Aplicação para Dispositivos Móveis para Gestão de Contactos

Contactos numa base de dados SQLite

Application for Mobile Devices for Contact Management

Contacts in a SQLite database

Cristiana Lopes Gabriel
Escola Superior de Tecnologia e Gestão
Instituto Politécnico da Guarda
Guarda, Portugal
cristianalg7@hotmail.com
GitHub: https://github.com/cristianalg/MCM_Contactos

Resumo — **Este documento descreve o projeto desenvolvido no âmbito da Unidade Curricular Programação Avançada para Dispositivos Móveis, do curso de Mestrado de Computação Móvel, da Escola Superior de Tecnologia e Gestão, do Instituto Politécnico da Guarda. O projeto desenvolvido consiste numa aplicação para dispositivos móveis *Android*, dirigido ao utilizador comum, para gestão de contactos telefónicos. A aplicação foi desenvolvida utilizando como linguagem de programação o Java com recurso à base de dados *SQLite*.**

Palavras Chave - aplicação; dispositivos móveis; Java; SQLite.

Abstract — **This document describes the project developed in the scope of the Advanced Programming for Mobile Devices Course, of the Master's Degree in Mobile Computing, of the School of Technology and Management, of the Polytechnic Institute of Guarda. The project is an application for Android mobile devices, aimed at the ordinary user, for managing telephone contacts. The application was developed using Java as programming language using the *SQLite* database.**

Keywords - application; mobile devices; Java; SQLite

I. INTRODUÇÃO

A aplicação para gestão de contactos permite ao utilizador inserir um novo contacto, bem como, ver os detalhes, editar e eliminar. Esta aplicação usa como base de dados *SQLite*, pela razão de que o *Android* tem suporte nativo para isso, os dados serão armazenados localmente no dispositivo do utilizador.

Este artigo mostra como configurar *layouts* para um simples catálogo de contactos para dispositivos *Android*. Como usar vários *widgets*, componentes e *layouts*, ou seja, *LinearLayout*, *RelativeLayout*, *TableLayout*, *ScrollView*, *ListView*, *TextView*, *ImageView*, *Button* e *EditText*. Além de abordar como configurar a aparência de um *ListView item*.

A *Tabela 1* descreve sucintamente os tipos de *layouts* e components apresentados.

Layout e Componentes	Descrição
LinearLayout	Alinha todas as views numa única direção, vertical ou horizontalmente.
RelativeLayout	Mostra todas as suas views em posições relativas.
TableLayout	Agrupa as views em linhas e colunas.
ListView	Exibe uma lista de items com suporte de scrolling.
TextView	É a view preferencial para apresentação de texto.
ImageView	É a view para exibir imagens.
Button	É uma view que permite ao utilizador desencadear uma determinada ação programada.
EditText	É a view preferencial para entrada de texto na aplicação.

Tabela 1 - Lista de layouts e componentes disponibilizados pelo SDK do Android

A base de dados é constituída por uma tabela designada de "*contacts*" com a estrutura que se pode ver na *Tabela 2*.

Columm name	Type	Key
id	INTEGER	PRIMARY
name	TEXT	
phone	INTEGER	
email	TEXT	
address	TEXT	
photograph	BLOB	

Tabela 2 - Estrutura da tabela Contactos

II. TRABALHO RELACIONADO

Existem atualmente algumas aplicações de gestão de contactos, para além da gestão de contactos que o *Android* nos fornece ao adquirir um smartphone *Android*. Descrevem-se a seguir aplicações semelhantes à aplicação desenvolvida em âmbito da unidade curricular.

