Contacts Design Document

Document Template

Documentation Support

Common Platform

Doc No:
Version: A1
Release date: 2016-11-17
Classification: Internal

Keywords
Template, System Design Document

MediaTek Inc.

Postal address
No. 1, Dusing 1st Rd. , Hsinchu Science Park, Hsinchu City, Taiwan 30078

MTK support office address
No. 1, Dusing 1st Rd. , Hsinchu Science Park, Hsinchu City, Taiwan 30078

Internet
http://www.mediatek.com/

Document Revision History

Revision	Date	Author	Description
A1	2016-11-17	Qinghua Liu	Review and Change some statement

Table of Contents

Document Revision History **3**
Table of Contents **4**
Lists of Tables **7**
Lists of Figures **8**
1 Introduction **9**
1.1 Purpose 9
1.2 Scope 9
1.3 Who Should Read This Document 9
1.4 How to Use This Manual 9
1.4.1 Terms and Conventions 10
2 References **11**
3 Definitions **12**
4 Abbreviations **13**
5 Overview **14**
5.1 Overall Structure 14
5.2 Contacts UI 14
5.2.1 People Main 15
5.2.2 Editor&AAS/SNE 16
5.2.3 Import/Export 18
5.2.4 Group 18
5.2.5 Quick Contact 20
5.2.6 Multi Operation 21
5.2.7 Share/VCard 21
5.2.8 Multi-Window 22
5.2.9 RunTime Permission 22
6 Class **23**
6.1 Common Classs 23
6.2 Submodule Classs 23
6.2.1 People Main 24

6.2.2 Editor&AAS/SNE 25
6.2.3 Import/Export 26
6.2.4 Group 28
6.2.5 Multi Operation 30
6.2.6 Runtime Permission 32
6.2.7 SimProcessor 32
7 Sequence 34
7.1 People Main 34
7.1.1 UI display flow 34
7.1.2 Configure Adapter 35
7.1.3 Search contact flow 35
7.2 Editor&AAS/SNE 36
7.2.1 load account flow 36
7.2.2 contruct UI flow 37
7.2.3 mState data filled in 37
7.2.4 bindEditor 39
7.2.5 save contact flow 40
7.2.6 Contruct AAS UI 41
7.3 Import/Export 43
7.3.1 UI flow 43
7.3.2 Import storage account into phone account flow 44
7.3.3 sim copy to phone flow 45
7.4 Group 46
7.4.1 New A Group 46
7.4.2 Delete A Group 46
7.4.3 Move Group Members 47
7.4.4 Add Contacts to Group 48
7.4.5 Remove Contacts from Group 49
7.5 QuickContact 49
7.5.1 Build QuickContact principle 49

7.5.2 QuickContact event response 50
7.6 Multi Operation 51
7.6.1 Google multiSelection 51
7.6.2 Mtk multiSelection 53
7.7 Share/VCard 55
7.7.1 share multi contacts flow in PeopleMain 55
7.7.2 QuickContactsActivity share flow 56
7.7.3 export Vcard 56
7.7.4 import Vcard 57
7.8 Multi-Window 58
7.8.1 Enable/disable Multi-window feature for app 58
7.8.2 Life-cycle 58
7.9 RunTime Permission 59
7.9.1 Contacts Permission encapsulation 59
7.9.2 RequestPermisionsActivity work flow 60
7.10 SimProcessor 60
7.10.1 Start flow 60
7.10.2 Handle phone book change 61
7.10.3 Handle boot completed 62
7.10.4 Refesh sim contacts 63

Lists of Tables

Table 4-1. Abbreviations 13

Lists of Figures

1 Introduction

1.1 Purpose

This document describes the detail design of Contacts application. The scope covers Contacts, ContactsCommon and ContactsProvider. Contacts is the UI application to the end user. ContactsCommon is a source package used by Contacts and Dialer. ContactsProvider provider the data and query API for Contacts and other application.

1.2 Scope

This design document is based on Android N and higher version.

1.3 Who Should Read This Document

This document is primarily intended for:

- Engineers with technical knowledge of Contacts
- Customers who integrate Contacts with user-defined applications

1.4 How to Use This Manual

This segment explains how information is distributed in this document, and presents some cues and examples to simplify finding and understanding information in this document. Table 1- presents an overview of the chapters and appendices in this document.

Table 1-1. Chapter Overview

#	Chapter	Contents
1	Introduction	Describes the scope and layout of this document.
2	References	Describes the reference documents of this document.
3	Definitions	List the definitions used in this document.
4	Abbreviations	List the abbreviations used in this document.
5	Overview	Describes the Contacts overall flow from UI to Database.
6	Class	Describes the main Contacts class.
7	Sequence	Describes the main Contacts work flow.

1.4.1 Terms and Conventions

This document uses special terms and typographical conventions to help you easily identify various information types in this document. These cues are designed to simply finding and understanding the information this document contains.

2 References

The following documents contain provisions which, through reference in this text, constitute provisions of the present document.

[1] MTK Company Profile, http://brandclips.mediatek.inc/uploads/Company-profile-1H-2016_0418-Lite-final.pptx

[2] MTK Word Template, http://brandclips.mediatek.inc/uploads/Microsoft-Office-Word-Oct-2014.rar

3 Definitions

For the purposes of the present document, the following terms and definitions apply:

4 Abbreviations

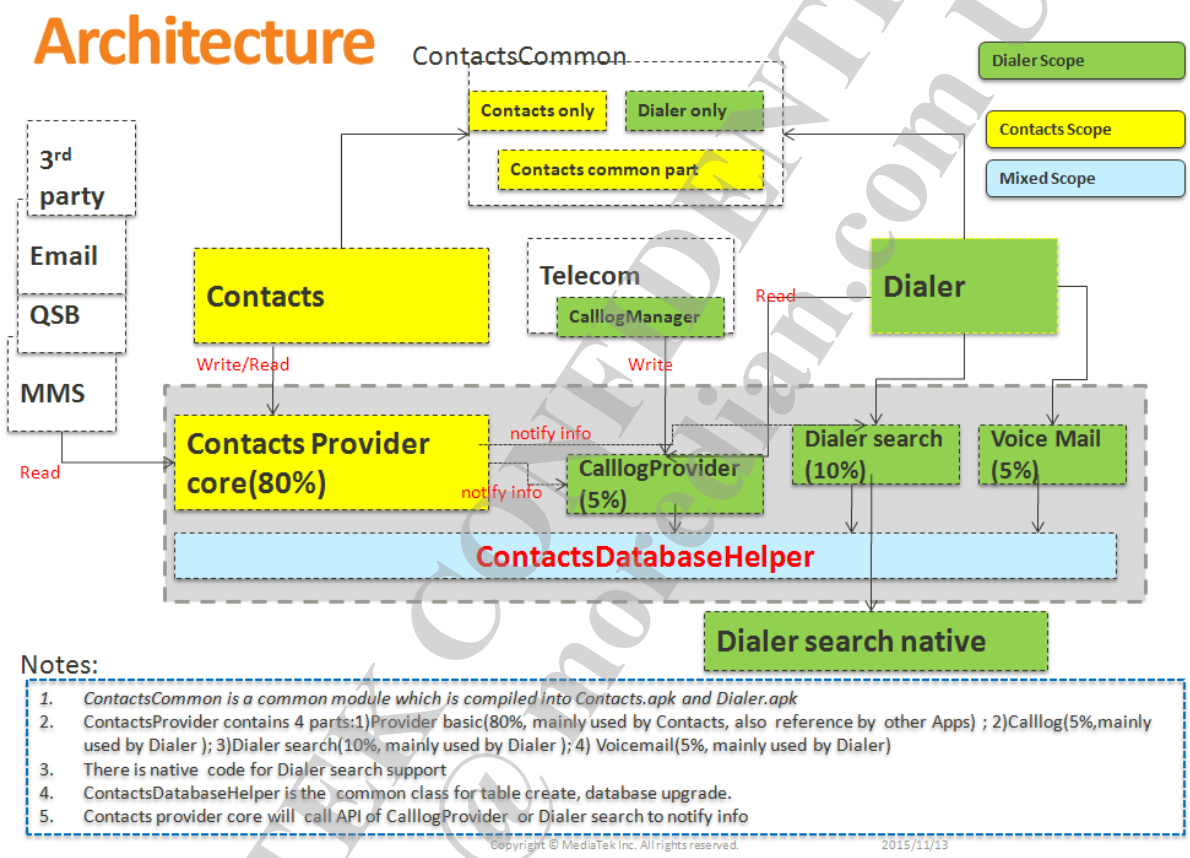
Please note the abbreviations and their explanations provided in Table 4-1. They are used in many fundamental definitions and explanations in this document and are specific to the information that this document contains.

Table 4-1. Abbreviations

Abbreviations	Explanation
AAS	Additional number alpha String
SNE	Second Name Entry
SDN	Service dialing number
USIM	Universal Subscriber Identity Module
PBR	Phone book reference

5 Overview

5.1 Overall Structure



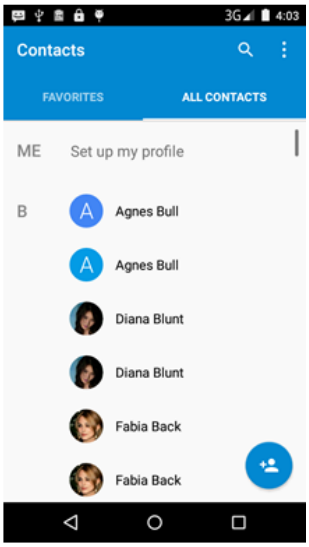
Contacts Feature Overall Structure

5.2 Contacts UI

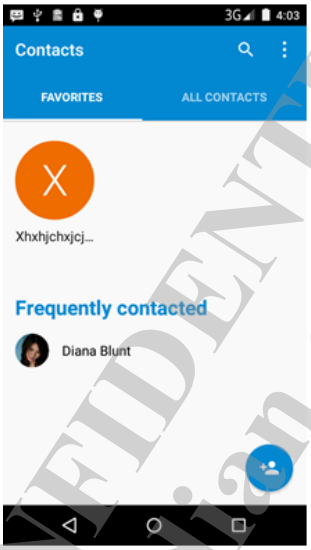
Contacts application mainly contains the following submodules:

People Main, Editor&AAS/SNE, Import/Export, Group, QuickContact, Multi Operation, Share/VCard, Multi-Window, RunTime Permission, SimProcessor.

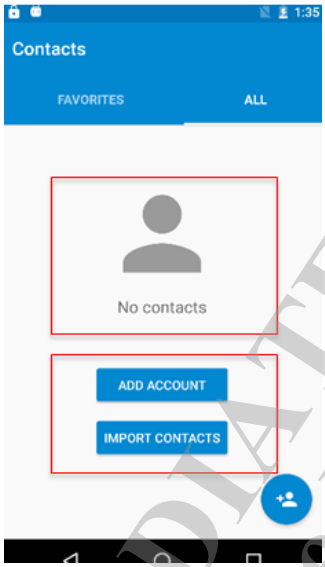
5.2.1 People Main



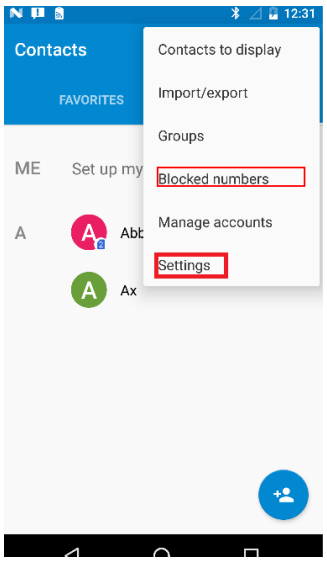
mAllFragment

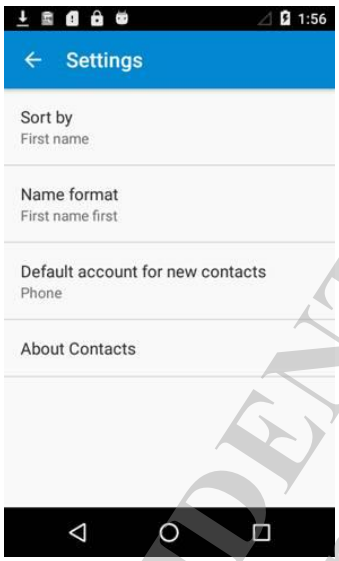


mFavoritesFragment



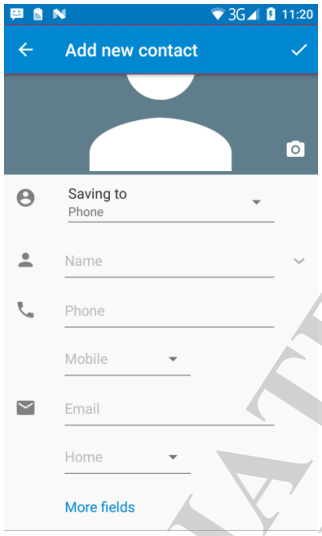
mContactsUnavailableFragment: when no contacts, show two default button



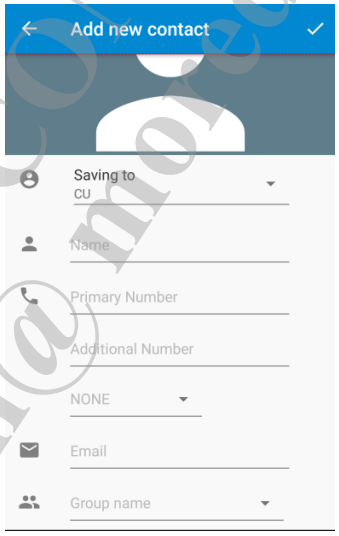


People main option menu(Blocked numbers and Settings are added on Android N)

