Relatório de Trabalho Final de Curso

Licenciatura em Engenharia Informática
2013

Tema:

Programa Descontos Santander Totta

Aplicação de descontos baseada na localização para dispositivos móveis

Trabalho realizado por:
Pedro Miguel Serrudo da Silva - 20085712

Orientado por:
Prof. Adriano Couto

Índice

1- Resumo

O objectivo principal deste projecto prende-se em criar uma aplicação para dispositivos móveis com sistema operativo Android que utilizando o sistema de localização dos dispositivos lista e mapeia uma série de ofertas perto do utilizador, essas ofertas são obtidas através de uma web API abordada mais a frente.

Este projecto foi produzido para uma entidade externa de nome Dunne Premium Technologies, ao qual fiquei encarregue de todo o planeamento e desenvolvimento da aplicação em causa. Para o desenvolvimento da aplicação foi utilizado o *IDE* Eclipse com as ferramentas de desenvolvimento para Android, a linguagem de programação utilizada foi o Java.

2- Abstract

The main goal of this project is to develop an application for mobile devices with Android operating system, using the location of the device, application can list and map a deals near the user, these deals are retrieved through a web API discussed later.

This project was produced for an external entity named Dunne Premium Technologies, where I was in charge of all the planning and development of the application. For application development, I used the Eclipse IDE with the Android development tools, programming language used was Java.

3- Introdução

O projecto consiste em desenvolver uma aplicação para dispositivos móveis *Android* com target mínimo à versão 2.2, que têm como base uma web *API* com vários serviços ao qual faz constantes pedidos para mostrar as ofertas ou submeter dados. Essa API está protegida por autenticação para garantir uma maior segurança na troca de informações.

3.1- Como surgiu

Este projecto surgiu de uma conceito surgido pelo banco Santander Totta, que consistia em ligar os vários tipos de clientes do banco, particulares e empresariais, para que dessa forma os clientes empresariais pudessem colocar ao dispôr os seus serviços com descontos. Foi-lhe atribuído o nome de "Programa Descontos Santander Totta".

3.2- Como funciona

Através da aplicação móvel o utilizador poderá ter conhecimento das ofertas que estão próximas do local onde se encontra, poderá também filtrar as ofertas por categorias, marcar os seus descontos favoritos, subscrever Newsletter e ainda consultar informações de apoio. Caso tenha alguma dúvida relativamente às ofertas disponíveis, poderá contactar directamente a loja aderente e/ou efectuar uma reserva ou marcação.

3.3- Público-alvo

Foi criado a pensar no dia-a-dia e tempo livre das pessoas e as suas famílias, sendo que de uma forma fácil e cómoda oferece descontos imediatos que permitem poupar em lojas aderentes, essas lojas agrupam-se nas categorias bem-estar, comercio/serviços, lazer e restaurantes.

3.4- Vantagens

Com o Programa Descontos Santander Totta o utilizador tem acesso a descontos imediatos no ponto de venda em comércio de bens e serviços. Não é necessária qualquer adesão prévia ao programa, o utilizador apenas terá de apresentar o seu cartão do Banco Santander Totta e proceder ao pagamento das compras nas lojas aderentes.

4- Enquadramento teórico

4.1- Android

Na cadeira de Sistema Embebidos foram abordadas questões fundamentais para o desenvolvimento de aplicações Android, essas bases serviram para aumentar o meu interesse por este tipo de aplicações, até porque, nos tempos que decorrem os telemóveis/smartphones são um companheiro de bolso de qualquer pessoa e com tanta afluência acabaram por disputar um grande uma reviravolta no mercado tecnológico, alguns dispositivos mais avançados até rivalizam com os computadores de secretária. Após todos estes estímulos tecnológicos, acabei por aumentar a minha experiência na área e candidatei-me a vários trabalhos, num dos quais surgiu a aplicação referida neste relatório.

4.2- E-Commerce (Comércio Electrónico)

O comércio é uma actividade social e económica associada ao ser humano praticamente desde a sua existência. Porém, se reflectirmos acerca da actividade do comércio electrónico, facilmente se conclui que este é um caso particular que o Homem mais recentemente adoptou, a partir do comércio tradicional (não electrónico).