A '*DW Contacts & Phone Dialer*' tem uma versão paga e outra gratuita e funciona como um excelente substituto à lista de contactos e marcador de chamadas. Pode ser usado para criar grupos e sub-grupos de contactos, enviar mensagens em série, pesquisar facilmente pelo nome e visualizar o histórico de contactos.[4]

A '*PureContact*' não foi desenvolvido para gerir uma lista grande contactos, mas sim para personalizar um grupo pequeno de contactos usados frequentemente tornando-os mais acessíveis. Funciona como um marcador de chamadas rápidas que permite realizar várias ações nos contactos. Em que se pode optar por efetuar chamadas, enviar SMS, emails e até mensagens do WhatsApp.[4]

A '*Contacts +*' consegue organizar os contactos de forma completa e eficaz além de sincronizar as contas das redes sociais com as suas comunicações. Consegue também obter e sincronizar as fotos do Facebook e Google+ com a sua lista de contactos. Além disso, permite a visualizar os contactos e publicar nas redes sociais convenientemente na própria aplicação do *Contacts +*.[5]

III. SISTEMA DESENVOLVIDO

A. *Arquitetura do Sistema Desenvolvido*

A arquitetura pode ser dividida em três componentes principais:

- Utilizador;
- Mobile app, aplicação *Android* desenvolvida em *Java*.
- Base de dados *SQLite*.

Figura 1 - Arquitetura do sistema

B. *Descrição dos Módulos do Sistema*

1) *Utilizador*

O módulo utilizador, é o responsável pela interação entre o utilizador comum e a aplicação mobile.

2) *Mobile app*

A aplicação é destinada à plataforma *Android* desenvolvida em *Java*.

- *Android* é um sistema operativo baseado no núcleo Linux e atualmente desenvolvido pela empresa de tecnologia *Google*. Com uma interface de utilizador baseada na manipulação direta.[1]

- *Java* é uma linguagem de programação orientada a objetos.[2]

3) *Base de dados SQLite*

Os dados relativos a um contacto são armazenados numa base de dados *SQLite*.

O sistema *Android* suporta base de dados *SQLite*. O *SQLite* é uma pequena biblioteca open-source, que implementa um amplo subconjuntos do standard *SQL 92*. [3]

O uso de uma base de dados numa aplicação *Android* restringe o acesso à aplicação que a criou, ou seja, a base de dados poderá ser acedida apenas pelas classes da aplicação.

Para manipular a base de dados *SQLite*, usou-se a classe *SQLiteOpenHelper*.

C. Protótipo do Sistema

No desenvolvimento da aplicação, tentou-se sempre procurar formas mais eficazes e simples de realizar o pretendido, de modo a traduzir numa maior eficácia e rapidez da execução das tarefas. E é essa a missão de um programador para tornar o produto final o mais simples e fácil de utilizar, não esquecendo a rapidez de execução da aplicação.

A *Figura 2* apresenta o interface principal da aplicação.

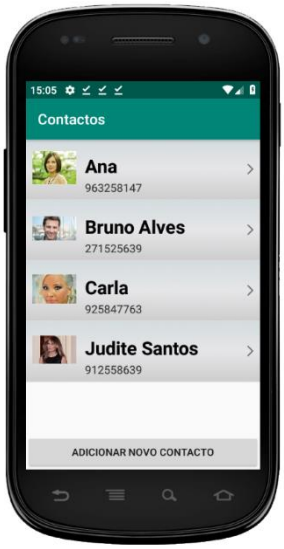

Figura 2 - Ecrã principal da aplicação

Ao executar a aplicação podemos ver toda a lista de contactos, através da atividade principal *Figura 2*. Para listar todos os contactos foi criado um adaptador personalizado para o *ListView*. Cada contacto é mostrado num *Item ListView*. A *Figura 3* apresenta o formato do *Item ListView* e os seus componentes usados para o efeito.

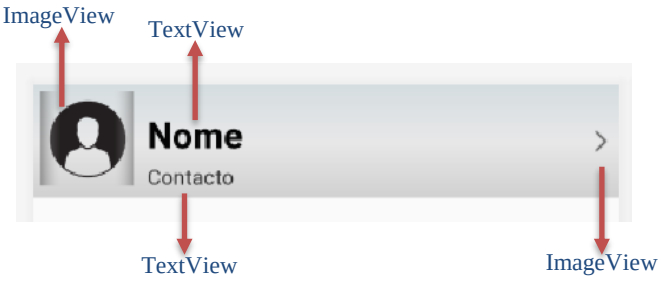

Figura 3 - Aparência do Item ListView personalizado

Ao clicar sobre um contacto é possível ver em detalhe os dados referentes ao mesmo. Para distinguir o contacto que está a ser selecionado foi implementado um selector personalizado para dar efeito de destaque, a azul, como se pode ser na *Figura 4*.