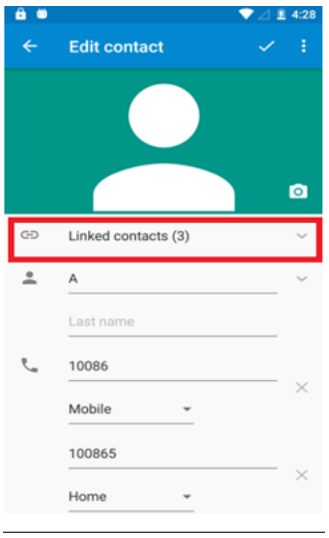
5.2.2 Editor&AAS/SNE

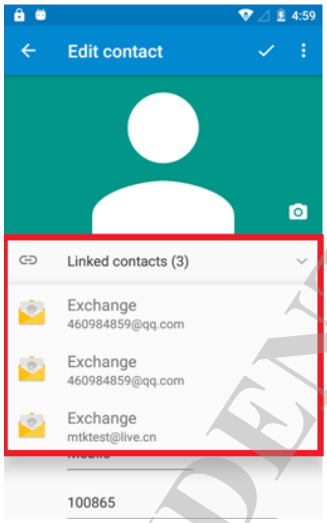


Add a phone contact

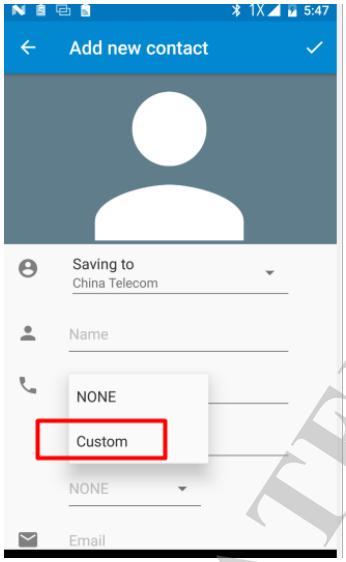


Add a usim contact





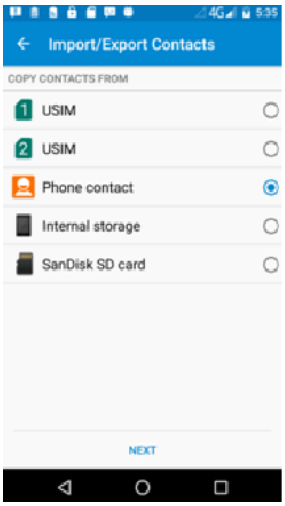
One contact merged by three contacts

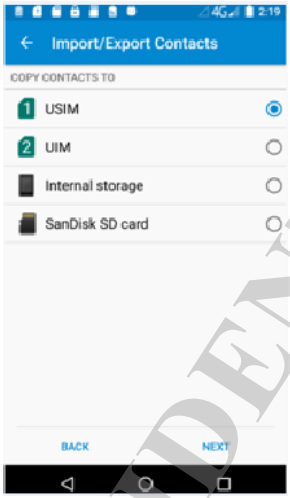


Add a custom label for contact

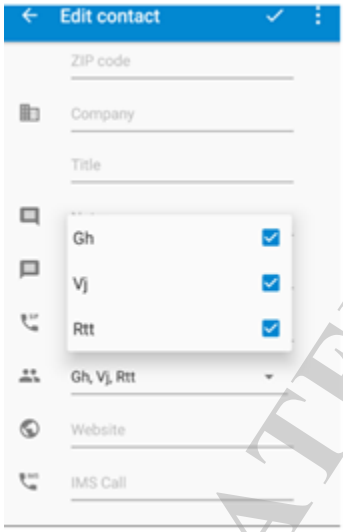
Select or editor custom label

5.2.3 Import/Export





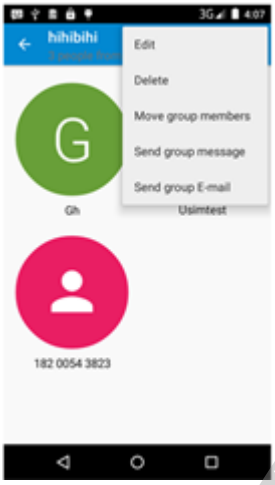
5.2.4 Group



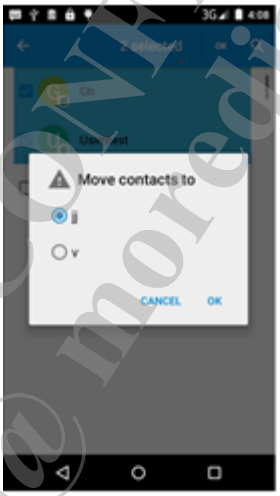
Create a contact with group info

GroupBrowseActivity

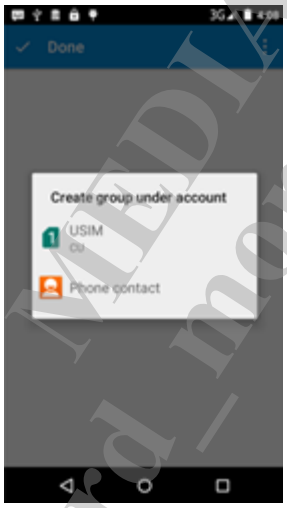
GroupDetailActivity

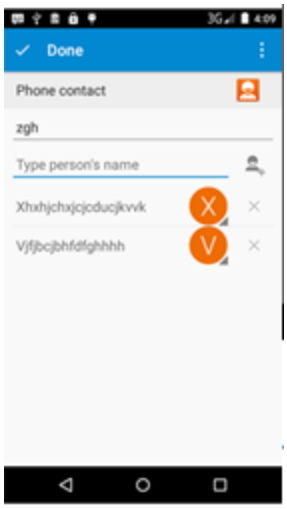


GroupDetailActivity OptionMenu



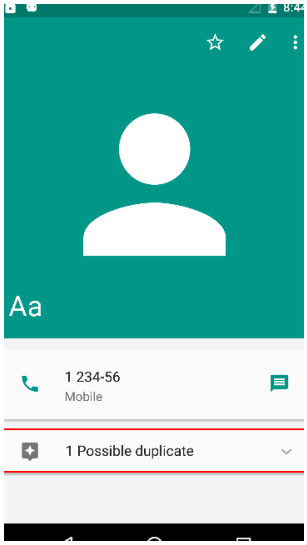
MultiGroupPickerFragment

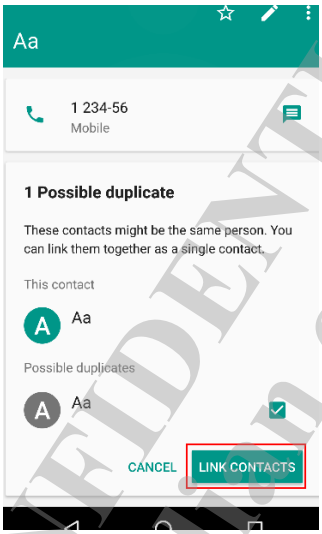


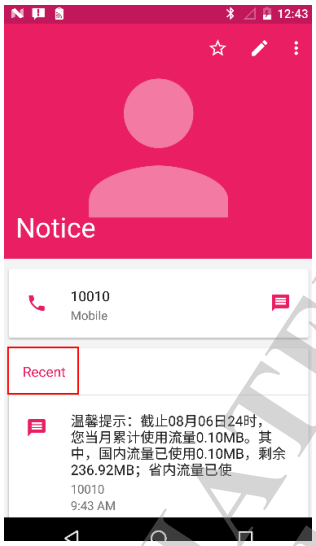


SelectAccountDialogFragment GroupEditorActivity

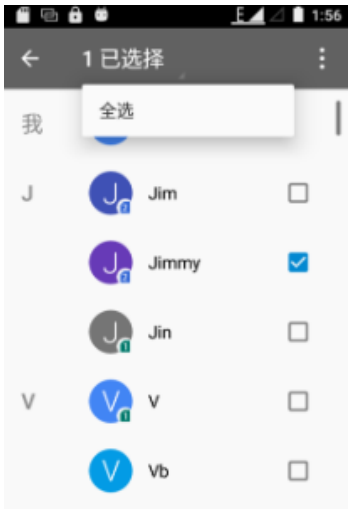
5.2.5 Quick Contact





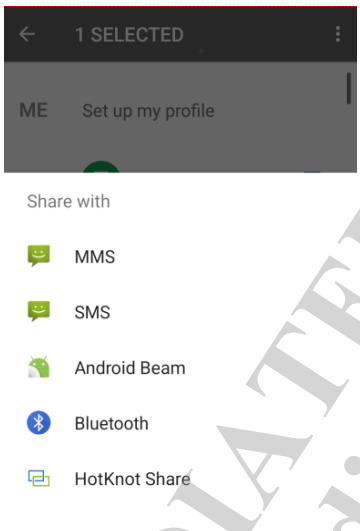


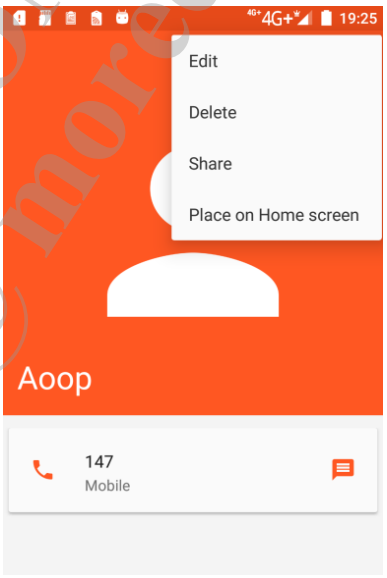
5.2.6 Multi Operation



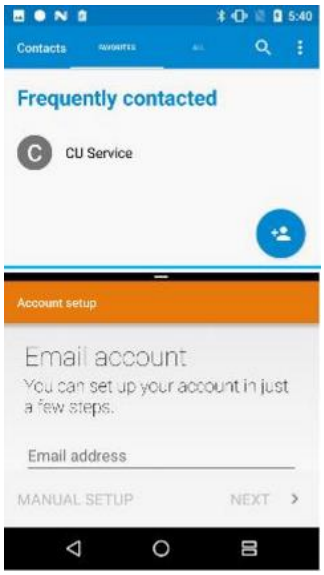
Multi select contacts

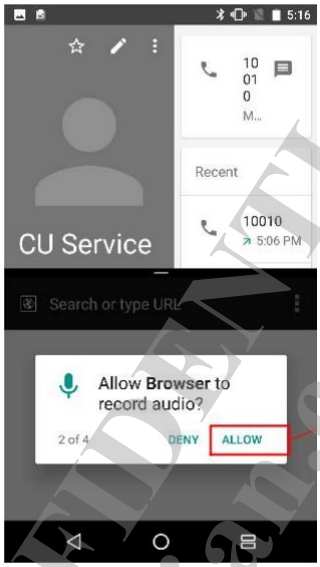
5.2.7 Share/VCard



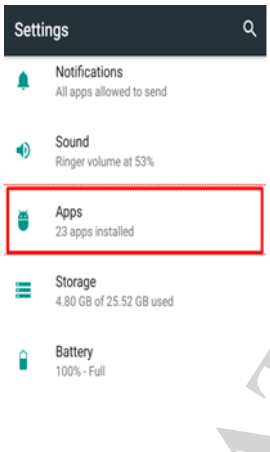


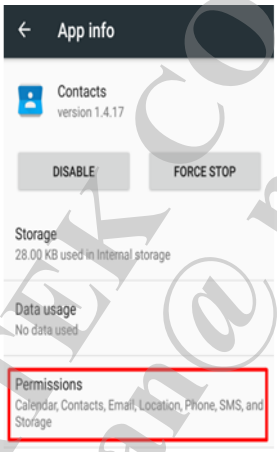
5.2.8 Multi-Window

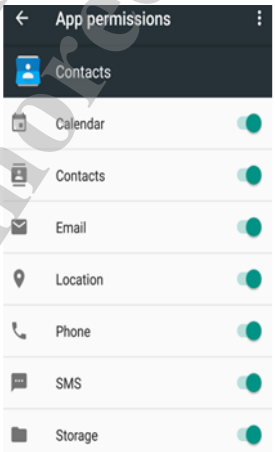


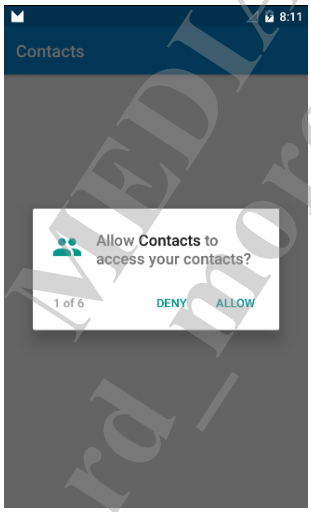


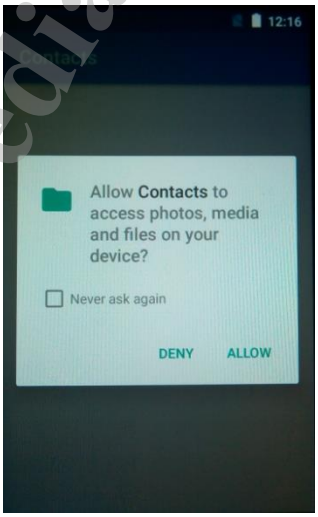
5.2.9 RunTime Permission







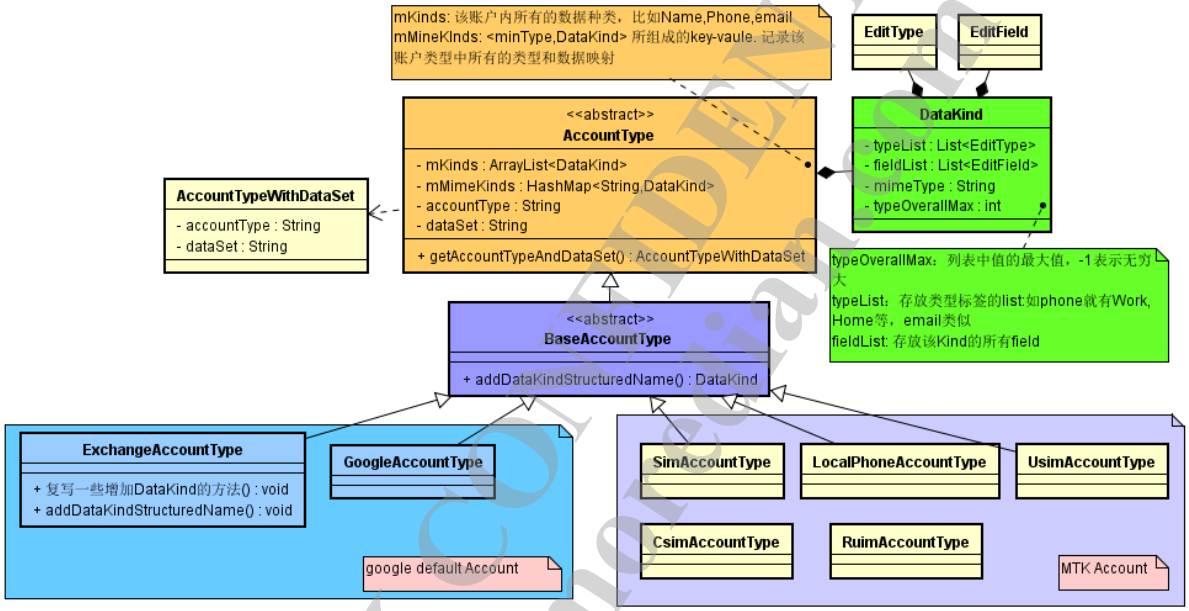




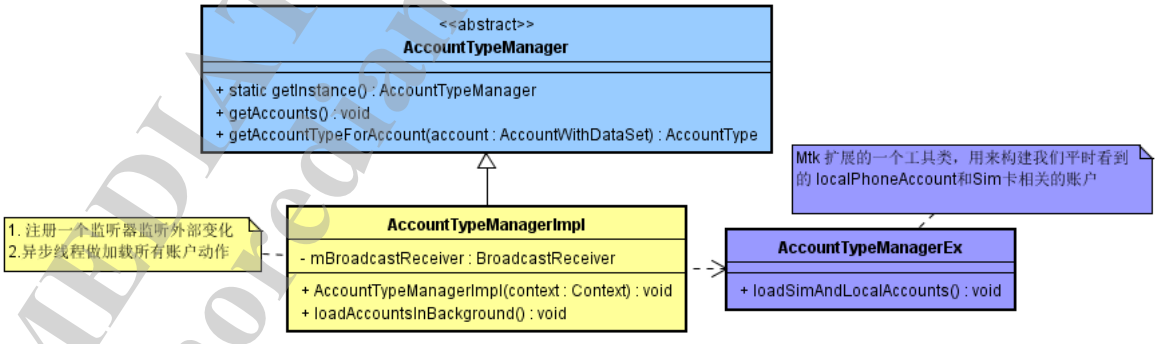
6 Class

6.1 Common Classs

Firstly, I introduce some common Class in Contacts and ContactsCommon.