Entende-se por comércio electrónico qualquer sistema tecnológico e económico que potencia ou facilita a actividade comercial de um conjunto variado de participantes e que inclua o suporte à generalidade das próprias transacções comerciais.

Existem três modelos principais de comércio electrónico na internet, que são:

- Comércio entre empresas (Business to Business – B2B).
- Comércio entre empresas e consumidores ou comércio de retalho (Business to Consumer – B2C).
- Comércio entre consumidores (Consumer to Consumer – C2C)

O projecto desenvolvido insere-se no modelo comércio entre empresas e consumidores.

5- API

Com já foi abordado na introdução, a aplicação tem como base uma web API, todas as respostas da API são em formato *JSON*. Os pedidos à *API* serão feitos por HTTP o que requer uma ligação de internet activa, pelo método *POST,* igualmente em formato JSON, em que incluirá sempre os dados de autenticação e possivelmente os parâmetros extra, esses dados de autenticação são enviados codificado em Base64.

5.1-Detalhes da API

A *API* dispõe de vários serviços necessários para o funcionamento da aplicação tais como:

- **Listar ofertas** – *path* -> "http://…/deals/list/"
  - Este serviço é o principal para o funcionamento da aplicação, como entrada recebe as coordenadas do utilizador e retorna uma resposta com todas as ofertas à sua volta ordenada por distância
- **Detalhe da oferta** – *path*-> "http://…/deals/detail/"
  - Neste serviço a aplicação envia um pedido com o ID de uma determinada oferta e recebe a informação relativa a essa oferta mais detalhada
- **Registo de Newsletter** - *Path*-> "http://…/users/newsletter/"
  - Neste serviço a aplicação envia um pedido com o endereço de correio electrónico, fornecido pelo utilizador, e será adicionada a lista de contactos para Newsletter, retorna uma resposta de sucesso se foi adicionado com sucesso
- **Enviar um contacto** - *path* -> "http://…/users/contact/"
  - Neste serviço o aplicação envia um pedido com o endereço de correio electrónico e um campo de texto (observações)
- **Top Visualizações** - *path* -> "http://…/deals/top/
  - Como entrada será enviado um algarismo, que designa o número máximo de ofertas que o serviço irá retornar, será usado 10 para o Top 10

6- Introdução ao *Android*

A plataforma do Google *Android* vem ganhando cada vez mais força no mercado de aplicativos móveis, e não é à toa. O SDK vem amadurecendo, já fez 5 anos de mercado, existindo cada vez mais um maior leque de dispositivos *Android* disponíveis.

As vantagens do Android não são apenas aos utilizadores dos dispositivos móveis, ele disponibiliza um rico *framework* para o *developer*. Até há bem pouco tempo o *IDE* de eleição para desenvolvimento seria o eclipse mas em Março a Google lançou o seu próprio *IDE*, o Android Studio, que traz o melhor do Java para seu desenvolvimento que somadas as vantagens do uso do *XML* para o *design* e textos (ideal para questões de internacionalização), torna o desenvolvimento de aplicativos *Android* uma tarefa confortável para o *developer*.

Alguns conceitos importantes para a construção de uma aplicação em Android serão abordados neste capítulo, tais como as ditas *Activity's*, *Intent's* e o *Android Manifest*.

6.1- Activity

Activity está relacionado com a tarefa que uma aplicação pode fazer. Essa tarefa pode ser, por exemplo, um o ecrã de boas vindas, um mapa, uma lista, um ecrã de opções… algo que possa ser apresentado ao utilizador. As *Activity's* são o componente chave do *Android*, para existir interacção com a interface uma classe deve herdar uma *Activity* ou outra classe que herde, no caso dos mapas herda *MapActivity* que por sua vez herda Activity. Um exemplo de uma *Activity* seria:

```
public class TottaActivity extends Activity{
   @Override
   public void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.main); // define o layout
   // todo o codigo relativo a este layout é definido ou invocado aqui
   }
}
```

