**I**nstituto **S**uperior de **E**ngenharia de **L**isboa

Licenciatura em Engenharia Informática e de Computadores

Semestre de Inverno 2014/2015



3ª Série de exercícios

PDM

Programação em Dispositivos Móveis

Eng.º Pedro Félix

Data de entrega: 22.12.2014

**Trabalho elaborado por:**

Pedro Lima

Nº. 33684

Sílvia Ganhão

N.º 35418

LI51D – LEIC

Índice

[Exercício 1 3](#_Toc406962682)

[Exercício 2 5](#_Toc406962683)

# Exercício 1

O objetivo deste exercício foi o de alterar a aplicação de notícias *Thoth*, realizada ao longo das últimas duas séries, de forma a serem usados fragmentos na implementação da *user interface*. Para tal, foi necessário acrescentar suporte para interface *master-details* com dois painéis, e navegação entre notícias através de *swipe*.

Para a realização foram efetuadas várias alterações nas classes já existentes, bem como o acrescento de novas classes.

Em termos de alterações, foram efetuadas as seguintes:

* ***MainActivity:*** Esta classe passou a extender de *FragmentsActivity* para assim ser possível o suporte de fragmentos na implementação da *user interface*. Os fragmentos neste caso passam a ter o seu ciclo de vida dependente do ciclo de vida da *MainActivity*.

A principal diferença nesta classe é que em vez de ser a partir daqui que é chamado o *NewsCustomAdapter*, este vai passar a ser chamado dentro do fragmento criado, *NewsItemListFragment*. Esta classe tem um método *OnItemClickListener*, onde serão lançados novos serviços consoante o tipo de ação efetuada pelo utilizador, que poderá ser a apresentação dos detalhes de um item da lista de notícias.

* ***NewsCustomAdapter:*** O *layout* para apresentação da lista de notícias deixou de ser apresentado a partir da uma *ExpandableListView*, passando esta classe a extender de *BaseAdapter* e a implementar *OnItemClickListener*, para que ao se carregar num item de notícia seja lançado um fragmento. Com estas alterações procedeu-se à nova implementação dos métodos herdados, de forma a se realizar o novo *bind model* dos *widgets* associados ao fragmento.
* ***NewItem, NewsAsyncTask, NewsClassOpenHelper, ViewModelParent, NewsService, HttpRequestsToThoth:*** Estas classes foram alteradas unicamente com o intuito de suportar a alteração efetuada a base de dados das notícias que a partir deste momento passam a ter associado o identificador e o nome da classe a que pertence a notícia.

Em termos de introdução de novas classes:

* ***NewsItemActivity:*** Esta classe será a atividade que irá ser lançada quando se carregar num elemento da lista de notícias. Extende de *FragmentActivity*, e será usada, para lançamento do fragmento onde se encontrará os detalhes da notícia. Tem associado um *ViewPager*, o qual é o que suporta o requisito de navegação *swipe* entre as notícias. A partir desta classe é requisitado um *adapter* para suporte do fragmento, o qual se encontra implementado na classe *NewsItemFragment*.
* ***NewsItemFragment:*** É nesta classe que é feita a criação e preenchimento da *view* para apresentação dos detalhes da notícia. Esta classe extende de *Fragment*.
* ***NewsItemListFragment:*** Esta é a classe onde é chamado o *custom adapter* da *view* que apresenta a lista de todas as notícias. E que ao detetar um clique sobre um desses items lança a atividade associada à apresentação de um fragmento, que é a *NewsItemActivity*.
* ***NewsListModel:*** Implementa *Serializable*, e é usada para se obter a lista das notícias, ou um item dessa lista.

Para suporte da apresentação dos dois novos tipos de *views*, sendo os mesmos a partir de fragmentos foram criados os ficheiros xml, *news\_item\_fragment\_layout*, e  *news\_item\_list\_layout*.

Foi ainda criado o *layout*, *activity\_two\_pane*, no qual é indicado a existência de dois *FrameLayout*, isto para suportar o requisito de *master\_details*, no que diz respeito à visualização das notícias.