Account type used in Contacts



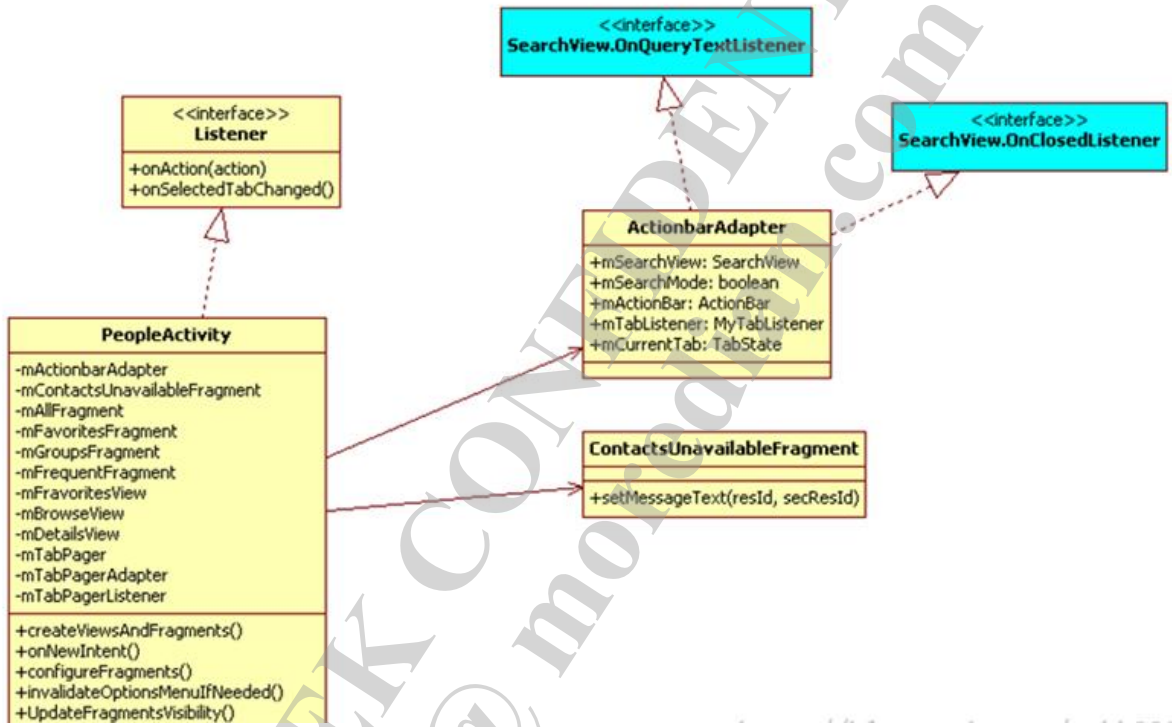
AccountTypeManager : Manage the accounts that contacts use

6.2 Submodule Classs

Contacts application mainly contains the following submodules:

People Main, Editor&AAS/SNE, Import/Export, Group, Multi Operation, Share/VCard, Multi-Window, RunTime Permission, SimProcessor.

6.2.1 People Main



Key variable

DefaultContactBrowseListFragment mAllFragment - all contacts

ContactTileListFragment mFavoritesFragment - My favorite contacts

ContactsUnavailableFragment mContactsUnavailableFragment - Indicates the Fragment that is not currently available.

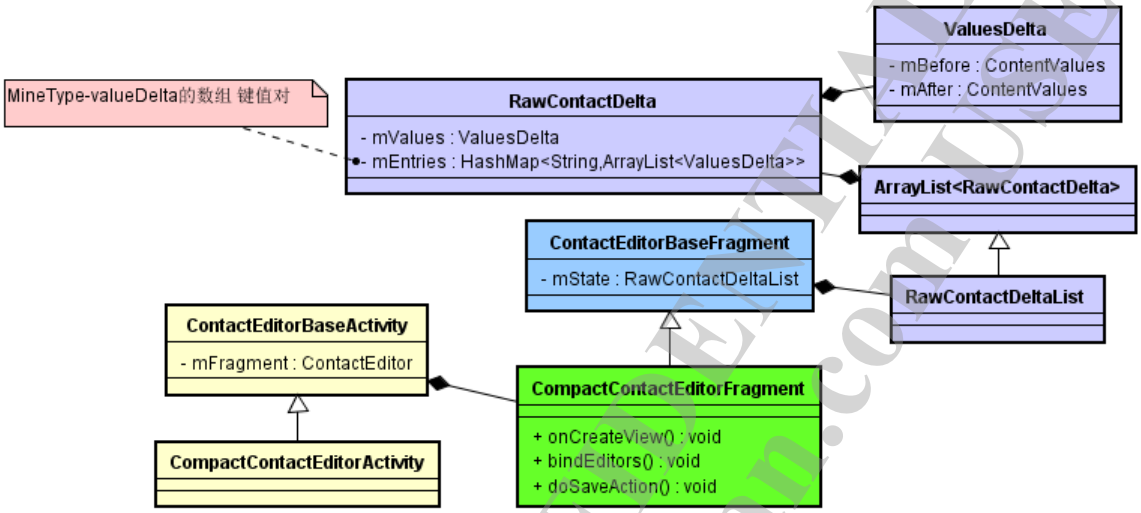
The listener is installed in PeopleActivity.onAttachFragment ()

TabPagerAdapter mTabPagerAdapter - Used to place each fragment

TabPagerListener mTabPagerListener - The event used to listen for the tag's scroll and select

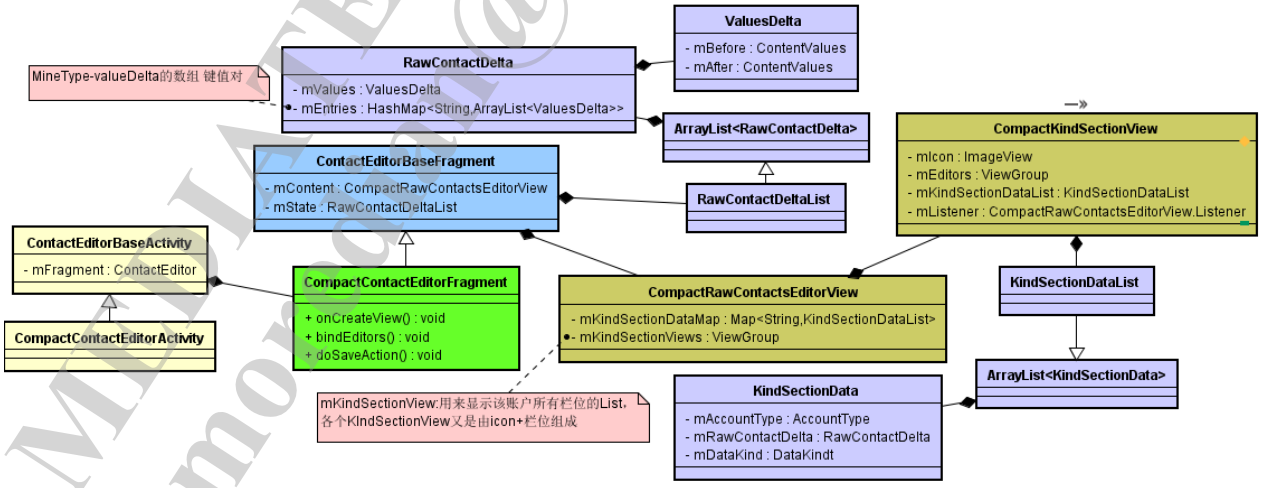
MAllFragment is an instance of DefaultContactBrowseListFragment, which is the "all contacts" fragment.

6.2.2 Editor&AAS/SNE



The data storage of the contact needs to have a data structure corresponding to the UI. When the user fills in the information with Ui, the data of the corresponding point will be stored correctly in the database. So, in the Fragment need to maintain such a record all the user operation of the data, here is ContactEditorBaseFragment: mState to complete this matter.

MState data structure as shown above, it is an array of RawContactDelta, and ultimately some key-vaule composition, it should be noted that the ValueDelta by two ContentValues constitute (mBefore, mAfter), the purpose of this design in order to have a contact Information needs to be updated and restored when used.

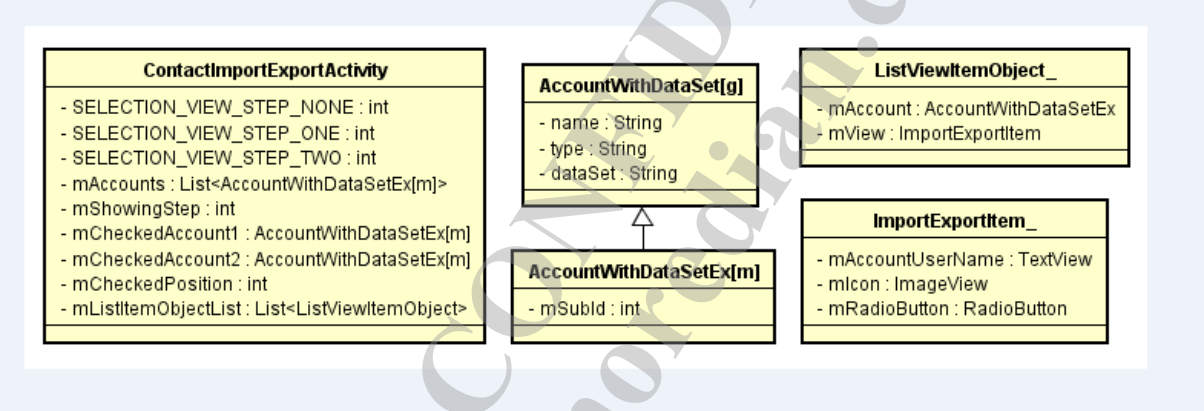


CompactEditorBaseActivity uses the CompactContactEditorFragment to render the UI, which has a CompactRawContactdEditorView, which is an XMl layout layout that is loaded in the Fragment's onCreateView, which defines the base structure frame, an avatar domain, an account selection field, a field field , A more field field. From this see CompactRawContactdEditorView is almost the main body of the UI, which can be changed

in the "all field", because different accounts, can support the field is different. CompactContactEditorFragment There is also a mState data class, this is also an integrated parent class, the addition of this data class will be based on the current account type (account in the new time, the default set the account available DataKind and MineType key DataKind map) To create.

The more complex part is the mKIndSectionViews in the CompactRawContactdEditorView. The ViewGroup is a list of CompactKindSectionView, which consists of fields of different data types, such as the name field, the field of the phone number, and the field of the mailbox. Each field of data is represented by a CompactKindSectionView, so that multiple CompactKindSectionView together form a List, forming all the data fields of the account.

6.2.3 Import/Export



UI display involved in the data structure as shown above, the specific field analysis as follows:

ContactImportExportActivity

MListView: used to display the items of each item, the entry of each Item of the data is also displayed by the ListViewItemObject

MAcounts: store all the accounts in the phone, such as sim card accounts, storage accounts, etc., the structure for the List <AccountWithDataSetEx>, AccountWithDataSetEx for the mtk self-expansion to google AccountWithDataSet, the main increase subId.