6.2- Intent

Intent refere-se às intenções. São determinados comandos que podemos enviar ao Sistema Operativo para realizar uma acção e podem ser enviados dados adicionais ou podemos trabalhar os dados que recebemos após a *Intent* ter concluído a acção. Um exemplo de uma *Intent* seria:

```
Intent intent = new Intent(Intent.ACTION_SEND);
intent.setType("text/html");
intent.putExtra(Intent.EXTRA_SUBJECT, "Programa Descontos");
intent.putExtra(Intent.EXTRA_TEXT, "Optenha 50% de desconto em xpto");
startActivity(Intent.createChooser(intent, "Send Email"));
@Override
public void onActivityResult(int reqCode, int resultCode, Intent data) {
 if (resultCode == Activity.RESULT_OK && reqCode == 0) {
     // email enviado com sucesso
 }
}
```

Esta *Intent* pede ao sistema operativo para iniciar uma aplicação de envio de correio electrónico. Este exemplo, é real, utilizado na aplicação para a partilha de uma oferta por correio electrónico.

6.3- Android Manifest

AndroidManifest.xml é o ficheiro que mapeia todo o projecto, descreve todos os componentes da aplicação, define os nomes para as *activity's* (detalhado em 7.1), modos de orientação do ecrã, declaração das permissões para acesso a recursos dos SO e/ou Hardware (detalhado em 7.2), lista das bibliotecas que a aplicação irá usar (detalhado em 7.3), qual a primeira *activity* a ser lançada, entre outras configurações.

7- Estrutura da aplicação

Neste capítulo é abordado as questões chave da aplicação.

7.1- Activity's

A aplicação está dividida em 5 *Activity's*. Cada uma dessas actividades tem pelo menos 1 *layout* associado. Todas as acções que o utilizador pode realizar estão definidas em 8.1, diagrama de casos de uso.

- ***Splash Screen Activity*** – é a primeira *activity* a ser lançada, mostra a imagem inicial com o logo da aplicação verifica o estado das ligações de dados e se alguma estiver activa faz o primeiro pedido a *API,* para obter a lista de todas as ofertas perto do utilizador.
- **Main *Activity*** – é a *Activity* principal que surge a seguir à *Splash Screen Activity* e dispõe das seguintes acções:
  - Lista todas as ofertas ordenadas por distância.
  - Mapeia as ofertas e a posição do utilizador.
  - Filtra as ofertas por 4 categorias.
  - Ao clicar numa oferta será iniciada a *Activity* detalhe, onde é passado o ID da mesma.
  - Abrir a área pessoal.
- **Detalhe Activity** – recebe o *ID* de uma oferta e faz o pedido dos detalhes à API, mostra toda a informação da oferta e tendo as seguintes acções:
  - Partilhar a oferta, onde inicia a *Activity* de partilha.
  - Contactar o comerciante através de num contacto telefónico.
  - Adicionar oferta aos favoritos.
  - Obter as direcções para a oferta utilizando uma aplicação de navegação do dispositivo.
- **Partilha Activity** – recebe os dados da oferta e dispõe das seguintes acções:
  - Partilha no mural do *facebook.*
  - Partilha por *correio electrónico.*
  - Partilha por *SMS.*
- **Área Pessoal Activity** – dispõe das seguintes acções:
  - Favoritos: lista e mapeia as ofertas adicionadas aos favoritos.
  - Top de Visualizações: faz o pedido à *API* das 10 ofertas mais visualizadas.
  - Contactos: recebe e valida os dados submetidos pelo utilizador, para efectuar um contacto ao suporte e subscrever *newsletter.*

7.2- Permissões

Para o devido funcionamento da aplicação requer acesso as seguintes permissões:

- **Internet** – esta permissão é necessária para fazer qualquer tipo de pedido através do protocolo *HTTP*, necessário para a comunicação com a *Web API*
- **Acesso ao estado da rede** – esta permissão é necessária para verificar se alguma ligação de dados está activa, na aplicação se esta verificação falhar é pedido ao

utilizador para activar uma ligação de dados para o correcto funcionamento da aplicação

- **GPS** – apesar de a aplicação não usar directamente o GPS do telefone, esta permissão necessita de estar activa porque em algumas versões mais antigas do *Android* é a permissão mãe para termos acesso a localização por rede
- **Acesso à localização de rede** – esta permissão garante acesso à localização aproximada, proveniente das torres telefónicas e por *Wi-Fi*, necessária para mapear a posição do dispositivo e para calcular a distância relativa a uma oferta.
- **Escrita no armazenamento externo** – este tipo de permissão é necessário quando a aplicação necessita de escrever algo na memoria física do dispositivo, na aplicação, quando.

