TRABALHO FINAL DE IMPLEMENTAÇÕES DE APLICAÇÕES ESPECIAIS

Introdução

Este trabalho consiste no desenvolvimento de um aplicativo para a plataforma Android e conta como avaliação final para a cadeira Implementações de Aplicações Especiais do curso de Sistemas de Informação. O domínio do aplicativo é uma lista de compras que permite ao usuário criar, editar e deletar listas e produtos. Produtos podem ser adicionados a uma lista de compras indicando a quantidade. É mostrado também o total de produtos na lista.

Este relatório apresentará mais detalhes do domínio da aplicação, bem como a solução desenvolvida.

Problema

O aplicativo tem por objetivo prover uma solução para o usuário criar listas de compras de forma rápida, adicionando e removendo produtos dinamicamente. Este domínio foi escolhido pela necessidade, e um pedido de um usuário que, depois de ver os exercícios implementados para curso, solicitou-me uma aplicação com estes recursos.

Solução

Na implementação das classes de domínio foram aplicados os design patterns Singleton, Dependency Injection e Data Mapper.

O padrão Singleton “é um design pattern (padrão de projeto) que tem o objetivo garantir que existe apenas uma instancia de uma certa classe a qualquer instante e em qualquer ponto de um sistema.” (1). Este padrão foi utilizado na classe BancoDeDados que provê o acesso à base de dados.

O Dependency Injection (injeção de independencia) “consiste na passagem de uma classe para outra, sendo que esta última irá utilizá-la (consumí-la), visando eliminar o forte acoplamento geralmente existente entre os módulos da aplicação.” (2). Este padrão foi utilizado em todas as classes da aplicação.

O Data Mapper é um dos padrões de projeto utilizados para fazer o mapeamento de uma tabela do banco de dados em classes do sistema. Este padrão separa a representação das características do objeto de suas ações. Uma classe é o modelo que representa os dados da tabela em memória e outra somente é responsável pelos métodos de persistência.

A tela inicial do aplicativo é um ListActivity que lista as listas de compras cadastradas. Ao clicar em uma lista, dispara o método onItemClick que abre uma nova activity mostrando os produtos da lista. Nesta activity, um click no produto abre um dialog que permite editar os dados da produto na lista de compras. Ao pressionar um item, dispara o método onItemLongClick que abre um dialog perguntando se deseja remover o produto da lista de compras. Ao clicar no botão MENU do aparelho, abre um menu a opção de adicionar mais um produto na lista de compras.

De volta a tela inicial, pressionando um item da lista é disparado o método onCreateContextMenu que abre um menu de contexto com duas opções: editar ou remover a lista de compras.

Pressionando a tecla MENU do aparelho na tela inicial, abre um menu com as seguintes opções:

- Nova Lista: abre um dialog para inserir o nome de uma nova lista de compras;

- Ver Produtos: abre uma activity que lista todos os produtos cadastrados;

- Novo Produto: abre uma activity para cadastrar um novo produto;

- Mais Comprados: abre uma activity listando os produtos mais vezes inseridos em listas de compras cadastradas;

Na tela que lista os produtos cadastrados é possível editar ou remover um produto através do menu de contexto. É possível cadastrar um novo produto utilizando o menu principal.

Conclusão

Este aplicativo não contempla todas as funcionalidades que eu queria implementar, devido ao pouco tempo de desenvolvimento. Fiz muitas pesquisas para implementar os menus e entender quais métodos necessários deveriam ser sobrescritos.

A criação dos telas dialogs também foram frutos de muita pesquisa. Como adicionar um layout ? Como tratar o click dos botões ? Como disparar um método ao clicar no botão do dialog ? Foram questões que encontrei no desenvolvimento.

Implementei os design patterns como forma de aplicar o conhecimento que estou adquirindo com os estudos de programação. À principio, o alvo dos meus estudos é o padrão MCV em PHP e C#, mas os padrões de projetos são independentes da linguagem.

Pretendo continuar desenvolvendo este aplicativo mesmo após o fim deste curso, pois, como disse, ele será utilizado, mesmo que seja por um usuário apenas.

Referências

1-<http://javafree.uol.com.br/wiki/Singleton>

2-http://imasters.com.br/artigo/20079/desenvolvimento/injecao-de-dependencia-desacoplando-sua-aplicacao/