MShowingStep: used to control the current display steps, initialized to SELECT_VIEW_STEP_NONE

MCheckAccount1, mCheckAccount2: Indicates the source and destination accounts, respectively

MChcekedPosition: used to indicate the current interface to select the location of the account

MListItemObjectList: used to display the page needs to display the account, such as filtered by the filter rules account

Note: mAccounts records the current device can use the account, and mListItemObjectList is to record the current page needs to display the account, the realization of the process is through the filter rules, traverse mAccounts, and then the appropriate AccountWithDataSetEx combination listViewItemObject, into the mListItemObjectList , ListView in the show, is to use this data source to display the account name

AccountWithDataSet

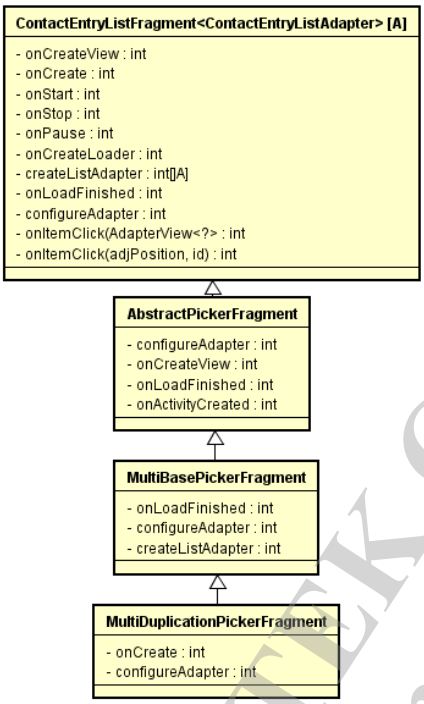
Name: account name, type: account type, dataset: carry data

AccountWithDataSetEx

Extended subId

ImportExportItem

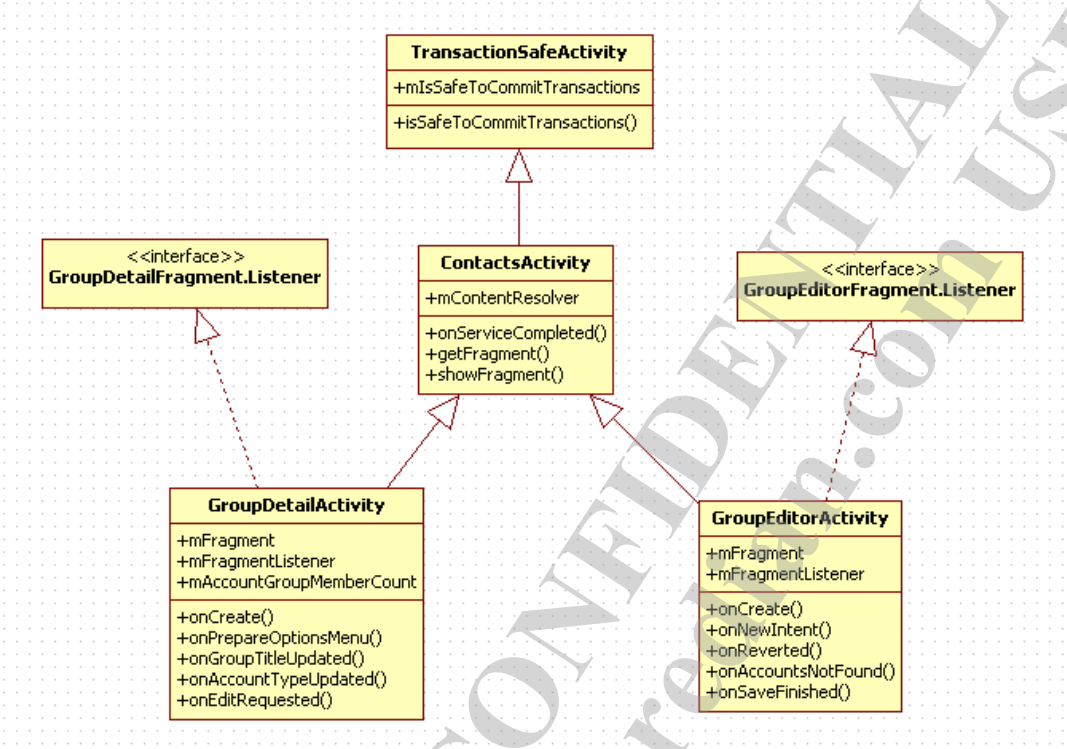
ListView Item display item, to the listView Item to provide data



MultiDuplicationPickerFragment construction process:

The base class is responsible for the construction of the main body, responsible for the entire interface scheduling, the most important is createAdapter, configAdapter, configure the adapter configLoader method, these methods will complete the Adapter, loader creation and configuration, providing asynchronous query, each sub-Fragment according to the differences between the various interfaces The configuration of the class diagram as shown above.

6.2.4 Group

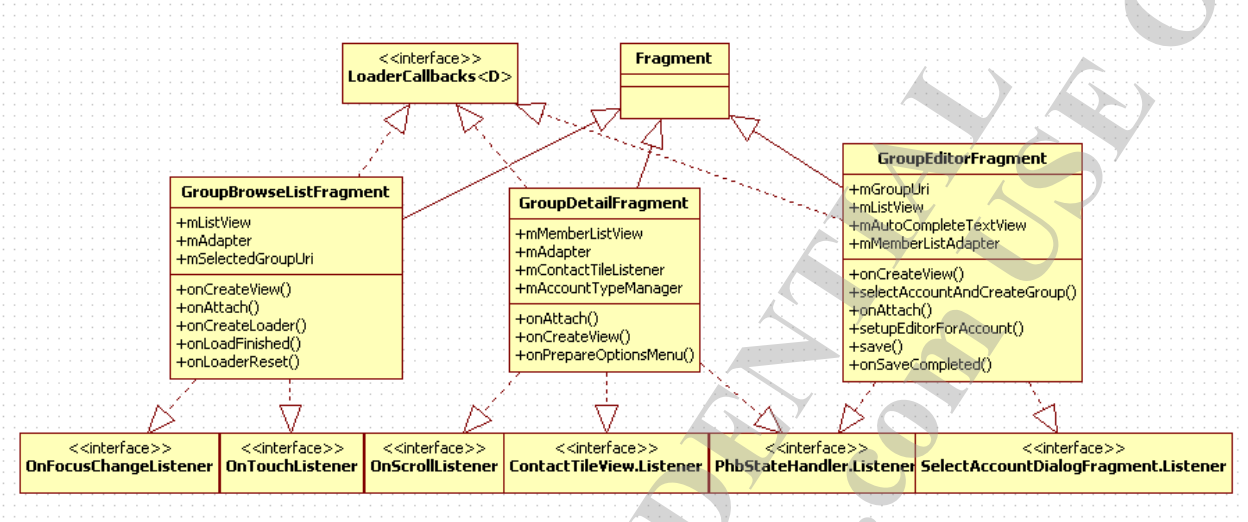


TransactionSafeActivity and **ContactsActivity**: Activity in the base class in Contacts, which provides a public method of the Activity of Contacts that is actually used.

GroupBrowseActivity: This Activity is a ListActivity, in the upper right of the Activity is a "Add Group" button, the bottom is a ListView, which lists all the groups.

GroupEditorActivity: This Activity can edit a Group, including modifying the Group's Name, adding or deleting members of the Group.

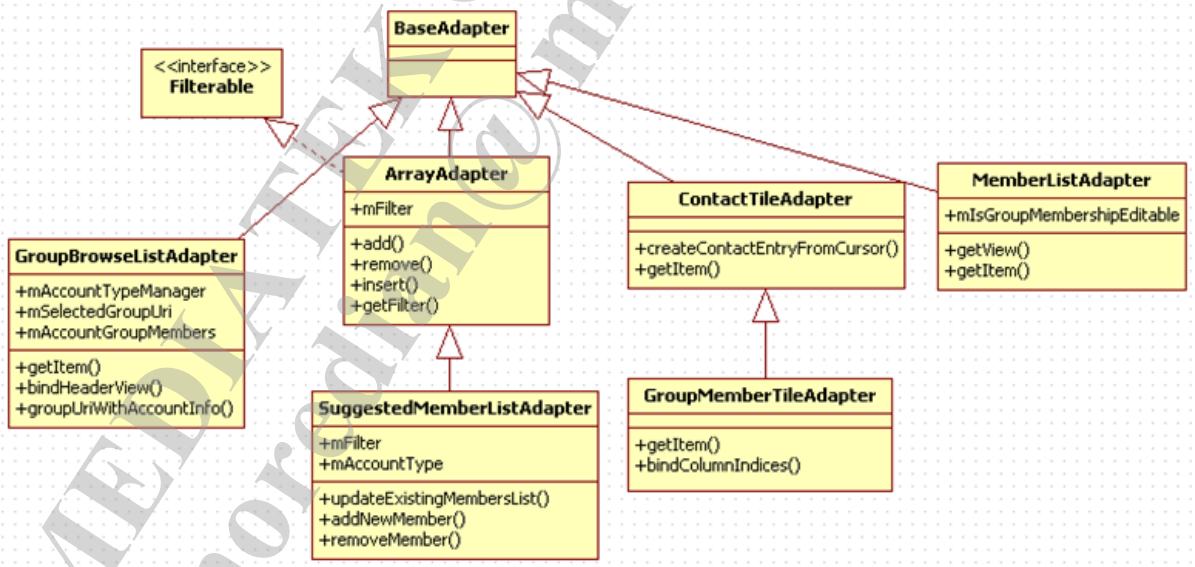
GroupDetailActivity: This Activity lists all members of a Group and can send messages and email to members of this group, as well as mobile group members.



GroupBrowseListFragment: GroupFrowseActivity added Fragment, mainly contains a ListView, lists all the groups.

GroupEditorFragment: The Fragment added in GroupEditorActivity, which contains the EditorText for modifying the Group Name and a ListView for the Group Member to be added.

GroupDetailFragment: GroupDetailActivity added Fragment, mainly contains a ListView, which lists a group of all members.



GroupBrowseListAdapter: GroupBrowseListFragment corresponding Adapter, mainly contains a ListView, lists all the groups.

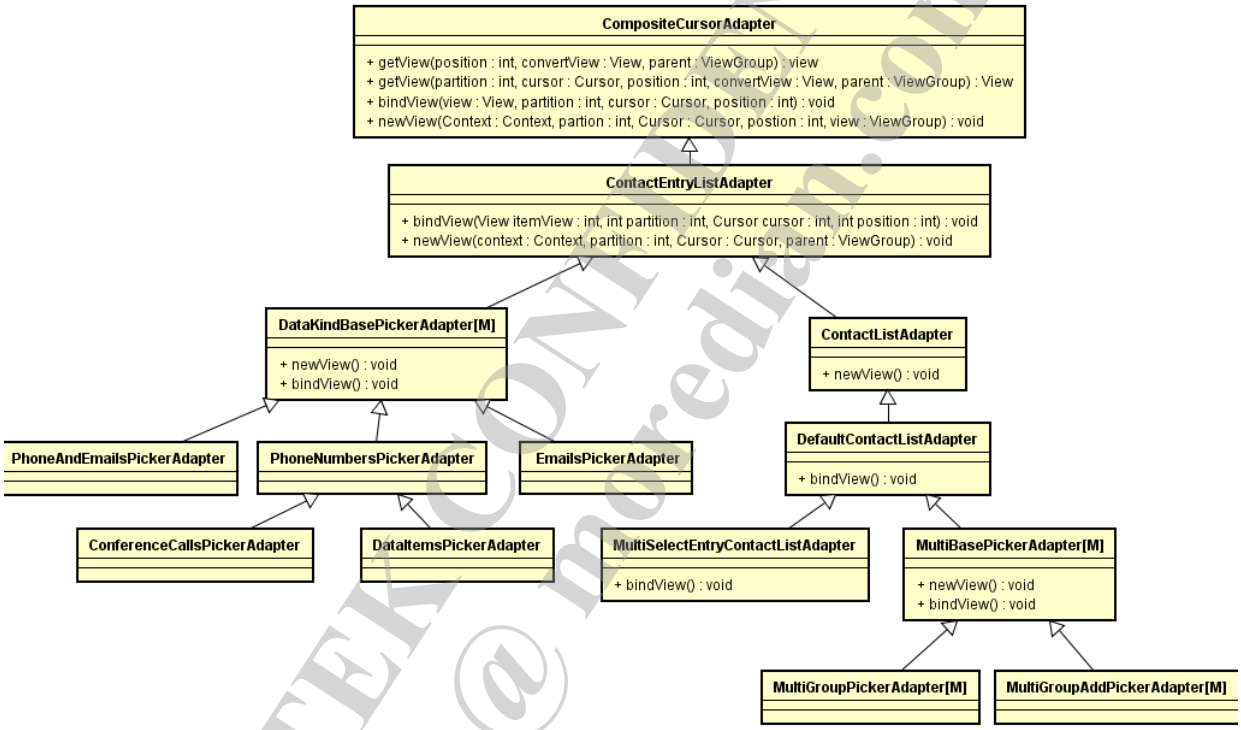
GroupMemberTileAdapter: GroupDetailFragment added in the Adapter, mainly used to display a list of all members of the Group.

MemberListAdapter: Adapter in GroupEditorFragment, used to display the ListView of the Group Member to be added.

SuggestMemberListAdapter: Adapter in the GroupEditorFragment, used to display the Suggested Contacts list for String and Search when adding the Group Member.

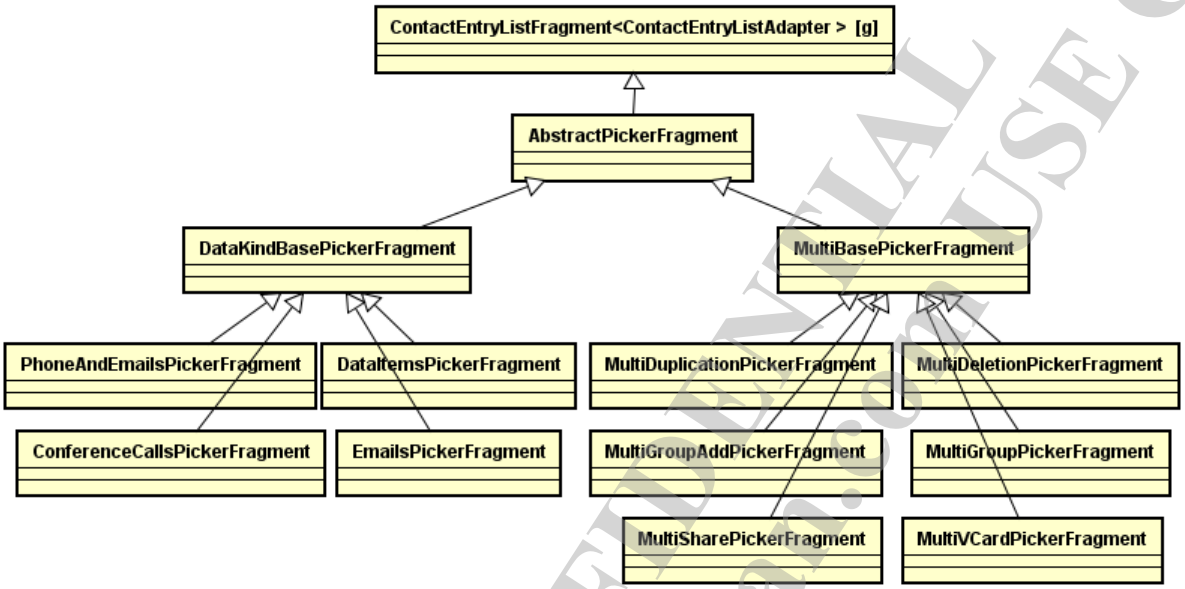
6.2.5 Multi Operation

Google multiSelection:



The top of the base class is CompositeCursorAdapter, which is a multi-cursor can contain a number of adapter, the middle there is a lot of inheritance adapter, the middle of some of the map is not marked. The interaction between ListView and Adapter depends on the getView () in the top-level class CompositeCursorAdapter to the virtual class method newView, the bindView method. The concrete implementation is done by the subclass. The inheritance relationship is shown in the figure above.

