Implementação do código de conversão de moedas pelo Android Studio

Nome: Taís Baierle
Programação 4 - 5N - 2019/02
Professor Gabriel da Silva Simões

O aplicativo foi desenvolvido no Android Studio na API 28, android 9.0 Pie, com o ambiente de testes um aparelho com o android especificado de tela 1480x720. O aplicativo funciona buscando arquivos em formato JSON em um webservice e alimentando um banco de dados SQLite nativo da aplicação android. A atualização do banco foi implementada utilizando uma função em segundo plano que não ocupa o thread de tela. O cálculo das moedas é feito utilizando a cotação vigente do dia, de acordo com a informação do webservice.

1. Classes Java da aplicação

1.1 Classe BancoDados

A classe BancoDados centraliza o CRUD do aplicativo utilizando a importação das classes SQLiteDataBase e SQLiteOpenHelper. O SQLiteDataBase possui os métodos para executar os comandos SQL e tarefas comuns dos sistemas de gerenciamento de banco de dados. O SQLiteOpenHelper faz o gerenciamento da criação do banco e o controle de versões.

Foi criada uma classe estática para fazer a montagem do SQL de criação de tabela implementando a classe BaseColumns que retorna uma string com o CREATE TABLE. Depois foi criada a classe privada BancoDadosHelper que implementa os métodos da classe SQLiteOpenHelper, onde executa a criação do banco e controle de versões.

O método maxData faz uma busca no banco de dados retornando a última linha de registro, com um objeto cotação na qual a data é a informação de interesse, pois essa data indica até que ponto as informações do banco estão atualizadas em relação ao dia atual. Existe um try/catch nesse método que retorna nulo quando captura uma exceção, esse retorno nulo nos indica que o banco está vazio, portanto será tratado diferente na hora de fazer as inserções do webservice.

O método insertCotacao recebe um objeto Cotacao e faz a inserção da informação do id, do tipo de moeda, do valor do dia e da data do JSON.

O método verificarDuplicatas recebe um objeto recém instanciado do JSON e verifica se as informações já existem no banco de dados, caso existam retorna verdadeiro, e caso não tenha, retorna falso. Como os arquivos de JSON são lidos, esse método faz o filtro de informações que possam estar duplicadas e que não precisam ser inseridas novamente.

O método consulta recebe uma string moeda, e faz uma busca no banco de dados listando todas as cotações e datas daquela determinada moeda. É criado um ArrayList de Cotacoes, e dentro de um laço de repetição são criados objetos Cotacoes, que são instanciados pelo retorno do cursor e adicionados no ArrayList, que é retornado.

O método moedaDia faz uma busca no banco de dados recebendo uma string com a moeda e outra string com a data da última data de cotação no banco de dados, se as informações estiverem devidamente atualizadas no banco, traz a cotação do dia de hoje. Retorna um objeto Cotacao com as informações da busca.

1.2 MainActivity

Tela principal da aplicação que contém a activity principal, no thread de tela é feita as conversões de moedas e no thread secundário ele executa o download e a leitura das informações.

No onCreate da MainActivity foram criados dois spinners que recebem cada um array de strings com as moedas que existem no banco, por exemplo “USD-Dólar, e se faz um o uso do método substring para pegar as três primeiras letras, e recupera com getters e setters para mandar para o método de cálculo.

Foi criado um menu de opções na barra superior, foi criado um modelo na pasta menu de layouts que contam com duas opções, créditos e data. Ao clicar nas opções do menu, foi feito alertas de tela, como o AlertDialog. O AlertDialog foi instanciado e foi utilizada algumas funções de formatação de texto, o primeiro alerta mostra os créditos do autor e o segundo mostra sempre a última data gravada no banco de dados, caso o banco esteja vazio, aparece um aviso de banco desatualizado.

O método pegarValor pega o valor do campo de texto e converte para decimal e retorna para o cálculo de cotações.

Os métodos sair e telaHistorico respectivamente fecham o aplicativo e encaminham para a tela de histórico.

O método conversao recebe as informações do método pegarValor e faz duas buscas no banco de dados pela última data da cotação das moedas selecionadas nos spinners, e após isso faz um cálculo de regra de três para determinar o valor da moeda convertida. Também é feito uma máscara na saída dos valores monetários, e gera uma captura de exceção quando é feita a conversão de um campo vazio.

