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

Licenciatura em Engenharia Informática e de Computadores

Semestre de Inverno 2014/2015



Projeto Final

PDM

Programação em Dispositivos Móveis

Eng.º Pedro Félix

Data de entrega: 19.01.2015

**Trabalho elaborado por:**

Pedro Lima

Nº. 33684

Sílvia Ganhão

N.º 35418

LI51D – LEIC

Índice

[Introdução 3](#_Toc408869648)

[IselAppServer 4](#_Toc408869649)

[IselApp 6](#_Toc408869650)

[Entidades 6](#_Toc408869651)

# Introdução

O objetivo deste trabalho é a implementação de uma aplicação Android que interage com a ferramenta de ensino Thoth, através da sua API. As funcionalidades presentes na aplicação são: a escolha de turmas, o visionamento das notícias e dos trabalhos associados a cada uma das turmas escolhidas e ainda é possível ver os participantes de cada turma.

Para a melhor utilização desta aplicação foram criados 2 projetos, o IselAppServer e o IselApp, para que o utilizador possa aceder á informação sem ser necessário ligação à Internet.

As soluções utilizadas para solucionar os diversos desafios encontrados durante a implementação da aplicação são descritas ao longo deste relatório.

# IselAppServer

Esta aplicação foi criada de forma a implementar o Content Provider e a criar a base de dados que dá suporte à informação disponibilizada ao utilizador. É nesta base de dados criada que a informação fica armazenada para que o utilizador possa aceder-lhe sem que seja necessário fazer qualquer tipo de conexão à Internet. Criou-se uma base de dados SQLite, de nome **IselApp.db**. Para fazer a gestão desta base de dados é necessário implementar um Content Provider.

Na base de dados foram criadas 3 tabelas:

* A tabela **classes**, que armazena a informação relativa às turmas existentes no Thoth;
* A tabela **news**, que armazena a informação relativa às notícias das turmas selecionadas pelo utilizador;
* A tabela **workItems**, que armazena a informação relativa aos trabalhos associados a cada turma escolhida pelo utilizador.

O modelo Entidade-Associação destas tabelas está definido na Figura 1.

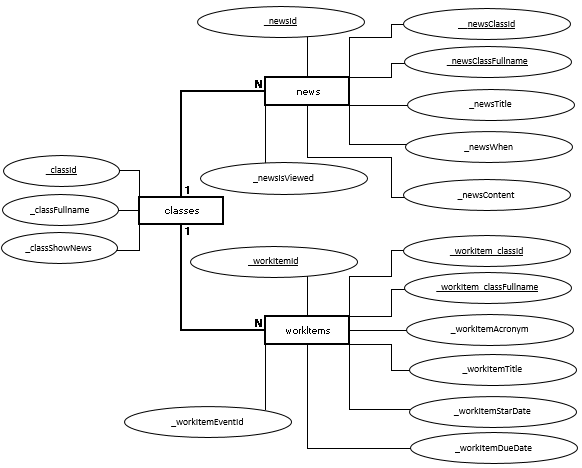


Figura - Modelo EA

Para criar a base de dados IselApp.db foi definida a classe **IselAppOpenHelper**. De modo a que a classe tenha os requisitos necessários para criar a base de dados é preciso que estenda de **SQLiteOpenHelper**, classe que permite a criação e a gestão de bases de dados. É nesta classe que é criado o ficheiro da base de dados assim como as tabelas que suportam o modelo de dados obtido a partir do Thoth.

Para além desta classe, foram ainda definidas mais 2 classes: a classe **IselAppContentProviderContract** onde está definido o valor do campo **Authority** associado a este Content Provider, que é utilizado pela outra classe implementada, a classe **IselAppContentProvider**. Esta última classe referida é responsável por fazer toda a gestão da base de dados da aplicação (inserts, deletes, updates e querys) através de uma instância da classe **IselAppOpenHelper**, do campo **Authority** e de um **Uri** que identifica univocamente cada uma das possíveis tabelas presentes na base de dados. A combinação destes elementos possibilita a correta gestão da base de dados da aplicação.

# IselApp

É neste projeto que está feita toda a implementação das funcionalidades tidas como objetivo desta aplicação. Para o correto e eficiente funcionamento desta aplicação foram implementadas diversas soluções, recorrendo aos mais variados objetos disponibilizados pelo Android (AsyncTask, Service, Fragments, …).

Para melhor organização do projeto, foram criados vários packages, de forma a separar as classes por funcionalidades.

## Entidades

De forma a guardar a informação obtida através do Content Provider ou de um pedido HTTP para auxiliar na manipulação dos dados ao longo da execução da aplicação, foram criadas 4 classes representando cada uma delas uma entidade diferente. As classes são:

* **ClassItem** - armazena os dados associados a uma turma;
* **NewsItem** – armazena os dados associados a uma notícia;
* **WorkItem** - armazena os dados associados a um trabalho;
* **ParticipantItem** – armazena os dados associados a um participante.

Todas estas classes implementam a interface **Serializable**, estando o motivo explicado mais á frente no relatório.