Mtk multiSelection：

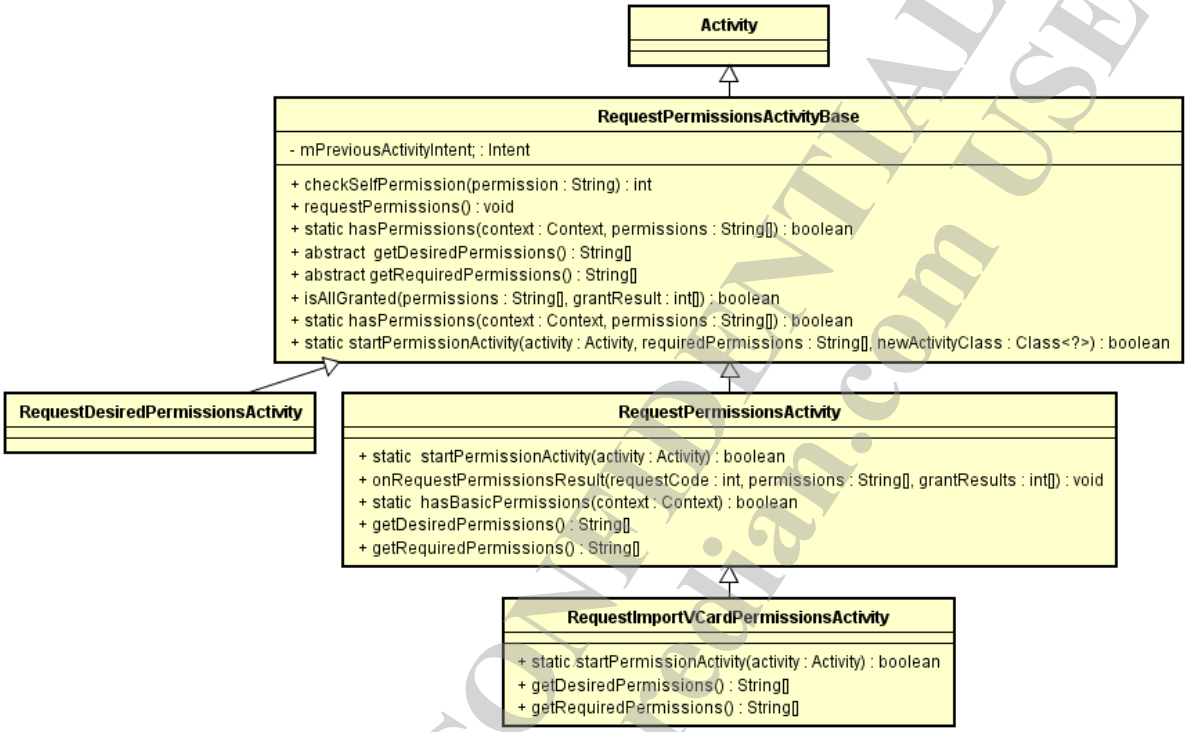


DataKindBasePIckerFragment differs from the design of MultiBasePickerFragment:

DataKindBasePIckerFragment is mainly used to display and data related to the UI, such as SMS multiple choice contacts, in the multi-select interface is to see the phone number.

MultiBasePickerFragment is simply used to select multiple contacts, does not display any including phone number, email content information. So from its inheritance relationship can be seen, which interface to show detail.

6.2.6 Runtime Permission



RequestPermissionActivityBase :

This is a base abstract Activity, it hold PreviousActiviyt Intent which be used for re-Create origin activity. the base class provide base function,like hasPermission, startPermissionActivity,requestPermission,

RequestPermissionActivity:

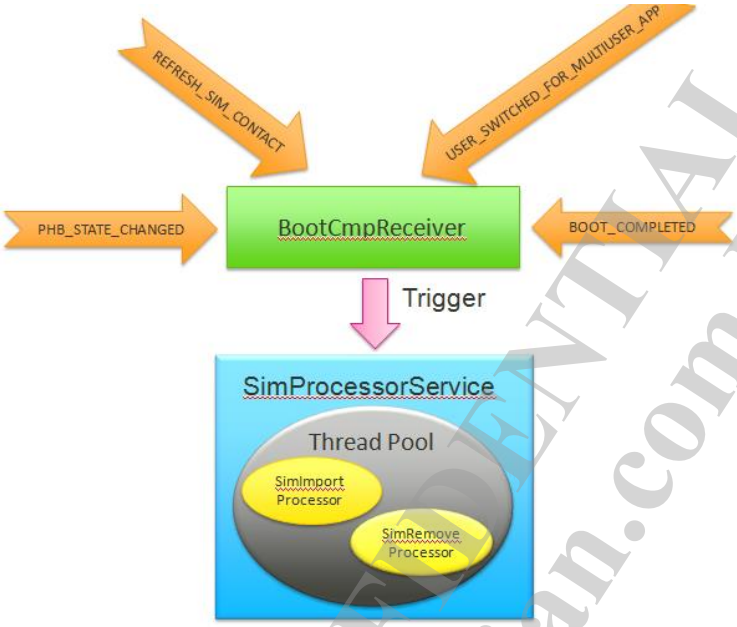
This class override getRequiredPermissions, getDesiredPermission and receive request result .

RequestImportVcardPermissionActivity:

This class should be used for import/Export function. because of request permission is diff with other activity, so it should separate part.

6.2.7 SimProcessor

It mainly has a receiver that can receive broadcast, then trigger SimProcessorService to do load or remove contacts. SimProcessorService maintain a ThreadPool, used to control related Processor(SimImportProcessor and SimRemoveProcessor) to do job. the below picture can show you the frame.



Related Broadcast

PHB_STATE_CHANGED: when Receiver received this broadcast, SimProcessor will be aroused to load sim contacts

BOOT_COMPLETED: sim contacts should be loaded when devices boot completed

USER_SWITCHED_FOR_MULTIUSER_APP: when user change multi-user, this broadcast will be send out, only owner can use sim contacts.

REFRESH_SIM_CONTACT: this broadcast is send by only contacts App , not system broadcast, it is used to do something permission related.

7 Sequence

7.1 People Main

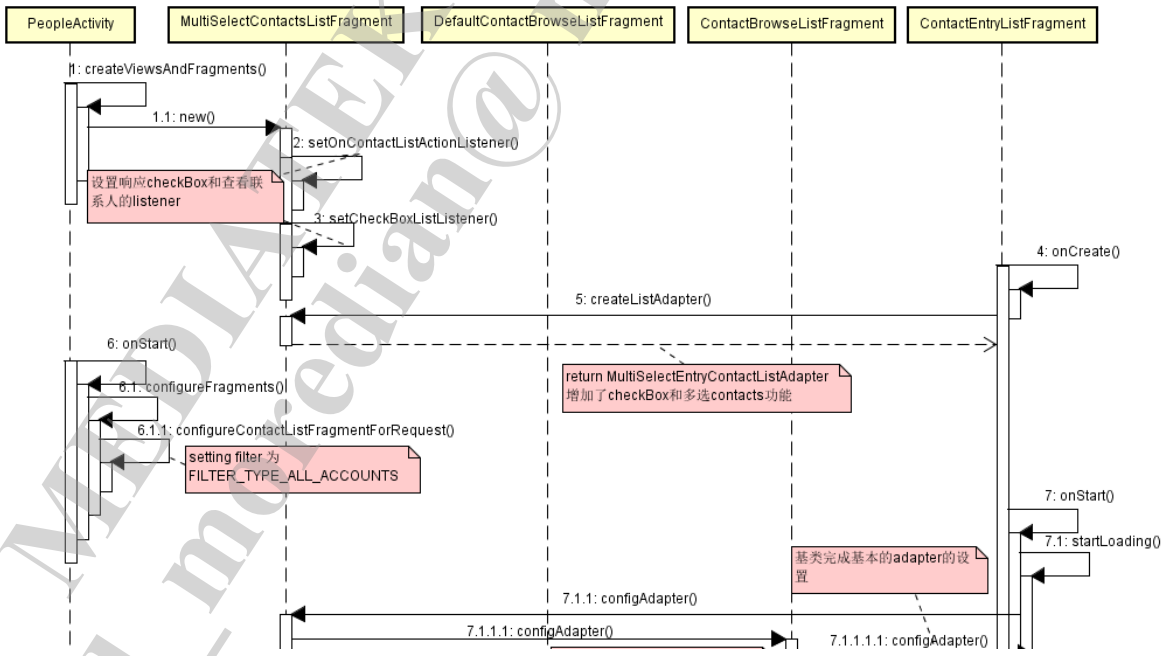
7.1.1 UI display flow

In onCreate method of PeopleActivity， it will new MultSelectContactListFragment object， with seting releated listener for checkbox and view contacts, from life cycle of fragment to see, it will build adater which for overall time for fragment in onCreate method of ContactEntryListFragment object. Abstract method createlistadadapter method will be called and make MultiSelectEntryContactListAdapter in this case ,the main role of this adapter is that copy data to listview which showing in the main page.for this common Ui, it will use filter to diff showing Ui by diff Uri for loader to load contacts.

Look up

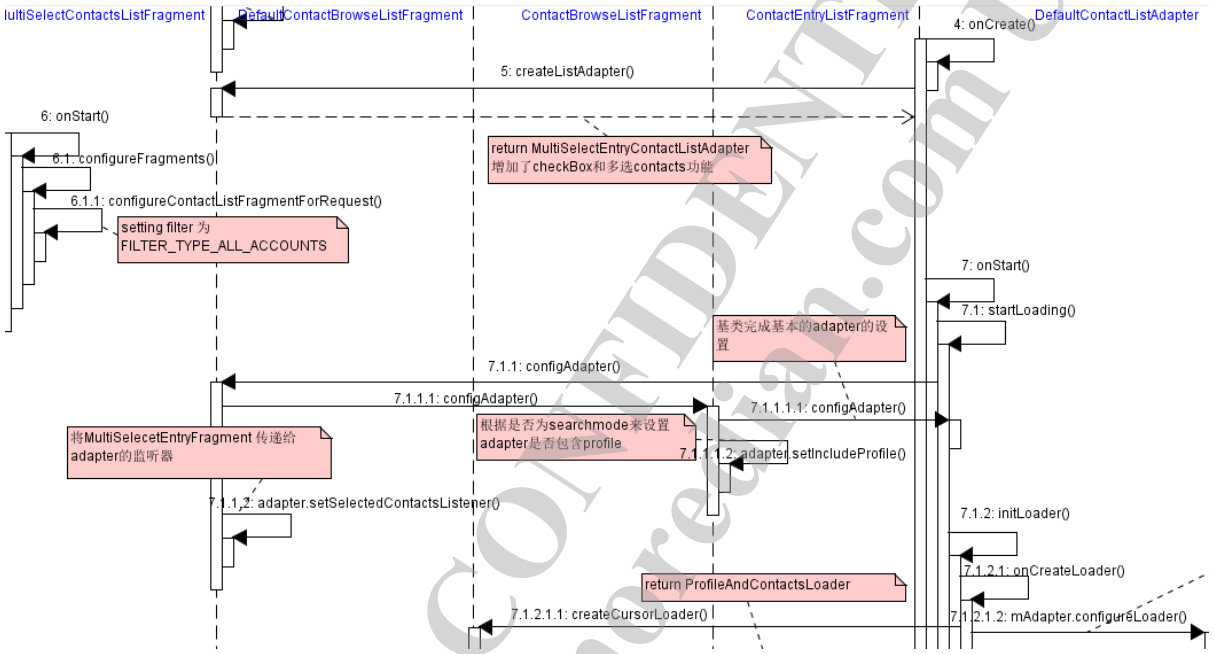
/packages/apps/ContactsCommon/src/com/android/contacts/common/list/DefaultContactListAdapter.java

In configureSelection method.different filter will build out different query selection which will be used for loader. ContactEntryListFragment will load data and configture adapter in onStart method. The flow chat as below.



7.1.2 Configure Adapter

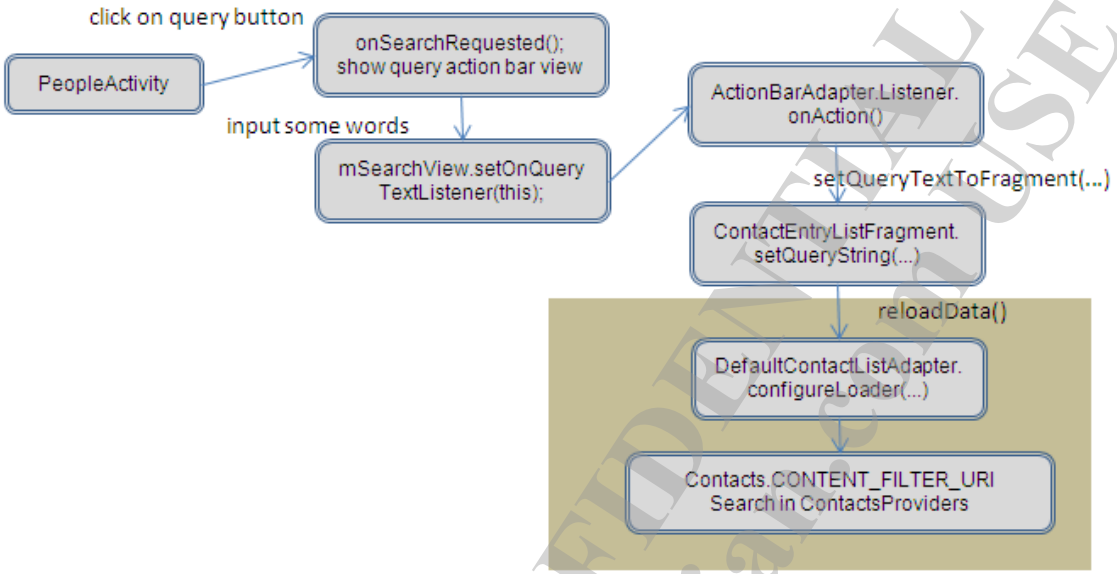
It can add more setting when override configAdapter method in related class. For Example , ContactBrowserListFragment will consider whether is or not show profile by serarchMode flag.and it will transfer MultiSelectEntryFragment instance to adapter for later callback. It will show the selected contacts and checkbox for user. The flow chat as below.