O método atualizar é onde ocorre a conexão com o webservice, bem como o download e leitura das informações dos arquivos baixados num diretório previamente gerado pela aplicação. São criadas duas instâncias da classe Calendar, onde uma pega o dia de hoje e outra recebe o retorno do método diasAnteriores, que pega o dia atual e desconta 40 dias e retorna um Date, que é previamente convertido para Calendar. Posterior a isso, é feita uma busca no banco pela última data, se a busca cair na exceção e retornar nulo, o banco está vazio. Portanto é feito um laço de repetição enquanto a data atual for depois da data de 40 dias atrás, ficará iterando e gerando um List de URLs, que será mandado para a classe DownloadJson. Se a busca pela data gerar um resultado, será feita a iteração somente em cima dos dias de intervalo entre o banco de dados e o dia atual.

A classe DownloadJson implementa o AsyncTask, que tem os métodos onPreExecute, que implementa as etapas antes da execução do AsyncTask, e são feitas pelo thread de tela, no caso do aplicativo em questão, nessa etapa desabilita os botões de atualizar e listar, e inicia a barra de progresso.

O método doInBackgrournd faz o processamento das informações em um outro thread, e não ocupa o thread de tela. O processo feito em segundo plano na aplicação consiste em recepcionar as URLs que foram passados por parâmetro na forma de vetor, estabelecer uma conexão com o método URLConection, e criar um diretório especifico para seu uso no dispositivo e salvar os arquivos de cada umas das requisições. A entrada de informações e a escrita é feita com ImputStream e OutputStream, esse último usando um laço de repetição para a escrita, o controle da barra de progresso acontece dentro do forEach, que itera sobre o vetor

de URLs, adiciona 1 a variável barProgress que depois é passada na forma de porcentagem para a barra de progresso da tela.

Após o download das informações e devidamente salvas no dispositivo, e feita a leitura dos arquivos em JSON, para isso foi criada uma variável do tipo File, onde passa o diretório criado e faz um forEach iterando sobre a lista de arquivos, com auxílio das instancias das classe FileReader e BuffererReader foi feita a leitura dos arquivos que texto. Para pegar as informações do JSON foi utilizado a classe JSONObject, na qual é possível se instanciar ela informando a chave do JSON na qual deseja as informações, uma vez que a estrutura chave valor desses arquivos permita esse tipo de manipulação. Foram utilizados dois objetos, um para pegar as informações mais externas do JSON e outro mapeando a chave "rates" do JSON, onde ficavam os valores por moeda.

Feito isso, foram criados 4 objetos Cotacao e instanciados com os objetos JSON recém-criados. Foi feito um filtro pela chave de 4 moedas solicitadas. Após instanciar, e feito uma busca no banco de dados por registros duplicados, uma vez que não forem encontrados, é feito a inserção no banco de dados.

O método onPostExecute já feito pelo thread de tela após a finalização do doInBackground, onde os botões são habilitados novamente e a barra de progresso é zerada.

1.3 TelaHistorico e Cotacao

A classe Cotacao é uma classe para receber as informações do webservice e instanciá-las no banco de dados.

A classe TelaHistorico implementa um spinner, onde é instanciado por um vetor de strings com o nome das moedas, com getters e setters, os nomes são recuperados e utilizando o método substring, que pega o código da moeda. Após isso, o código da moeda selecionada é passado para um método que faz uma busca no banco das cotações dessa moeda, colocando-os em objetos Cotacao e retornando num ArrayList. Esse ArrayList é impresso num TextView com uma barra de rolagem, onde lista um a um as cotações dos dias no banco de dados, sem listar os fins de semana, onde a cotação de sexta-feira é repetida.

1.4 Telas da Aplicação

Consiste em duas telas, uma delas faz a conversão das moedas e a outra lista o histórico das cotações. A tela das conversões consiste em dois spinners, um campo textual para a entrada do valor a ser convertido, dois TextViews, um para os valores das moedas após a conversão e outro para mostrar a mensagem de atualização de dados, uma barra de progresso e quatro botões, um para sair da aplicação, um para atualizar o banco de dados, outro para ir na tela de histórico e o último, para converter.

Na tela de histórico, existe um spinner para seleção de moedas, um botão que gera o histórico e uma TextView rolável que exibe a lista, e um botão que volta para a tela principal.