7.3- Bibliotecas

Neste capítulo são abordados os principais problemas e as soluções encontradas, que remetem para as bibliotecas usadas.

7.3.1- Lazzy List

Lazzy List é uma das bibliotecas que usei para preencher a lista de ofertas, o diagrama de classe pode ser consultado em 8.2.2. Esta biblioteca surgiu da necessidade de popular uma *ListView* com inúmeros elementos, o *Android* originalmente não apresenta nenhuma biblioteca para resolver esta situação eficientemente.

O *LazyAdapte*r, presente nesta biblioteca, herda as propriedades do *BaseAdapter* presente nas bibliotecas padrão do *Android*, este adaptador preenche a lista com imagens e textos ou outro conteúdo visual de forma assíncrona. Usando o *BaseAdapter* original para preencher uma lista com imagens pode ser tornar um grande problema, uma vez que as imagens são conteúdo mais pesado que o texto e como *BaseAdapter* corre na *thread* principal pode bloquear a aplicação e até mesmo a *crashar*, não suportando a hipótese de preencher uma *ImageView* com uma imagem proveniente da internet, que é o nosso caso, a API fornece-nos o *url* da imagem para uma dada oferta e a aplicação tem que fazer o *download* da imagem e preencher a *ImageView* com essa imagem.

Uma das classes presente na *LazzyList* é a *ImageLoader*, que recebe o *url* da uma dada imagem e o elemento onde a imagem deve ser mostrada. Correndo um processo em segundo plano é feito o *download* da imagem de forma assíncrona e grava essa imagem em cache, numa directória na memória interna do dispositivo, a chave para essa imagem é o url. Uma vez a imagem na cache não irá ser feito o *download* novamente da imagem, apenar irá definir o caminho para a imagem no elemento pretendido, na aplicação existem ofertas que usam a mesma imagem, neste caso apenas um *download* é efectuado.

Para poupar recursos as imagens e os textos só são preenchidos quando estes estão dentro do *ViewPort* da aplicação e aquando do *download* é feito um redimensionamento da imagem para poupar antes de gravado em cache. A classe *ImageLoader* é utilizada também para carregar a imagem no detalhe de uma oferta, mas, nesta caso não é feito um redimensionamento da imagem uma vez que o elemento é maior e requer uma imagem com mais qualidade.

Esta biblioteca foi criada baseado no padrão de desenho *Lazy Loading* de Engenheira de Software, que consiste no adiamento da inicialização de um objecto até ao momento em que ele é necessário, assim consegue-se criar aplicações mais responsivas e mais eficientes.

7.3.2- Map View Ballons

Map View Ballons é outra biblioteca que veio resolver um problema que o Android originalmente não tem solução, que é, ao adicionar anotações ao *MapView* não permite criar um *pop-up* com informação extra acerca desse item. Com esta biblioteca é possível criar um balão, que é gerado quando se clica em uma anotação, para podermos mostrar alguma informação extra. Na aplicação temos 3 campos, a imagem da oferta que é carregada usando o mesmo método que na LazzyList, abordado em 7.3.1, a percentagem do desconto e o nome do comerciante.

Esta biblioteca está dividida em 2 partes:

- **CustomOverlayItem** que herda as propriedades do objecto nativo OverlayItem, necessário para mapeamento das anotações no mapa, relativamente a posição geográfica. É o item inserido na camada de anotações do mapa, onde estão mapeados os pontos das ofertas.
- **CustomItemizedOverlayBallon**, a criação deste elemento é desencadeado no evento de clique de *CustomOverlayItem*, e cria o dito balão com as informações adicionais. Este elemento tem também um evento de clique, que neste caso, está trabalhado de forma a abrir o detalhe da oferta.