7.1.3 Search contact flow

User click “serach button” -> onSearchRequested() -> mActionBarAdapter.setSearchMode()

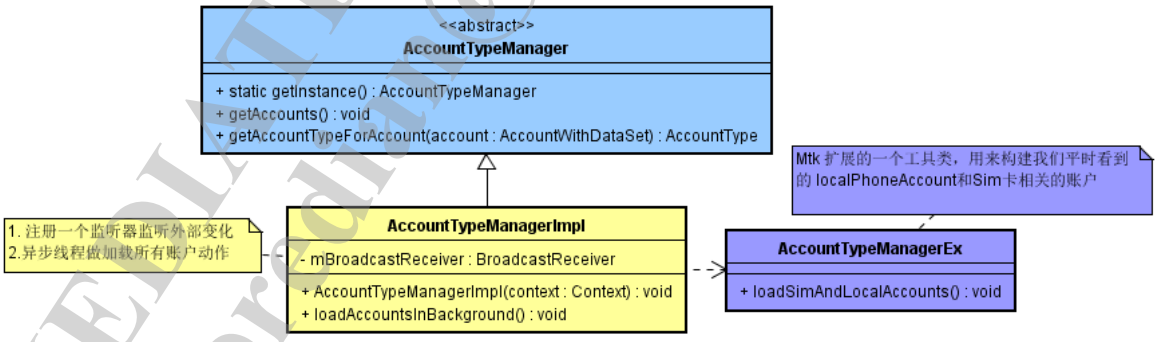
User input search content -> onAction() -> mActionBarAdapter.getQueryString() ->setQueryTextToFragment() -> mAllFragment.setQueryString()



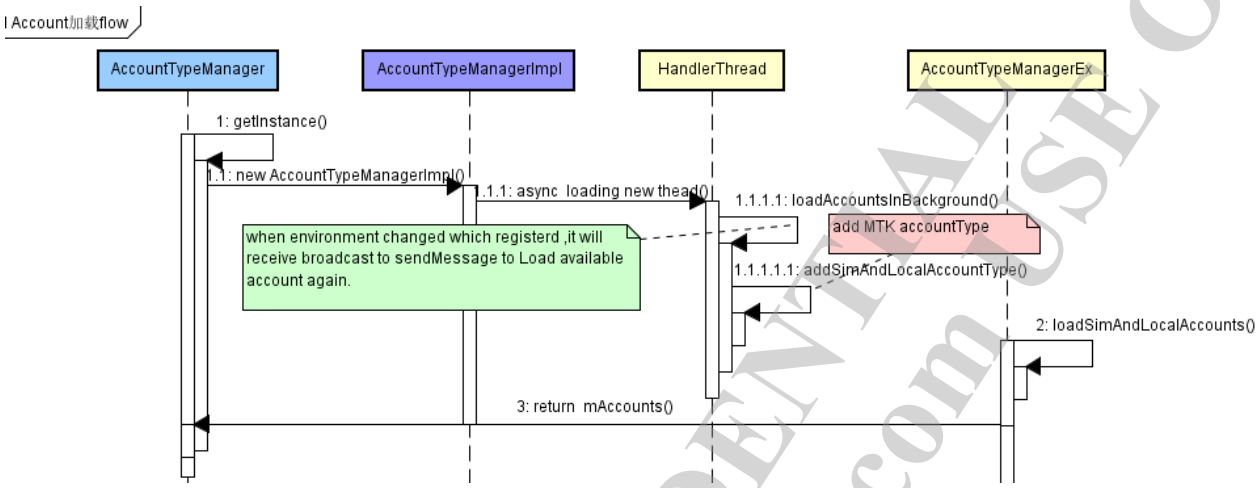
7.2 Editor&AAS/SNE

7.2.1 load account flow

AccountTypeManager related inhert class as below. AccountTypeManager is abstract . the implement is AccountTypeManagerImpl ,included broadcast receiver that used trigger to load account. Mtk account (local phone account,sim account) will load in this . MTk extend AccountTypeManagerEx to load sim and local account.



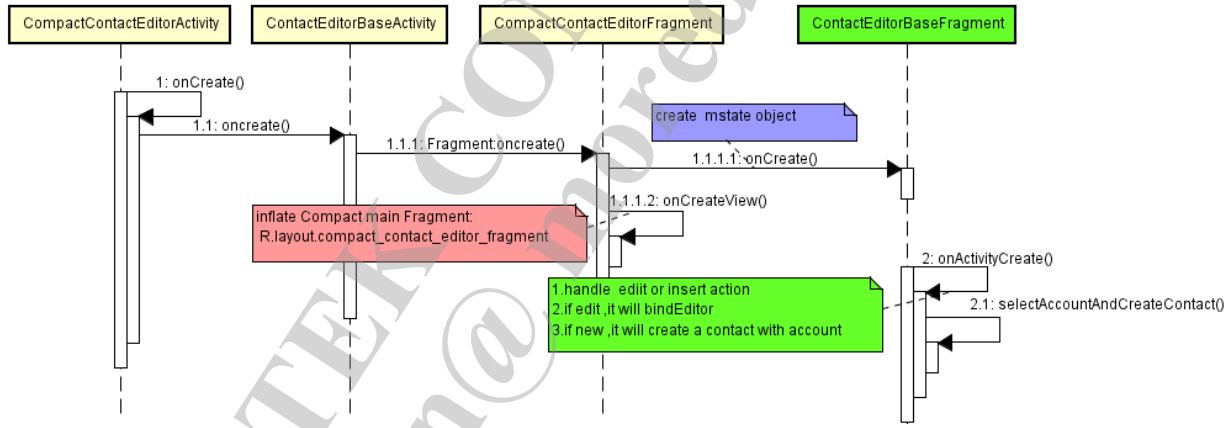
Load account flow as below.



7.2.2 contruct UI flow

Activity is main show for UI in android , fragment also include in Activitity UI to make easy change for different style. The change flow as below for CompactContactEditorFragment create view.

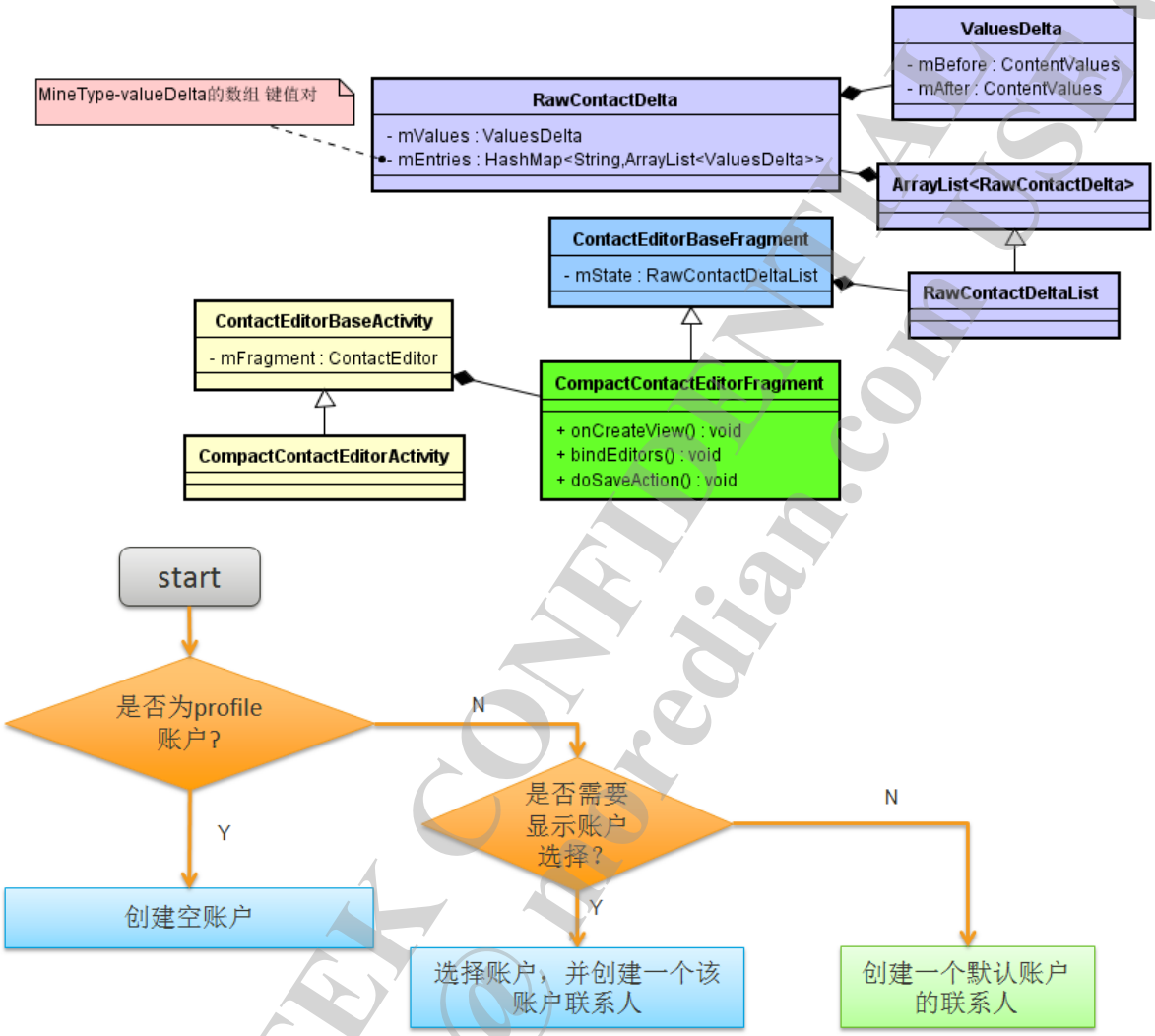
,



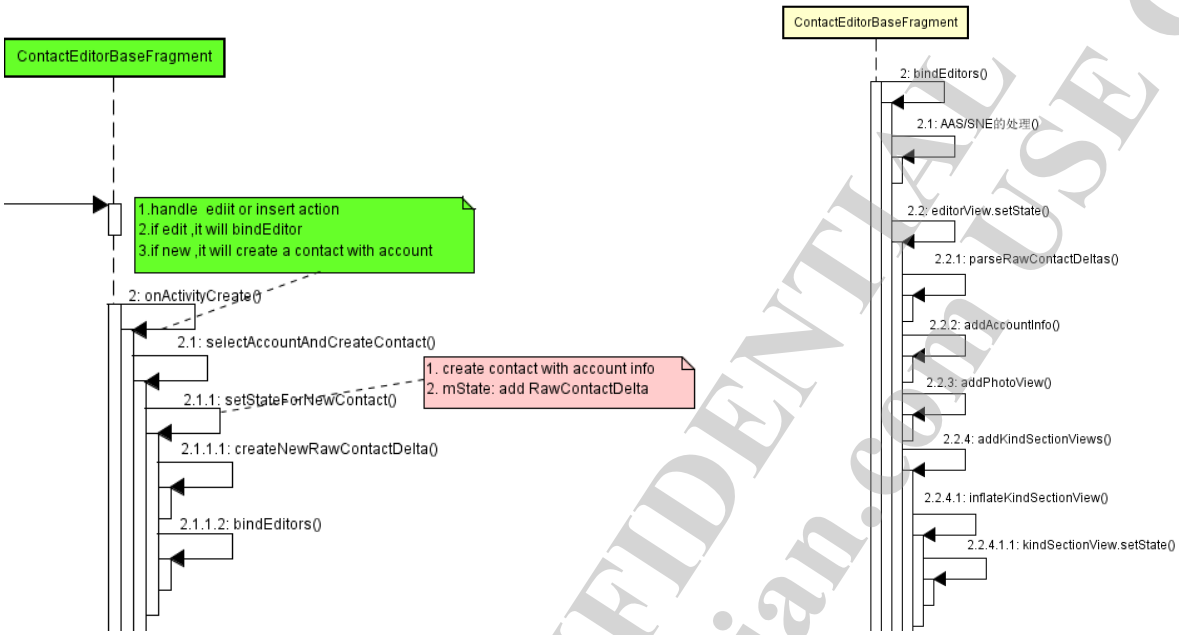
7.2.3 mState data filled in

UI view is mapped with a related with a DataStructure. When user input some information to mark a contacts, the related data will load in related data item. So Ui (fragment) should maintain a this data which is mState member in ContactEditorBaseFragment.

mState is a RawContactDelta rarrry which contains some key-vaule. ValuesDelta is made of two ContentValues(mBefore,mAfter) used to reset or update for one contacts information.the related flow as below.



When user selected a account to edit or new a contacts,it will build out RawContactDelta this data. This data is stand for the contacts information. The data of RawContactDelta must depend on the support type which AccountType apply. For example, Usim account not support origration this field. So account type cannot have this field. The way get this field is through MineType to get,if not have ,cannot get it. So RawContactDelta will not create this date filed. The flow as below chat.



7.2.4 bindEditor

It will map into Ui View after mState data had been created. It has four steps.

Parse mSate: call parseRawContactDeltas method, make data (mKindSectionDataMap) for CompactRawContactsEditorView.

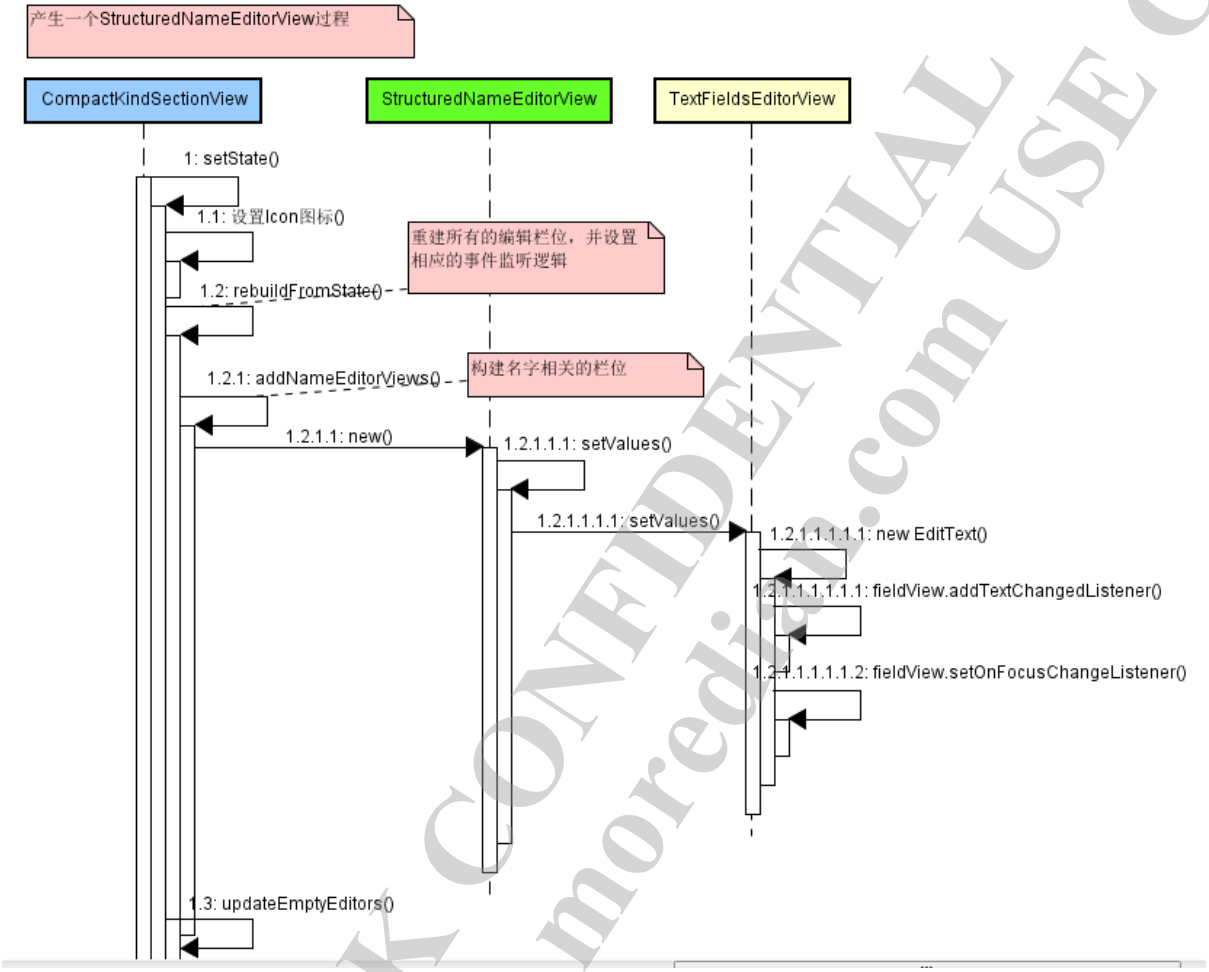
Add account information: if current device have more one account, it will show a list to select for end user.