Foi também criada a pasta *values-sw600dp*, a qual contém um ficheiro *refs*, onde se indica a existência de *layouts* que suportam o *master\_details*, pela criação de um atributo *item*, com o nome *activity\_masterdetail*. Esta pasta é para suporto à visualização em dispositivos maiores, como por exemplo, *tablets*.

# Exercício 2

Este exercício teve como objetivo a adição à aplicação *Thoth*, a capacidade de apresentar os participantes de uma turma. Esta apresentação deverá conter a fotografia do participante e deverá ter em conta a minimização dos consumos de recursos necessários para esta tarefa.

Para a realização deste exercício também foram efetuadas alterações em classes já existentes bem com criadas novas classes.

No que diz respeito a alterações:

* ***HttpRequestsToThoth:*** Nesta classe foram acrescentados novos métodos, que permitem a realização de pedidos ao *Thoth*, para obtenção dos participantes de uma dada turma.

No que diz respeito a novas classes:

* ***Participant:*** Esta é classe onde é definido o tipo *Participant*, o qual é definido pelo seu número, nome, email, fotografia do *Thoth*, e se é professor ou estudante.
* ***ParticipantsCustomAdapter:*** classe que extende de *BaseAdapter* e implementa *OnItemClickListener*, e onde é garantida a correta associação dos nomes dos *widgets* com a informação dos participantes que se pretende apresentar. Esta classe é auxiliada, pela classe *ParticipantViewModel*.
* ***ParticipantViewModel:*** classe que disponibiliza os *widgets* da *view* de forma a facilitar a associação com a informação do participante a apresentar.
* ***ParticipantsAsyncTask:***  Classe onde se faz o pedido de todos os participantes da turma, com o auxilio da classe *HttpRequestsToThoth*. Esta classe devolve um *array* contendo esses mesmos participantes.
* ***ParticipantsActivity:*** Esta classe será a atividade que irá ser lançada quando se carregar no botão, que se encontra no *layout* da apresentação das turmas, que representa a visualização dos participantes dessa turma. Extende de *FragmentActivity* e implementa a interface callback definida na classe *ParticipantItemListFragment*, a qual define o fragmento que está associado a esta classe. A partir do método *onItemClickListener*, herdado de *Callback*, quando se deteta que o utilizador carregou num participante, é lançada uma nova atividade, *ParticipantItemActicity*, que é a responsável pela apresentação dos detalhes do participante.
* ***ParticipantItemListFragment:*** Classe a partir de onde é chamado o *custom adapter* associado aos participantes, *ParticipantsCustomAdapter*.
* ***ParticipantItemActivity:*** Esta classe será a atividade que irá ser lançada quando se carregar num elemento da lista de participantes. Extende de *FragmentActivity*, e será usada, para lançamento do fragmento onde se encontrará os detalhes do participante. Tem associado um *ViewPager*, o qual é o que suporta navegação *swipe* entre os participantes. A partir desta classe é requisitado um *adapter* para suporte do fragmento, o qual se encontra implementado na classe *ParticipantItemFragment*.
* ***ParticipantItemFragment:*** É nesta classe que é feita a criação e preenchimento da *view* para apresentação dos detalhes do participante. Esta classe extende de *Fragment*.
* ***ParticipantListModel:*** Implementa *Serializable*, e é usada para se obter a lista dos participantes, ou um participante dessa lista.

Foram ainda criados novos *layouts* para suportar o visionamento dos fragmentos da lista dos participantes bem como de cada participante por si. Para apresentação da lista dos participantes, existe o ficheiro *xml* com o nome *participant\_item\_list\_layout*, e para a apresentação dos detalhes de um único participante, existe o ficheiro *xml* com o nome, *participant\_item\_fragment\_layout*.

Foi ainda criado o *layout*, *participant\_two\_pane*, no qual é indicado a existência de dois *FrameLayout*, isto para suportar o requisito de *master\_details*, no que diz respeito à visualização dos participantes das turmas.

Na pasta de nome *values—sw600dp*, referida anteriormente, foi adicionado um novo atributo *item*, de nome *participant\_masterdetail*.