Para cada oferta é criado um CustomOverlayItem que tem associado um CustomItemizedOverlayBallon, esses itens são adicionados a uma lista e esta é adicionada à camada superficial do mapa, MapView é o elemento que desenha o mapa com base na data obtida através do serviço *Google Maps*, esse mapa estende. Esta biblioteca tornou-se fundamental para aumentar a interacção do utilizador com a aplicação, aumentando directamente a experiência de utilização.

8- Análise e Desenho

A linguagem *UML* surgiu para especificação, construção, visualização e documentação de artefactos de um sistema de informação. É promovida como *standard* pelo *OMG* com contribuições de diversas empresas da indústria de software.

O UML tem sido adoptado pelas empresas e instituições em todo o mundo, existindo actualmente mais de 50 ferramentas comerciais e académicas para modelação de software e de negócio baseadas em UML.

Actualmente a versão 2 do UML providência treze tipos de diagramas segundo três visões principais, sendo eles:

- Visão Estática ou Estrutural – Diagrama de Pacotes, Diagrama de Classes, Diagrama de Objectos, Diagrama de Estrutura Composta, Diagrama de Componentes e Diagrama de Instalação.
- Visão Funcional - Diagrama de casos de uso e Diagrama de actividade.
- Visão Dinâmica - Diagrama de máquina de estados e Diagrama de interacção (Diagrama de Sequência, Diagrama de Comunicação, Diagrama de Visão Geral da Interacção e Diagrama Temporal).

Dentro destas visões principais decidi usar a Visão Funcional que contém o Diagrama de Casos de Uso para descrever a relação entre actores e casos de utilização da aplicação, e a Visão Estrutural para Diagramas de Classes.

8.1- Diagrama de Casos de Uso

Casos de uso, são definidos como narrativas em texto. Descrevendo a unidade funcional, e são amplamente utilizados para descobrir e registrar requisitos de sistemas. Os diagramas de Casos de Uso são representações dos mesmos, podendo ser representados por uma elipse contendo, internamente, o nome do caso de uso.

Neste projecto vamos ter apenas 1 actor, ou seja, o utilizador que acede a aplicação no dispositivo móvel (Fig. 1).