Add photoView: this step will handle photo view .

Add all field information: this step is core. all information for one contacts will filled in this view.such as name,email.

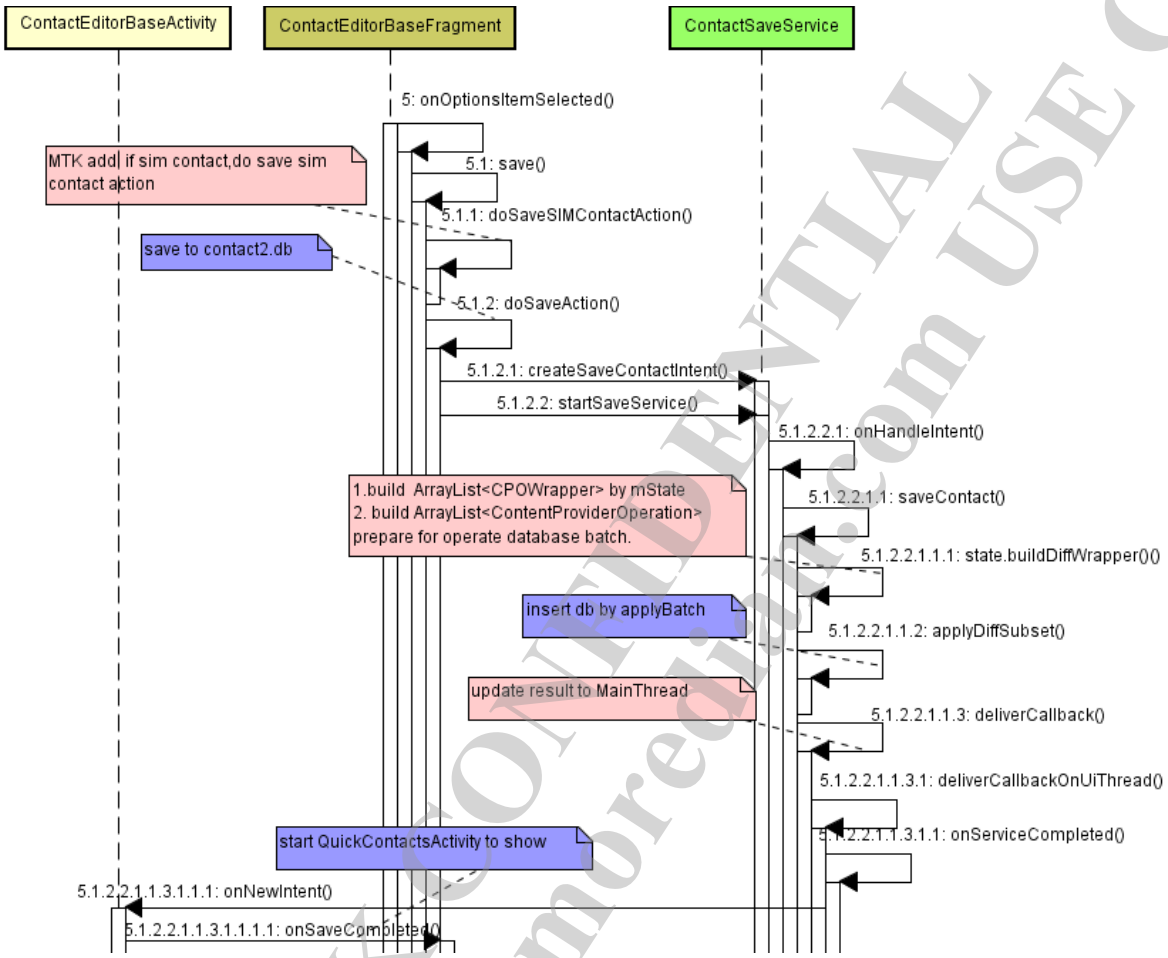
The flow as below .

build StructureName filed flow as below.



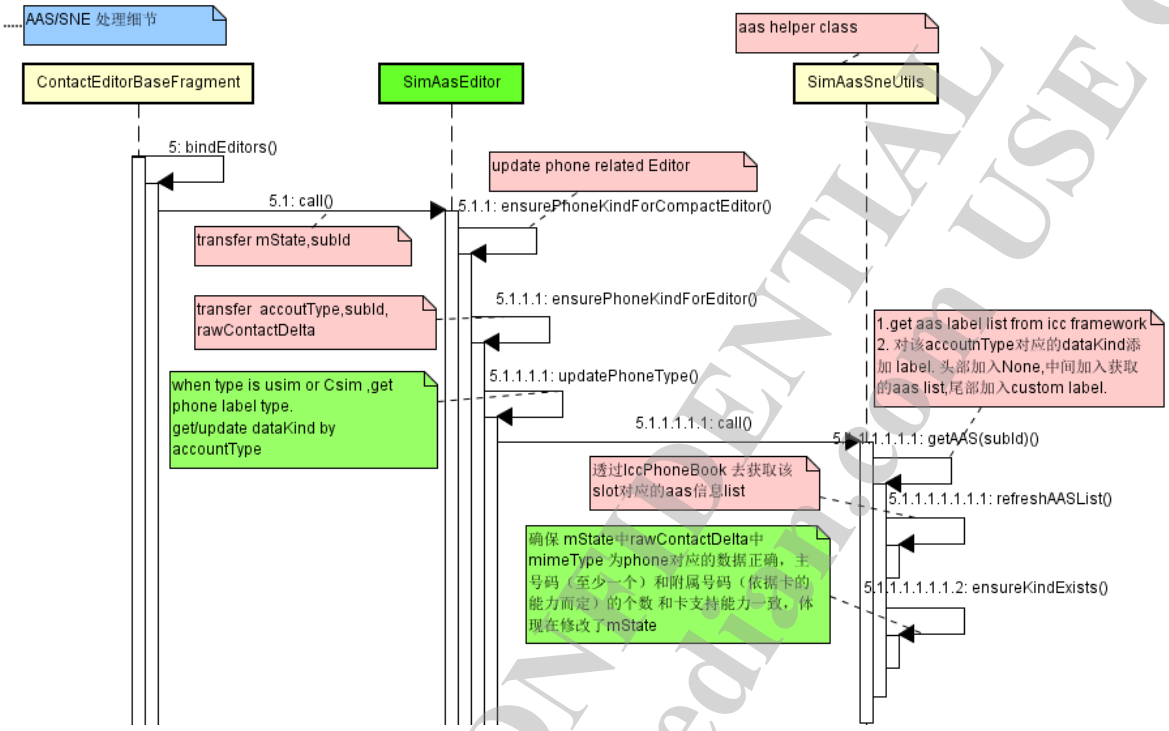
7.2.5 save contact flow

When user finished input information ,click save button to save. The flowo chat as below. mState will be transfered to ContactSaveService (intentService) to handle. At last , mState will changed to ContactProviderOperation to applyBatch to contacts2.db. after that it will show quickContactsActivity for this contacts detail.



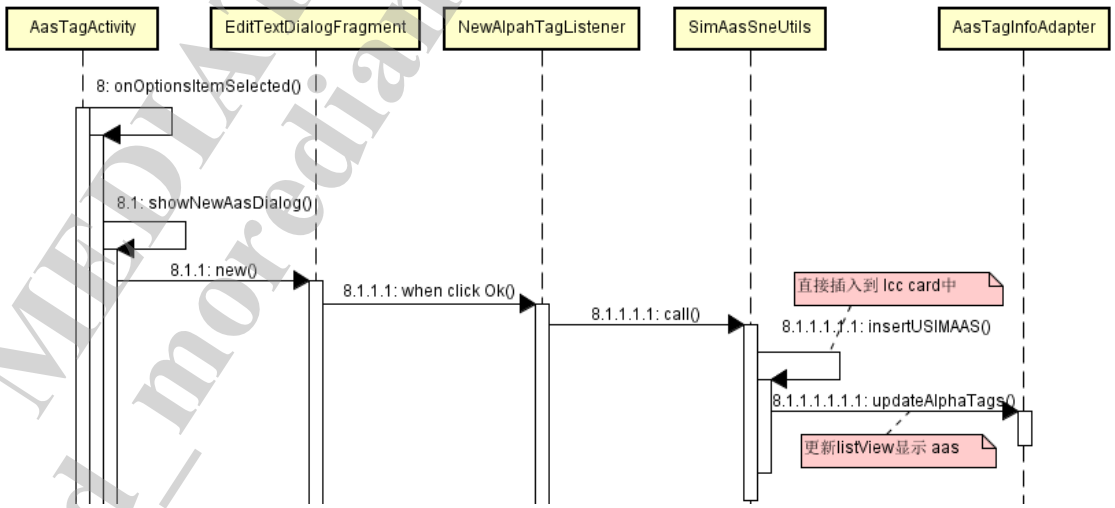
7.2.6 Contruct AAS UI

The fllowing flow chat is get aas/sne flow .



Mtk feature about addition number/aas also run in bindEditor method. The value will get in that flow. mState data content is depend on the actually capacity of sim card. For example for Usim/csim card type, first, it will get all aas label list which stands in sim card. Then copy that into typelist filed of dataKind. At the same time, it will also add two label (None,custom) .after that , the phone type minetype update done. Following step is addition number this filed whether is or not need by support the capacity of sim card.

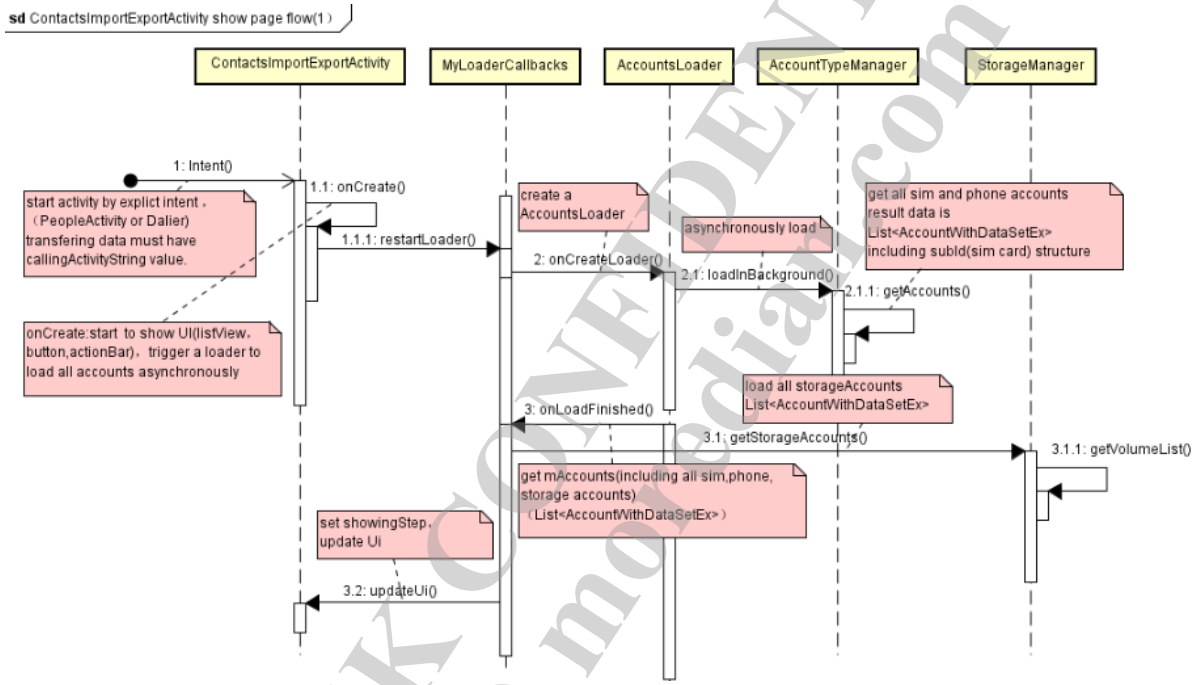
When user click customization button to customizate aas label ,it will jump into a Activity to edit aas. This Ui can make add/delete/update aas label. The flow as below.



