Report of Dragable Notebook

Lu Guandong, Qi Zhenlin, Mao Yong, Han Bing

May 18, 2017

Abstract

This report shows you why we develop the app, the function of the app and the features of the app. We include some source code in this report.

Contents

1 A brief introduction of the app

This app tends to manage things you should do in the future. You can add things in natural language. Then the app can sort by the importance of the things. The most important things will be listed at the top. Finally, if you are not satisfied with the result, you can slide the items and change the importance of the things.

You can also rebuild the data and in this way the app will try to resort and give a better management of the items.

2 The motivation of developing the app

2.1 The exist problems in our life

We are often overwhelmed by endless things. Especially in the university, there are infinity things to do. And the only way to do everything best is to try to manage time well.

However, many university students cannot manage time well. For example, some students cannot measure what is important, what should do immediately and what is less important. This can lead to a mess of a student's study, living and so on.

Unfortunately, in app store there are so many notebooks, but none of them can help a student to manage time. Most of them only give the deadline and how many days left.

2.2 Our solution to this situation

To deal with the situation I mentioned, we develop a sort system. This system sort by the "importance" of every things. Here "importance" considers many aspects such as time, that is the deadline, the things we should do("Math Exam" or "Throw Trash" and so on) and some words represents importance("immedately", "must to do").

After evaluate the importance of every things, the sort begins. After the sort, the most important things is put in the top. Then the user can do things according to the list.

However, maybe the user are not so satisfied with the sorting result. For example, I can throw my trash immediately but take exam later. At this point, users can drag the things to change the importance to make it more reasonable.

3 The function of the app

3.1 The evaluate system

The evaluate system gives each sentence a score. Every words in the sentence can affect the score of importance. The words include time, things, and other words and each word has a different score.

We use matching method to find the words. The orginal score of each sentence is 1. And every time we find a matching word, we multiply the score of the words(and typically, the score is greater than 1).

And after the matching process completed, every sentence has a score, and the score represents the importance the app calculated.

The source code shows below:

3.2 The sorting system

The sorting algorithm is quite simple. Since we have got every sentence a score, we can get every sentence's importance. We sort by importance, that is, sort by the score. The larger value will be put on the top.

We use the quick sort algorithm to sort the score. The source code shows below:

```java
  import java.util.*;
public class autoSort
{
        private static vitalTask[] storeTasks = new vitalTask[11];

        public static void loading()
        {
                storeTasks[0] = new vitalTask("      ",1.73);
                storeTasks[1] = new vitalTask("      ",1.65);
                storeTasks[2] = new vitalTask("      ",2.2);
                storeTasks[3] = new vitalTask("      ",2.0);
                storeTasks[4] = new vitalTask("      ",2.4);
                storeTasks[5] = new vitalTask("      ",1.77);
                storeTasks[6] = new vitalTask("      ",2.05);
                storeTasks[7] = new vitalTask("      ",1.16);
                storeTasks[8] = new vitalTask("      ",1.14);
                storeTasks[9] = new vitalTask("      ",1.75);
                storeTasks[10] = new vitalTask("         ",1.25);
        }

        public static void autosort(ArrayList<String> tasklist)
        {
                loading();
                oneOfTask[] list = new oneOfTask[tasklist.size()];

                for (int i = 0; i < tasklist.size(); i++)
                {
                        String tmp = tasklist.get(i);
                        list[i] = new oneOfTask(tmp,i);
                }

                for (int i = 0; i < tasklist.size(); i++)
                        autoSetValue(list[i]);

                SortTasks(list, 0, list.length);

                for (int i = 0; i < tasklist.size(); i++)
                {
                        tasklist.set(i, list[i].getName());
                }
        }

        public static void autoSetValue(oneOfTask atask)
        {
                String name = atask.getName();
                for (int i = 0; i < storeTasks.length; i++)
                        for (int j = 0; j < name.length() - storeTasks[i].getNa
                        {
                                String piecename = name.substring(j,j + storeTa
                                if (piecename.equals(storeTasks[i].getName()))
```

```
                                        atask.changeValue(storeTasks[i].getValu
                        }
        }

        public static void SortTasks(oneOfTask[] list, int begin, int end)
        {
                if (end - begin <= 1) return;
                else
                {
                        double tailnum = list[end-1].getValue();
                        oneOfTask tailobj = list[end - 1];
                        int index = begin;
                        for (int i = begin; i < end - 1; i++)
                        {
                                if (list[i].getValue() > tailnum)
                                {
                                        oneOfTask tmp1 = list[i];
                                        list[i] = list[index];
                                        list[index] = tmp1;
                                        index++;
                                }
                        }
                        for (int i = end - 1; i > index; i--)
                        {
                                list[i] = list[i - 1];
                        }
                        list[index] = tailobj;
                        SortTasks(list,begin,index);
                        SortTasks(list,index,end);
                }
        }

}
```