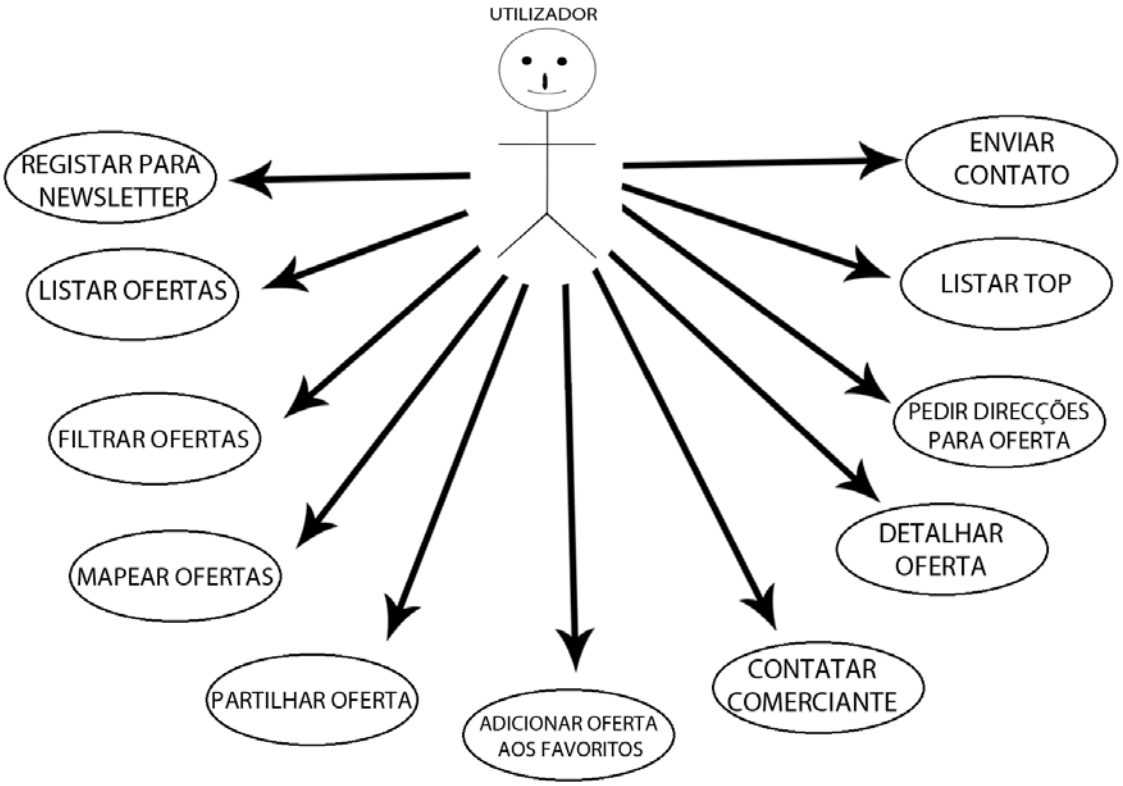


Figura 1 - Diagrama de Casos de Uso

8.2- Diagrama de Classes

8.2.1- Oferta

Objecto que armazena os dados de cada oferta, as ofertas são armazenadas em um *array* de objectos oferta.

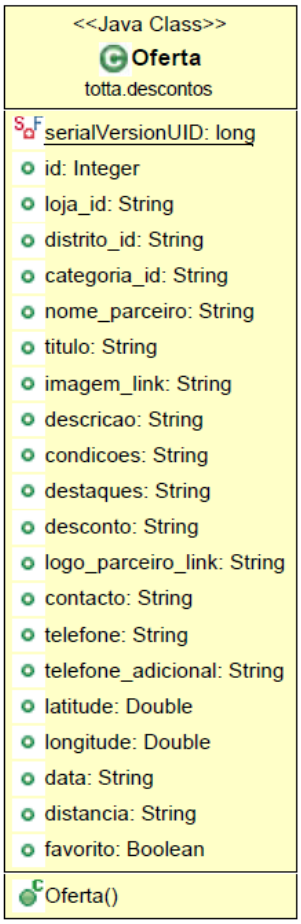
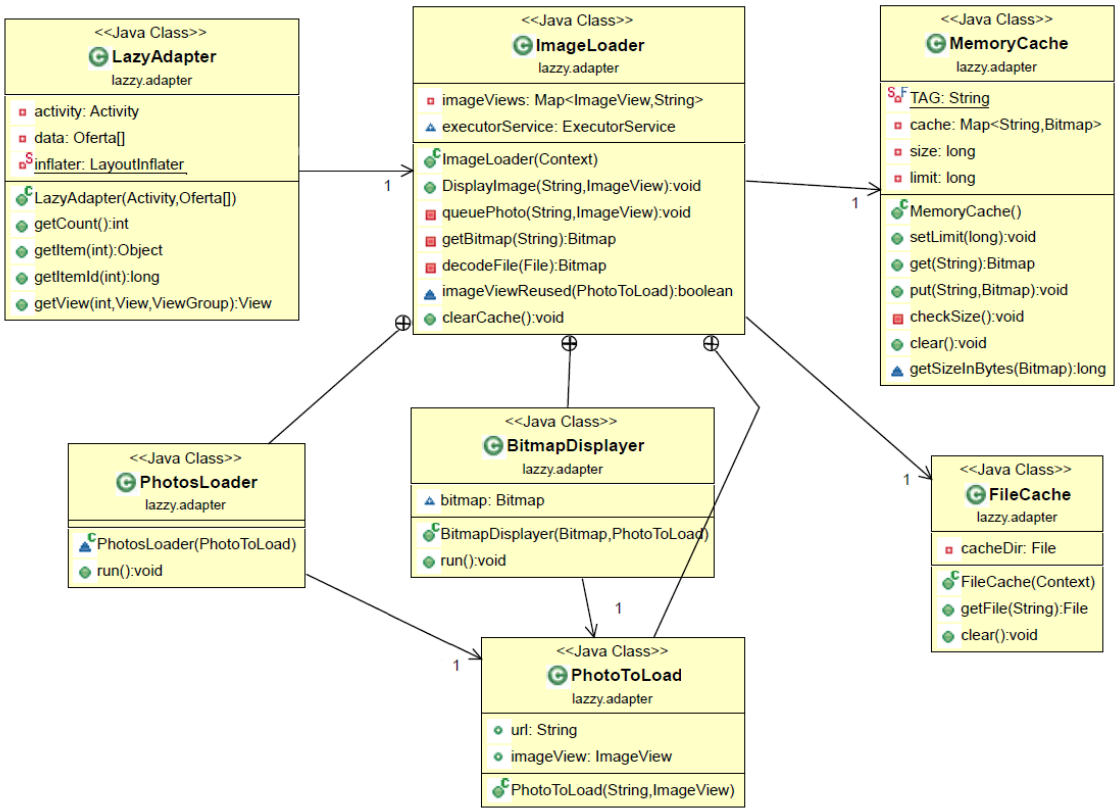


