ocupa cada um dos outros objetos. Para manter a complexidade baixa, propõe-se que crie um novo ficheiro de *layout* só para definir cada uma das linhas mencionadas. O ficheiro deve chamar-se `line.xml`, e deve ser colocado na mesma diretoria que o `main.xml`. O seu conteúdo deve ser o que se apresenta a seguir:

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/
    res/android"
    xmlns:app="http://schemas.android.com/apk/res-
        auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:orientation="horizontal"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <EditText
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:id="@+id/ED1"
        app:layout_constraintHorizontal_chainStyle="
            spread"
        app:layout_constraintStart_toStartOf="parent
            "
        app:layout_constraintHorizontal_weight="2"
        app:layout_constraintRight_toLeftOf="@+id/
            ED2"
        />
    <EditText
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:id="@+id/ED2"
        app:layout_constraintLeft_toRightOf="@id/ED1
            "
        app:layout_constraintHorizontal_weight="1"
        app:layout_constraintEnd_toStartOf="@+id/
            EDIT"
        app:layout_constraintStart_toEndOf="@id/ED1"
        />
    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="EDIT"
        android:id="@+id/EDIT"
        android:onClick="onEDITclick"
        app:layout_constraintLeft_toRightOf="@id/ED2
            "
        app:layout_constraintHorizontal_weight="1"
        app:layout_constraintEnd_toStartOf="@+id/DEL
            "
        app:layout_constraintStart_toEndOf="@id/ED2"
        />
    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        app:layout_constraintHorizontal_weight="1"
        android:text="DEL"
        android:id="@+id/DEL"
        android:onClick="onDELclick"
        app:layout_constraintLeft_toRightOf="@id/
            EDIT"
        app:layout_constraintEnd_toEndOf="parent"
        />
</android.support.constraint.ConstraintLayout>
```

Note que cada um dos *widgets* do *layout* anterior tem um ID definido, e que os últimos dois botões especificam métodos de *callback*.

Tarefa 17 *Task 17*

Compile o projeto, instale e teste a aplicação antes de prosseguir.

Tarefa 18 *Task 18*

A implementação da parte de listagem dos resultados de uma consulta à base de dados é uma das mais exigentes que já foi tentada nas aulas práticas, talvez não pela sua complexidade, mas por ser necessária a adição de objetos interativos programaticamente, que ainda não foi tentada antes. O racional elabora nos seguintes passos:

1. Faz-se uma consulta (`SELECT`) à base de dados para obter todos os registos;
2. Para cada linha devolvido pelo resultado, adiciona-se um novo conjunto de *widgets* ao `ConstraintLayout` que está vazio dentro da `ScrollView`.

Para ajudar neste processo, sugere-se uma abordagem faseada, em que o código é dado gradualmente.

Comece por **adicionar** as duas instruções seguintes ao método `onCreate()` da sua atividade:

```
ConstraintLayout oLL = (ConstraintLayout)
    findViewById(R.id.llsv);
Cursor oCursor = oSQLiteDB.query(oAPABD.TABLE_NAME,
    new String[]{"*"} ,null, null, null, null,
    null);
```

Use o espaço seguinte para indicar os *imports* que também precisa adicionar ao seu código para poder compilar o projeto com sucesso:

Procure agora explicar o que faz cada uma das linhas anteriores:

Linha 1:

Linha 2:

Como medida de precaução, pode considerar compilar o projeto, para saber se não foram introduzidos erros ou se falta algum *import*.

Tarefa 19 *Task 19*

Adicione agora, e após as duas linhas anteriores, o excerto de código seguinte:

```
boolean bCarryOn = oCursor.moveToFirst();
while( bCarryOn ){
  // FALTA CODIGO
  bCarryOn = oCursor.moveToNext();
}
```

Note que faltam instruções no excerto anterior, mas deve conseguir compilar o projeto de qualquer forma. **Q25.: O que faz o excerto de código anterior?**

- ☐ Coloca o `Cursor` a apontar para a linha imediatamente antes da primeira, movendo-o de seguida para a primeira linha.
- ☑ Coloca o `Cursor` a apontar para a primeira linha, e itera-o até chegar ao fim da tabela de resultados.
- ☐ O código contém um erro, e coloca a aplicação num *loop* interminável.

Q26.: O que é que devolve o método `moveToNext()`?

- ☑ Verdadeiro caso tenha conseguido mover-se para a próxima linha, e falso caso contrário;
- ☐ Verdadeiro caso tenha chegado ao fim da tabela, e falso caso contrário;
- ☐ Falso caso tenha chegado ao fim da tabela, e verdadeiro caso contrário.

Tarefa 20 *Task 20*

Para terminar, e no local onde falta o código no ciclo `While` anterior, considere colocar agora as seguintes linhas de código Java, respondendo depois às questões que se lhe seguem:

```
ConstraintLayout oLL1 =
    (ConstraintLayout) getLayoutInflater()
    .inflate(R.layout.line, null);
oLL1.setId(oCursor.getInt(0)*10+4);