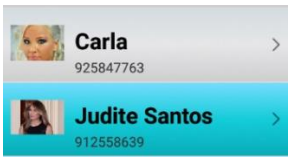

Figura 4 - Efeito do seletor personalizado

Para adicionar um contacto é através do botão *"Adicionar Novo Contacto"* na atividade principal. A atividade de adicionar novo contacto tem o aspeto apresentado na *Figura 5*.

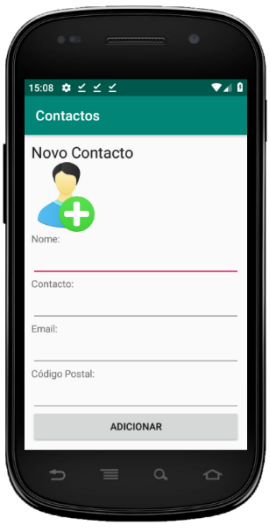

Figura 5 - Ecrã para adicionar um novo contacto

Como referido anteriormente, é possível ver em detalhe os dados de um contacto ao clicar sobre um contacto.

Na atividade de detalhes de um contacto encontra-se um menu com duas opções, a de editar e eliminar.

A *Figura 6* demonstra a atividade visualizar os detalhes de um contacto.

Figura 6 - Ecrã para visualizar dos detalhes de um contacto

Para editar um contacto é usado o mesmo layout da atividade de *"Adicionar um Novo Contacto"*. Para tal, foi criada uma classe de edição que carrega todos os dados que vêm da atividade *"Detalhes do contacto"* na atividade de novo contacto, como pode ver na *Figura 7*.

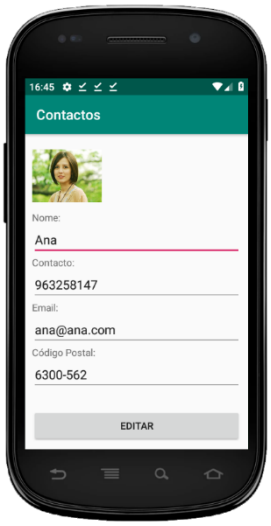

Figura 7 - Ecrã para Editar um contacto

Para eliminar um contacto basta clicar na opção *"Eliminar"* do menu (*Figura 6*), e o mesmo será eliminado.

IV. Testes e Resultados

Os testes realizados até ao momento testaram a comunicação e interoperabilidade entre os diferentes componentes da arquitetura implementada.

Durante toda a fase de desenvolvimento da aplicação, foram realizados testes graduais com o objetivo de validar todas as funcionalidades que iam sendo desenvolvidas e assim testar o funcionamento de todas as condições que eram impostas nas funcionalidades.

V. Conclusões

O projeto descrito, neste artigo, acompanhou a construção de uma aplicação *Android* para gestão de contactos.

O presente trabalho permitiu um aumento de conhecimentos relativos ao mundo do *Android*, assim como uma melhor implementação do código tendo por base o que se aprendeu ao longo do semestre e ainda o conhecimento de alguns componentes que até ao momento eram pouco conhecidos pelo aluno. Em suma o desenvolvimento deste trabalho revelou ser bastante enriquecedor.

Como trabalho futuro aponta-se a possibilidade de melhorar a aplicação implementando novas funcionalidades. Tais como criar grupos de contactos para o envio de mensagens em série e criar um marcador de chamadas rápidas que permita realizar várias ações nos contactos, até mesmo sincronizar os contactos com as contas das redes sociais.

Referências Bibliográfica

[1] Google. (s.d.). *Official Blog Android*. Obtido de https://blog.google/products/android/

[2] Oracle. (s.d.). *Java*. Obtido de https://www.java.com

[3] SQLite. (s.d.). Obtido de https://www.sqlite.org

[4] wondershare. (s.d.). *dr.fone*. Obtido de https://drfone.wondershare.com/contacts/best-android-contacts-app.html

[5] Play, G. (s.d.). *Contacts+*. Obtido de https://play.google.com/store/apps/details?id=com.contapps.android&hl=pt_PT