Figura 2 – Diagrama de classe para o objecto oferta.

8.2.2 – Lazzy List

Classe responsável pela listagem das ofertas, carregamento e armazenamento das respectivas imagens.



igura 3 – Diagrama de classe da biblioteca Lazzy List

8.2.3 – MapViewBallon

Classe responsável pelo mapeamento das ofertas, e construção dos balões informativos.

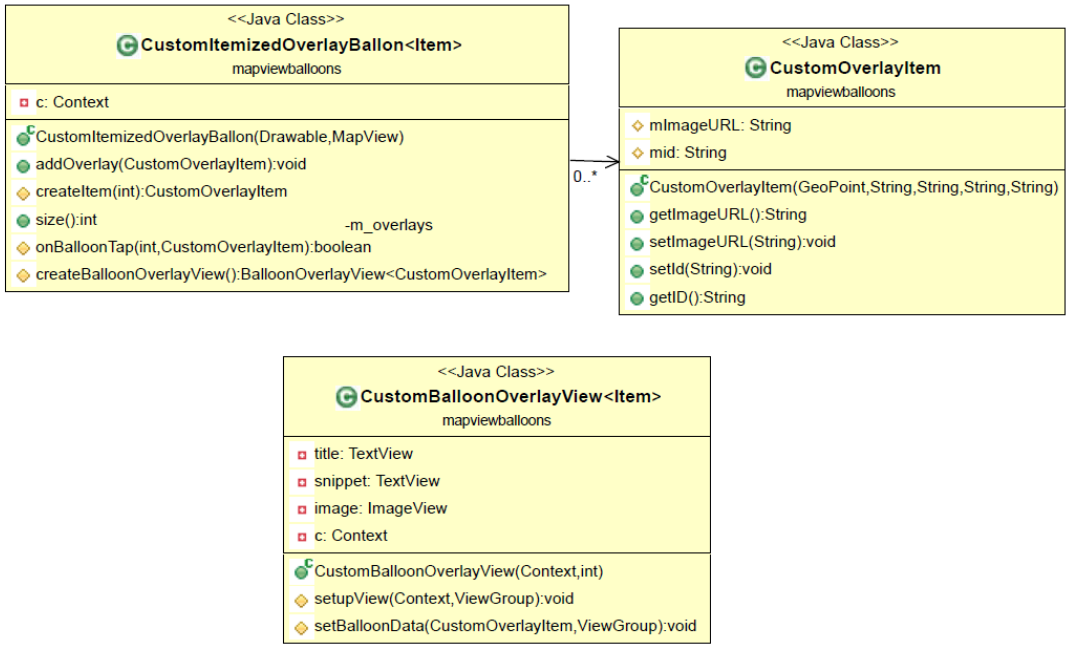


Figura 4 – Diagrama da classe MapViewBallon

9- A aplicação

Neste capítulo são apresentadas algumas imagens dos ecrãs principais da aplicação.