3.3 The drag algorithm

Our drag algorithm focus on the drag process. When you click on the item for a while, the item will be activated, then you can drag it over to other line. And of course, this operation can be done only after the sort.

The source code shows below:

```
import android.content.Context;
import android.os.Bundle;
import android.support.annotation.Nullable;
import android.support.v4.app.Fragment;
import android.support.v4.content.ContextCompat;
import android.support.v4.util.Pair;
import android.support.v4.widget.SwipeRefreshLayout;
import android.support.v7.app.AppCompatActivity;
import android.support.v7.widget.GridLayoutManager;
import android.support.v7.widget.LinearLayoutManager;
import android.view.LayoutInflater;
import android.view.Menu;
import android.view.MenuInflater;
import android.view.MenuItem;
import android.view.View;
import android.view.ViewGroup;
import android.widget.TextView;
import android.widget.Toast;

import com.woxthebox.draglistview.DragItem;
import com.woxthebox.draglistview.DragListView;
import com.woxthebox.draglistview.swipe.ListSwipeHelper;
import com.woxthebox.draglistview.swipe.ListSwipeItem;

import java.util.ArrayList;

public class ListFragment extends Fragment {

    private ArrayList<Pair<Long, String>> mItemArray;
    private DragListView mDragListView;
    private ListSwipeHelper mSwipeHelper;
    private MySwipeRefreshLayout mRefreshLayout;

    public static ListFragment newInstance() {
        return new ListFragment();
    }

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setHasOptionsMenu(true);
    }

    @Override
    public View onCreateView(LayoutInflater inflater, @Nullable ViewGroup conta
        View view = inflater.inflate(R.layout.list_layout, container, false);
        mRefreshLayout = (MySwipeRefreshLayout) view.findViewById(R.id.swipe_re
        mDragListView = (DragListView) view.findViewById(R.id.drag_list_view);
        mDragListView.getRecyclerView().setVerticalScrollBarEnabled(true);
        mDragListView.setDragListListener(new DragListView.DragListListenerAda
            @Override
            public void onItemDragStarted(int position) {
                mRefreshLayout.setEnabled(false);
                Toast.makeText(mDragListView.getContext(), "Start␣-␣position:␣"
            }

            @Override
            public void onItemDragEnded(int fromPosition, int toPosition) {
                mRefreshLayout.setEnabled(true);
                if (fromPosition != toPosition) {
                    Toast.makeText(mDragListView.getContext(), "End␣-␣position:
                }
            }
        });

        mItemArray = new ArrayList<>();
        String[] p = {"                    ", "          B   ppt", "
        for (int i = 0; i < 5; i++) {
            mItemArray.add(new Pair<>((long) i, p[i]));
        }

        mRefreshLayout.setScrollingView(mDragListView.getRecyclerView());
        mRefreshLayout.setColorSchemeColors(ContextCompat.getColor(getContext()
        mRefreshLayout.setOnRefreshListener(new SwipeRefreshLayout.OnRefreshLis
            @Override
            public void onRefresh() {
                mRefreshLayout.postDelayed(new Runnable() {
                    @Override
                    public void run() {
                        mRefreshLayout.setRefreshing(false);
                    }
                }, 2000);
            }
        });

        mDragListView.setSwipeListener(new ListSwipeHelper.OnSwipeListenerAdapt
            @Override
            public void onItemSwipeStarted(ListSwipeItem item) {
                mRefreshLayout.setEnabled(false);
            }

            @Override
            public void onItemSwipeEnded(ListSwipeItem item, ListSwipeItem.Swip
                mRefreshLayout.setEnabled(true);

                // Swipe to delete on left
                if (swipedDirection == ListSwipeItem.SwipeDirection.LEFT) {
                    Pair<Long, String> adapterItem = (Pair<Long, String>) item.
                    int pos = mDragListView.getAdapter().getPositionForItem(ada
                    mDragListView.getAdapter().removeItem(pos);
                }
            }
        });

        setupListRecyclerView();
        return view;
    }

    @Override
    public void onActivityCreated(@Nullable Bundle savedInstanceState) {
        super.onActivityCreated(savedInstanceState);
        ((AppCompatActivity) getActivity()).getSupportActionBar().setTitle("Ta
    }

    @Override
    public void onCreateOptionsMenu(Menu menu, MenuInflater inflater) {
        super.onCreateOptionsMenu(menu, inflater);
        inflater.inflate(R.menu.menu_list, menu);
    }

    @Override
    public void onPrepareOptionsMenu(Menu menu) {
        super.onPrepareOptionsMenu(menu);
        menu.findItem(R.id.action_disable_drag).setVisible(mDragListView.isDrag
        menu.findItem(R.id.action_enable_drag).setVisible(!mDragListView.isDrag
    }