EditText oED1 =
    (EditText) oLL1.findViewById(R.id.ED1);
oED1.setId(oCursor.getInt(0)*10+2);
oED1.setText(oCursor.getString(1));

EditText oED2 =
    (EditText) oLL1.findViewById(R.id.ED2);
oED2.setId(oCursor.getInt(0)*10+3);
oED2.setText(oCursor.getInt(2)+"");

Button oB1 =
    (Button) oLL1.findViewById(R.id.EDIT);
oB1.setId(oCursor.getInt(0)*10+1);

Button oB2 =
    (Button) oLL1.findViewById(R.id.DEL);
oB2.setId(oCursor.getInt(0)*10);

oLL.addView(oLL1);
```

Q27.: O que faz o método `inflate()`, presente na primeira linha de código do excerto anterior?

- ☐ Incha o `ConstraintLayout` até este não caber mais no ecrã, ficando a rebentar pelas costuras.
- ☑ Lê o conteúdo do recurso dado por `R.layout.line`, convertendo-o para um objeto da aplicação.
- ☐ Lê o conteúdo do recurso dado por `line.xml`, convertendo-o para um objeto da aplicação.

Q28.: O que fazem todas as linhas de código que contêm o método `setId(.)`?

- ☐ Na verdade mudam o identificador de cada objeto interativo a que se referem, de maneira a refletir a chave primária da tabela *Movie*.
- ☐ Na verdade deixam o identificador de cada objeto interativo a que se referem tal como estava.
- ☐ Na verdade colocam todos os identificadores dos objetos a que se referem com o mesmo valor, i.e., o valor da chave primária da linha atual.

Considere que, a determinado momento, o cursor `oCursor` estava sobre a linha com `id` igual a 5. **Q29.: Qual o ID atribuído a cada um dos objetos interativos representados?**

`oLL1`

`oED1`

`oED2`

`oB1`

`oB2`

Repita o exercício anterior, mas agora para quando o cursor `oCursor` está sobre a linha com `id` igual a 12:

`oLL1`

`oED1`

oED2 _____

oB1 _____

oB2 _____

Q30.: Considera que o esquema que o Prof. engendrou para os IDs dos *widgets* é efetivo?

☐ E como é que podia dizer o contrário, tratando-se do Prof.?

☐ De facto, este esquema garante que todos os *widgets* ficam com um ID único na aplicação!

☐ Não, porque há situações em que os IDs se podem repetir, nomeadamente:

_____

_____

Q31.: O que faz a última linha de código do excerto anterior (i.e., `oLL.addView(oLL1);`)?

☐ Elimina todos os *widgets* redefinidos dentro do `while` ao `ConstraintLayout` que estava (inicialmente) vazio.

☐ Coloca o conteúdo do `oCursor` nas caixas de texto.

☑ Adiciona todos os *widgets* redefinidos dentro do `while` ao `ConstraintLayout` que estava (inicialmente) vazio.

☐ Adiciona um novo objeto do tipo `View` à aplicação.

Tarefa 21 *Task 21*

Compile o projeto e instale a aplicação. Adicione vários filmes à base de dados através da funcionalidade que já havia implementado numa parte anterior deste guia laboratorial, ou através da *shell* SQLite3. Entre e saia da aplicação várias vezes entre inserções à base de dados, para estimar se a funcionalidade antes implementada está a funcionar ou não.

Tarefa 22 *Task 22*

Foque-se agora na funcionalidade associada ao botão DEL. Para eliminar algo de uma base de dados SQLite, basta fazer uso do método `delete()`, que aceita o nome da tabela, a String que define a cláusula `WHERE` e, eventualmente, os parâmetros de entrada desta última. Para o ajudar nesta tarefa, sugere-se a utilização do seguinte excerto de código, onde falta apenas um parâmetro, que deverá preencher:

```
public void onDELclick ( View v ){
  oSQLiteDB.delete(
    // FALTA PARAMETRO,
    "ROWID="+v.getId()/10,
    null);

  ConstraintLayout oLL1 =
     (ConstraintLayout) findViewById(v.getId()+4);
  ((ConstraintLayout) oLL1.getParent()).removeView(
      oLL1);
}
```

Q32.: O que fazem as últimas duas linhas de código do excerto de anterior?

☐ A primeira linha cria uma instância do contentor a retirar e a segunda linha retira-o do seu elemento pai.

☐ A primeira linha cria uma instância do contentor que contém o elemento a retirar e a segunda linha retira-o do seu elemento pai.

☐ A primeira linha cria um filho, a segunda cria uma filha, e não se entendem.

Tarefa 23 *Task 23*

Prepare, instale e teste a aplicação.

6 Edição de Registos

Editing Records

Tarefa 24 *Task 24*

Note que a funcionalidade de edição ainda não foi implementada, mas deve elaborar num racional semelhante ao que já foi feito para a eliminação e inserção de registos. A tarefa final deste guia consiste então na codificação da parte em falta, nomeadamente do método *callback* `onEDITclick(View)`.