Figura 5 – *Splash Screen*

Figura 6 – *Landing Screen* Mapa

Figura 7 – *Landing Screen* Lista

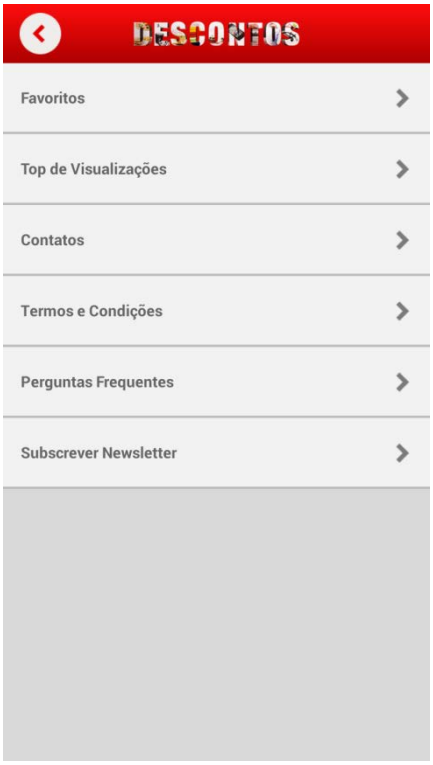


Figura 8 – Área Pessoal

Figura 9 – Detalhe oferta

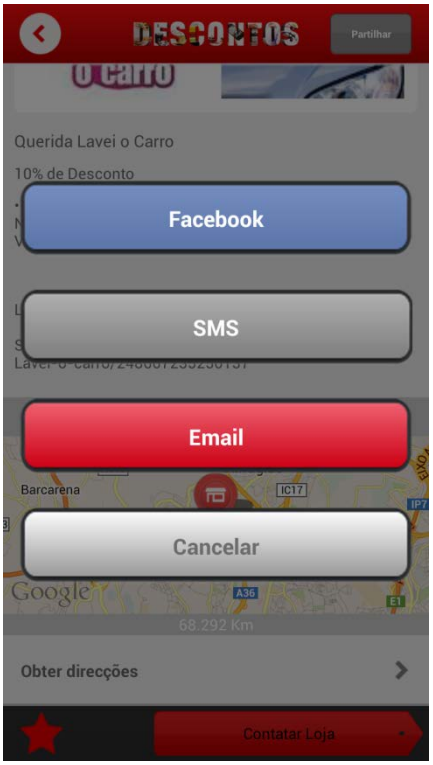


Figura 10 – Partilha da oferta

Figura 11 – Formulário de *Newsletter*

Figura 12 – Formulário de contacto

A versão final da aplicação pode ser descarregada em https://dl.dropboxusercontent.com/u/25495859/tfc_totta_pedrosilva.apk.

10- Bibliografia

1. MEIER, Reto. Android 4 Application Development, Wrox
2. QUEIRÓS, Ricardo. Android – Introdução ao Desenvolvimento de Aplicações, MyTi
3. FONSECA, Nuno; REIS, Catarina; SILVA, Catarina; MARCELINO, Luís; CARREIRA, Vítor. Desenvolvimento em iOS iPhone, iPad e iPod Touch - Curso Completo (2.ª Edição Actualizada), FCA
4. SILVA, Alberto; VIDEIRA, Carlos. UML Metodologias e Ferramentas CASE, Volume 1, 2ª Edição, CentroAtlantico.pt.
5. Stack Overflow – (Disponível em: http://stackoverflow.com/questions/tagged/android)
6. Documentação Android – (Disponível em: http://developer.android.com/guide/components/index.html)
7. SMITH, Dave; FRIESEN, Jeff; Android Revipes – A problem-Solution Approach, APRESS

11- Glossário

JDK - Java Development Kit
Android - É um sistema operacional baseado em Linux para dispositivos móveis.
API – Application Programming Interface ou Interface de Programação de Aplicativos.
REST - Representational State Transfer ou Transferência de Estado Representativo
POST – É um dos pedidos possíveis em HTTP.
ID – Identificação.
LAYOUT – Estrutura visual para interacção com a aplicação.
JSON – É um formato leve para intercâmbio de dados computacionais, é um subconjunto da notação de objecto em JavaScript.
JavaScript – Linguagem de programação.
Newsletter – Boletim informativo, geralmente em formato electrónico, de distribuição regular para os seus assinantes.
Splash Screen – Primeiro ecrã de uma aplicação.
UML- Unified Modeling Language ou Linguagem Unificada de Modelagem.
OMG- Object Management Group.
ViewPort- Campo visual da aplicação.
Pop-up- É uma janela criada dentro do campo visual da aplicação de forma a criar um efeito interactivo.