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        switch (item.getItemId()) {
            case R.id.action_disable_drag:
                mDragListView.setDragEnabled(false);
                getActivity().supportInvalidateOptionsMenu();
                return true;
            case R.id.action_enable_drag:
                mDragListView.setDragEnabled(true);
                getActivity().supportInvalidateOptionsMenu();
                return true;
            case R.id.action_list:
                setupListRecyclerView();
                return true;
            case R.id.action_grid_vertical:
                setupGridVerticalRecyclerView();
                return true;
            case R.id.action_grid_horizontal:
                setupGridHorizontalRecyclerView();
                return true;
            case R.id.add_task:
                ADD_TASK();
                return true;
        }
        return super.onOptionsItemSelected(item);
    }

    private void ADD_TASK() {

    }//To Be Realized

    private void setupListRecyclerView() {
        mDragListView.setLayoutManager(new LinearLayoutManager(getContext()));
        ItemAdapter listAdapter = new ItemAdapter(mItemArray, R.layout.list_iter
        mDragListView.setAdapter(listAdapter, true);
        mDragListView.setCanDragHorizontally(false);
        mDragListView.setCustomDragItem(new MyDragItem(getContext(), R.layout.l
    }

    private void setupGridVerticalRecyclerView() {
        mDragListView.setLayoutManager(new GridLayoutManager(getContext(), 4));
        ItemAdapter listAdapter = new ItemAdapter(mItemArray, R.layout.grid_iter
        mDragListView.setAdapter(listAdapter, true);
        mDragListView.setCanDragHorizontally(true);
        mDragListView.setCustomDragItem(null);

    }

    private void setupGridHorizontalRecyclerView() {
        mDragListView.setLayoutManager(new GridLayoutManager(getContext(), 4, L
        ItemAdapter listAdapter = new ItemAdapter(mItemArray, R.layout.grid_iter
        mDragListView.setAdapter(listAdapter, true);
        mDragListView.setCanDragHorizontally(true);
        mDragListView.setCustomDragItem(null);
    }

    private static class MyDragItem extends DragItem {

        MyDragItem(Context context, int layoutId) {
            super(context, layoutId);
        }

        @Override
        public void onBindDragView(View clickedView, View dragView) {
            CharSequence text = ((TextView) clickedView.findViewById(R.id.text)
            ((TextView) dragView.findViewById(R.id.text)).setText(text);
            dragView.findViewById(R.id.item_layout).setBackgroundColor(dragView
        }
    }
}
import android.content.Context;
import android.os.Bundle;
import android.support.annotation.Nullable;
import android.support.v4.app.Fragment;
import android.support.v4.content.ContextCompat;
import android.support.v4.util.Pair;
import android.support.v4.widget.SwipeRefreshLayout;
import android.support.v7.app.AppCompatActivity;
import android.support.v7.widget.GridLayoutManager;
import android.support.v7.widget.LinearLayoutManager;
import android.view.LayoutInflater;
import android.view.Menu;
import android.view.MenuInflater;
import android.view.MenuItem;
import android.view.View;
import android.view.ViewGroup;
import android.widget.TextView;
import android.widget.Toast;

import com.woxthebox.draglistview.DragItem;
import com.woxthebox.draglistview.DragListView;
import com.woxthebox.draglistview.swipe.ListSwipeHelper;
import com.woxthebox.draglistview.swipe.ListSwipeItem;

import java.util.ArrayList;

public class ListFragment extends Fragment {

    private ArrayList<Pair<Long, String>> mItemArray;
    private DragListView mDragListView;
    private ListSwipeHelper mSwipeHelper;
    private MySwipeRefreshLayout mRefreshLayout;

    public static ListFragment newInstance() {
        return new ListFragment();
    }

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setHasOptionsMenu(true);
    }

    @Override
    public View onCreateView(LayoutInflater inflater, @Nullable ViewGroup conta
        View view = inflater.inflate(R.layout.list_layout, container, false);
        mRefreshLayout = (MySwipeRefreshLayout) view.findViewById(R.id.swipe_re
        mDragListView = (DragListView) view.findViewById(R.id.drag_list_view);
        mDragListView.getRecyclerView().setVerticalScrollBarEnabled(true);
        mDragListView.setDragListListener(new DragListView.DragListListenerAda
            @Override
            public void onItemDragStarted(int position) {
                mRefreshLayout.setEnabled(false);
                Toast.makeText(mDragListView.getContext(), "Start␣−␣position:␣"
            }

            @Override
            public void onItemDragEnded(int fromPosition, int toPosition) {
                mRefreshLayout.setEnabled(true);
                if (fromPosition != toPosition) {
                    Toast.makeText(mDragListView.getContext(), "End␣−␣position:
                }
            }
        });

        mItemArray = new ArrayList<>();
        String[] p = {"                 ", "           B  ppt", "
        for (int i = 0; i < 5; i++) {
            mItemArray.add(new Pair<>((long) i, p[i]));
        }

        mRefreshLayout.setScrollingView(mDragListView.getRecyclerView());
        mRefreshLayout.setColorSchemeColors(ContextCompat.getColor(getContext()
        mRefreshLayout.setOnRefreshListener(new SwipeRefreshLayout.OnRefreshLis
            @Override
            public void onRefresh() {
                mRefreshLayout.postDelayed(new Runnable() {
                    @Override
                    public void run() {
                        mRefreshLayout.setRefreshing(false);
                    }
                }, 2000);
            }
        });

        mDragListView.setSwipeListener(new ListSwipeHelper.OnSwipeListenerAdapt
            @Override
            public void onItemSwipeStarted(ListSwipeItem item) {
                mRefreshLayout.setEnabled(false);
            }

            @Override
            public void onItemSwipeEnded(ListSwipeItem item, ListSwipeItem.Swip
                mRefreshLayout.setEnabled(true);

                // Swipe to delete on left
                if (swipedDirection == ListSwipeItem.SwipeDirection.LEFT) {
                    Pair<Long, String> adapterItem = (Pair<Long, String>) item.
                    int pos = mDragListView.getAdapter().getPositionForItem(ada
                    mDragListView.getAdapter().removeItem(pos);
                }
            }
        });

        setupListRecyclerView();
        return view;
    }

    @Override
    public void onActivityCreated(@Nullable Bundle savedInstanceState) {
        super.onActivityCreated(savedInstanceState);
        ((AppCompatActivity) getActivity()).getSupportActionBar().setTitle("Ta
    }

    @Override
    public void onCreateOptionsMenu(Menu menu, MenuInflater inflater) {
        super.onCreateOptionsMenu(menu, inflater);
        inflater.inflate(R.menu.menu_list, menu);
    }

    @Override
    public void onPrepareOptionsMenu(Menu menu) {
        super.onPrepareOptionsMenu(menu);
        menu.findItem(R.id.action_disable_drag).setVisible(mDragListView.isDrag
        menu.findItem(R.id.action_enable_drag).setVisible(!mDragListView.isDrag
    }

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        switch (item.getItemId()) {
            case R.id.action_disable_drag:
                mDragListView.setDragEnabled(false);
                getActivity().supportInvalidateOptionsMenu();
                return true;
            case R.id.action_enable_drag:
                mDragListView.setDragEnabled(true);
                getActivity().supportInvalidateOptionsMenu();
                return true;
            case R.id.action_list:
                setupListRecyclerView();
                return true;
            case R.id.action_grid_vertical:
                setupGridVerticalRecyclerView();
                return true;
            case R.id.action_grid_horizontal:
                setupGridHorizontalRecyclerView();
                return true;
            case R.id.add_task:
                ADD_TASK();
                return true;
        }
        return super.onOptionsItemSelected(item);
    }

    private void ADD_TASK() {

    }//To Be Realized

    private void setupListRecyclerView() {
        mDragListView.setLayoutManager(new LinearLayoutManager(getContext()));
        ItemAdapter listAdapter = new ItemAdapter(mItemArray, R.layout.list_iten
        mDragListView.setAdapter(listAdapter, true);
        mDragListView.setCanDragHorizontally(false);
        mDragListView.setCustomDragItem(new MyDragItem(getContext(), R.layout.l
    }

    private void setupGridVerticalRecyclerView() {
        mDragListView.setLayoutManager(new GridLayoutManager(getContext(), 4));
        ItemAdapter listAdapter = new ItemAdapter(mItemArray, R.layout.grid_iter
        mDragListView.setAdapter(listAdapter, true);
        mDragListView.setCanDragHorizontally(true);
        mDragListView.setCustomDragItem(null);

    }

    private void setupGridHorizontalRecyclerView() {
        mDragListView.setLayoutManager(new GridLayoutManager(getContext(), 4, I
        ItemAdapter listAdapter = new ItemAdapter(mItemArray, R.layout.grid_iter
        mDragListView.setAdapter(listAdapter, true);
        mDragListView.setCanDragHorizontally(true);
        mDragListView.setCustomDragItem(null);
    }

    private static class MyDragItem extends DragItem {

        MyDragItem(Context context, int layoutId) {
            super(context, layoutId);
        }

        @Override
        public void onBindDragView(View clickedView, View dragView) {
            CharSequence text = ((TextView) clickedView.findViewById(R.id.text)
            ((TextView) dragView.findViewById(R.id.text)).setText(text);
            dragView.findViewById(R.id.item_layout).setBackgroundColor(dragView
        }
    }
}
```

4 The feature of the app

We try our best to make the evaluate system more reasonable, so up to now it can evaluate how important the thing is very well. Only the evaluation system be more reasonable can we sort well and give the more reasonable solution.

As for the sort algorithm, we use the typical quick sort algorithm. It is more efficient than bubble sort.

The most important study is about the drag opration. We make the drag as smooth as possible, which gives the user a better and more convenient use.

5 The things we want to do next

We are going to make the app much more convenient, not just for this experiment. So next we are going to update in the following aspects:

5.1 The more beautiful UI

As at the first time we are try our best developing our algorithm, we are not focus as much on the UI and the interface.

So next time, we are going to develop a more beautiful UI and interface to satisfy the user's demand.

5.2 The learning system

Today our evaluate system is based on dictionary, that is, use dictionary to regulate the score of each words. However, this is not always the truth as the demand for each user is different.

So we are going to change our evaluate system into a data-based system, using users' sort and the big data to analyse what is more important.

6 The feeling about the experiment

From this experiment, we learn how to develop an app in the mobile phone, and try to solve the problems in our daily life.

At first time, we are not very clear about how to develop, but then we try to learn and try by hand, and finally we are able to manage some basic operation in Android Studio.

And in the experiment, we came across some difficulties. However, we didn't give up and try to find the solutions. Finally, we developed what we want. This is a great experience.

The experiment is coming to an end, but our development and study is not. So it is just a beginning to go and it encourages us to develop greater app in the future.