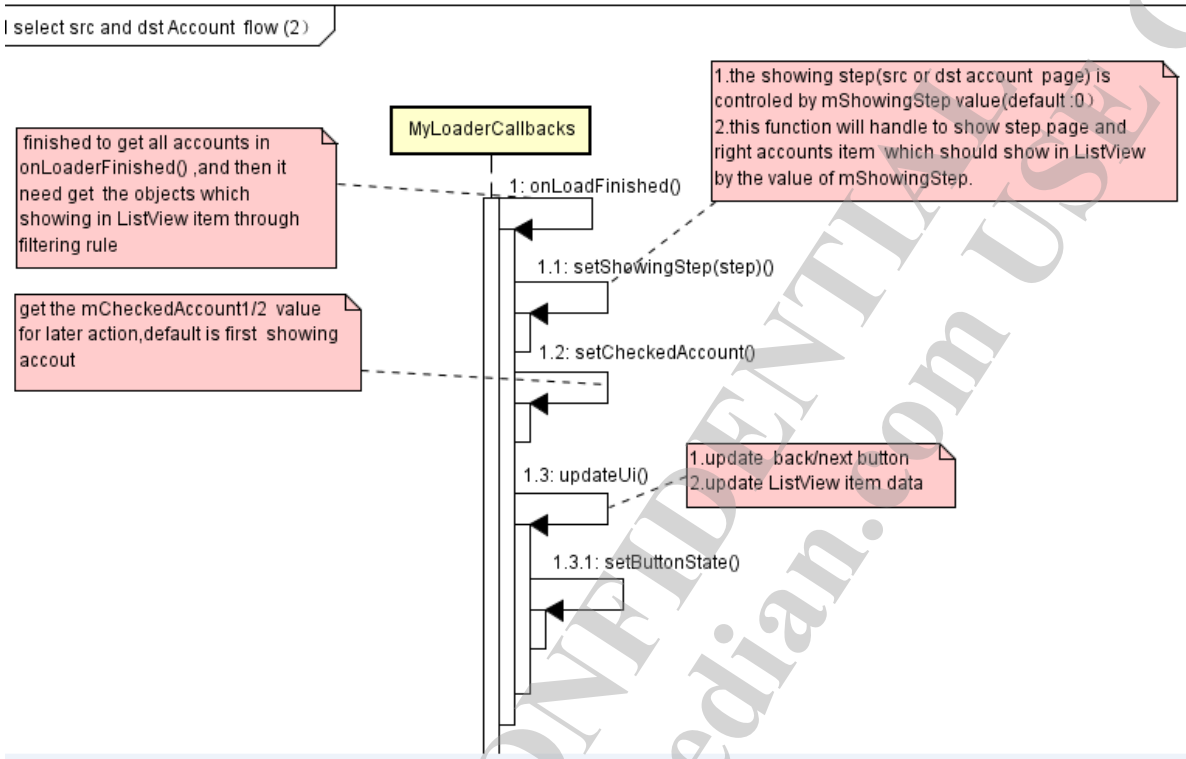
Pay your attenations, the aas label which you add will save into icc card. Not the cache. and the update UI(listView). Other operation like delete aas label also delete label which stands in icc card. In a word, operate aas label will refect into icc card.

7.3 Import/Export

7.3.1 UI flow



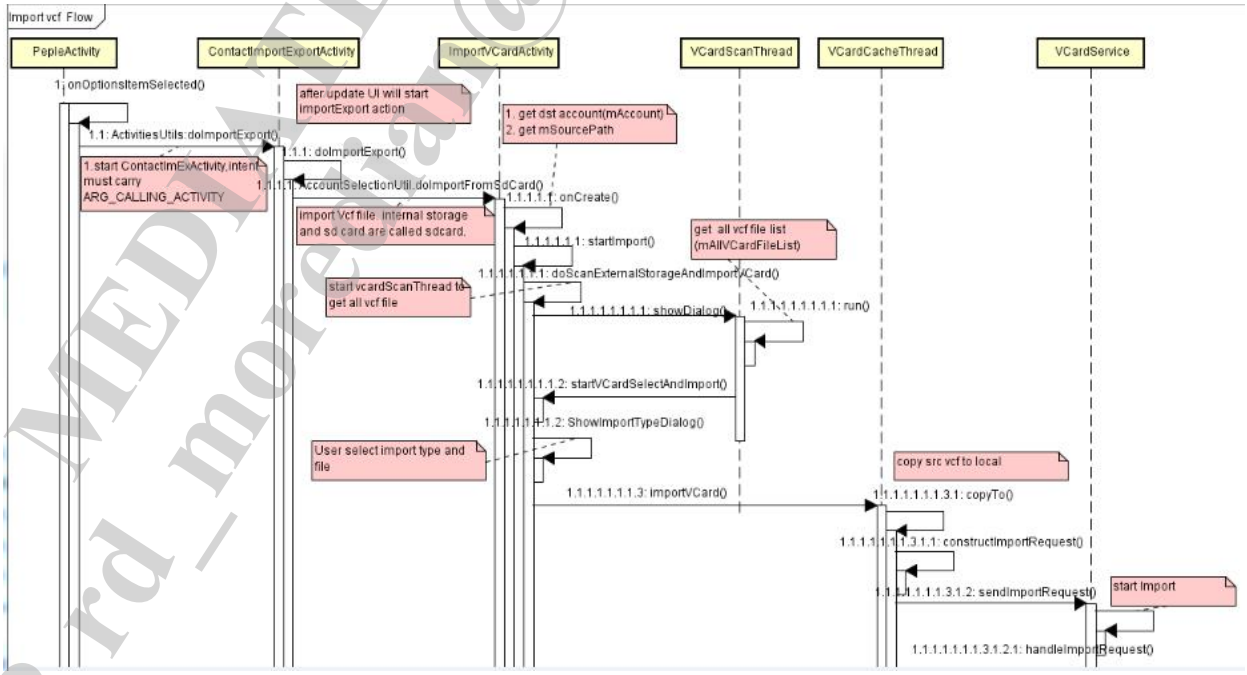
ContactsImportExportActivity is the enter UI for this feature. onCreate method will trigger a AccountLoader to load all sim ,phone account. And then collect all storage account like sd card,internal storage. The flow chat as below.



It will update source account and destination account by setshowingStep method. Related adapter is AccountListAdapter offer data for listView.

7.3.2 Import storage account into phone account flow

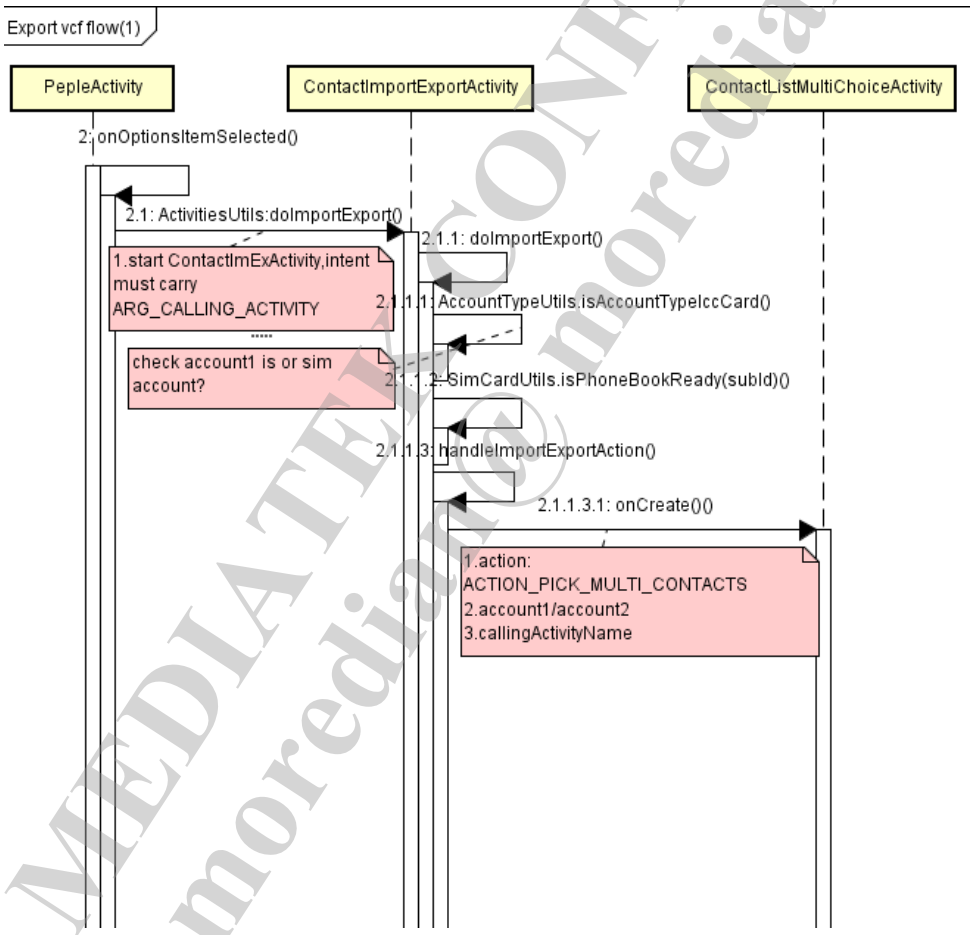
Import is from the internal storage or sd card or email attachment vcf file into the phone, the following describes from the storage (internal and sd card) import vcf file process.



the trigger of import and export , if from the Contacts, it is initiated from the PeopleActivity menu, through the menu response processing onOptionsItemSelected method, start ActivitiesUtils: doImportExport to do import and export behavior. After the selection is complete, the source account for the internal storage or sd card, the destination account can only choose phone contacts. The actual import and export will be from the AccountSelectionUtil.doImportFromSdCard method, to start ImportVCardActivity, the transfer of parameters for the source account path (for example, internal Storage, / storage / emulated / 0, sd card, may / storage / 0403-0201, digital number on behalf of sd card, this information can be in the eclipse DDMS-> File Expplorer view) and destination account information.

7.3.3 sim copy to phone flow

First to introduce sim card contact to copy the entire flow to the phone. Sim card contact copy to the phone, the whole process and import the same period, select the sim card account for the source account, phone account for the purpose of the account, all the way click "next", will call its Multi interface to select the current need to copy the sim Contact, the process flow as shown below.

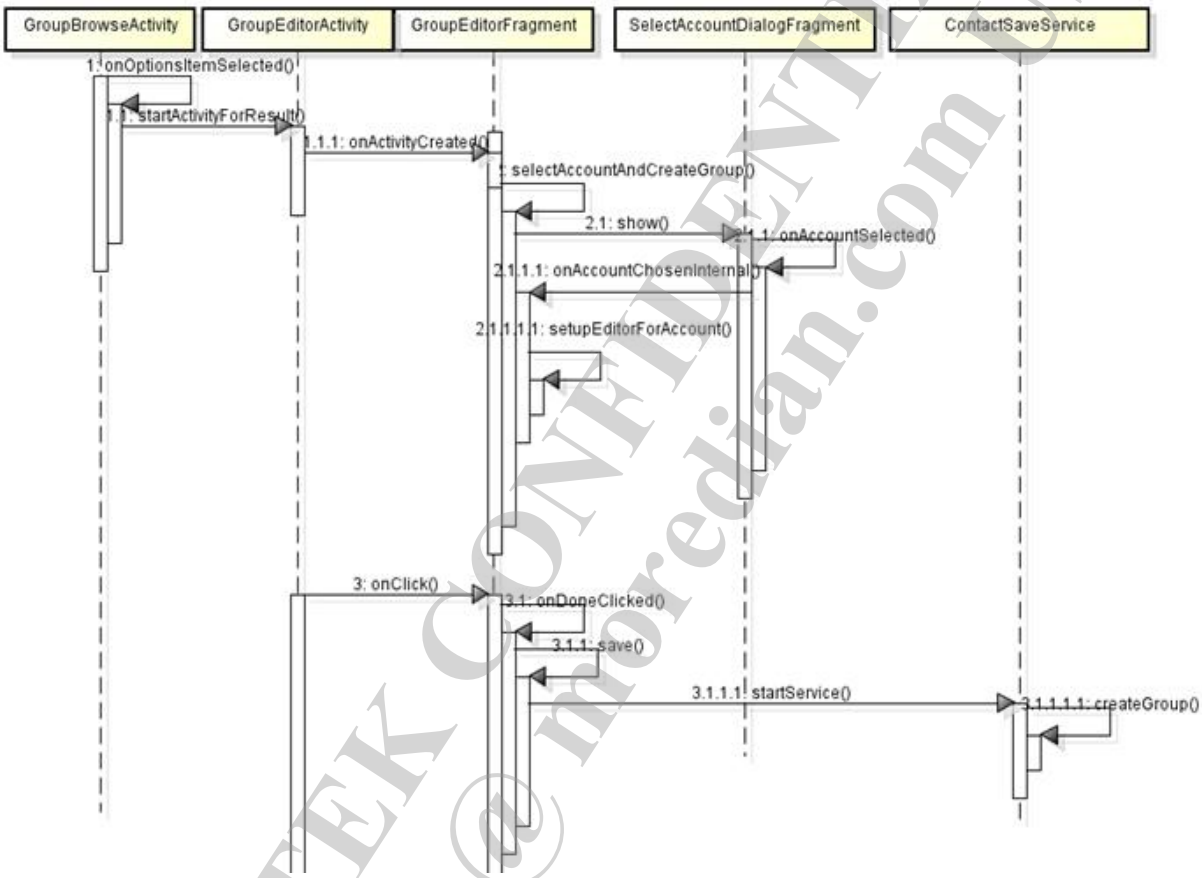


ContactImportExportActivity Initiate the copy action, starting ContactListMultiChoiceActivity, passing the action, source account and destination account, and callingActivityname to contactsListMultiChoiceActivity which fragement it should show,., According to the passed Action, ContactListMultiChoiceActivity will enable MultiDuplicationPickerFragment as the main interface.

7.4 Group

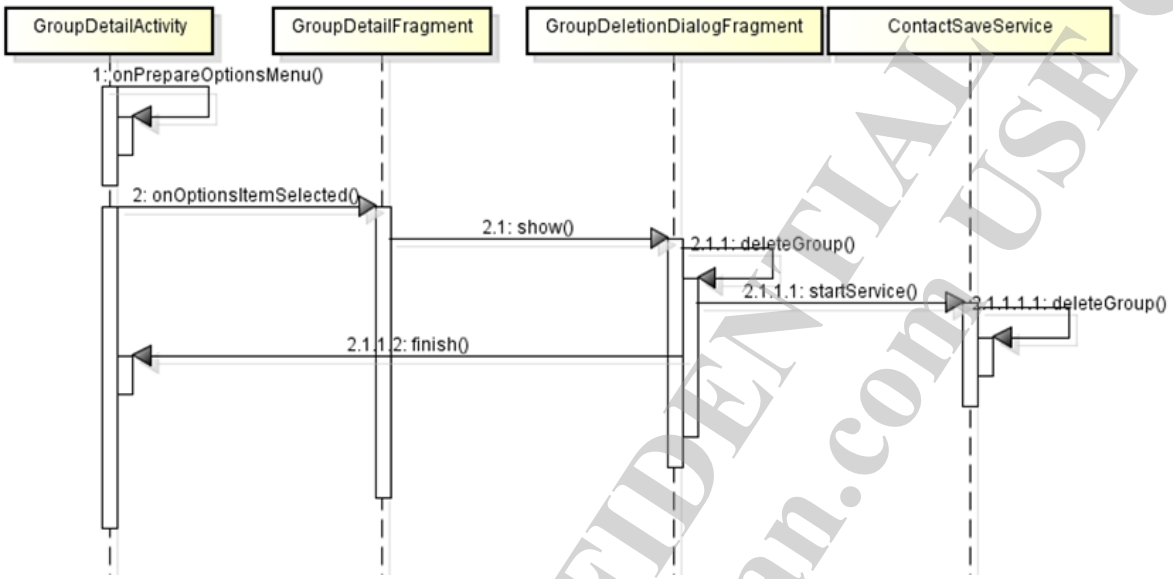
7.4.1 New A Group

New a group flow as below.



7.4.2 Delete A Group

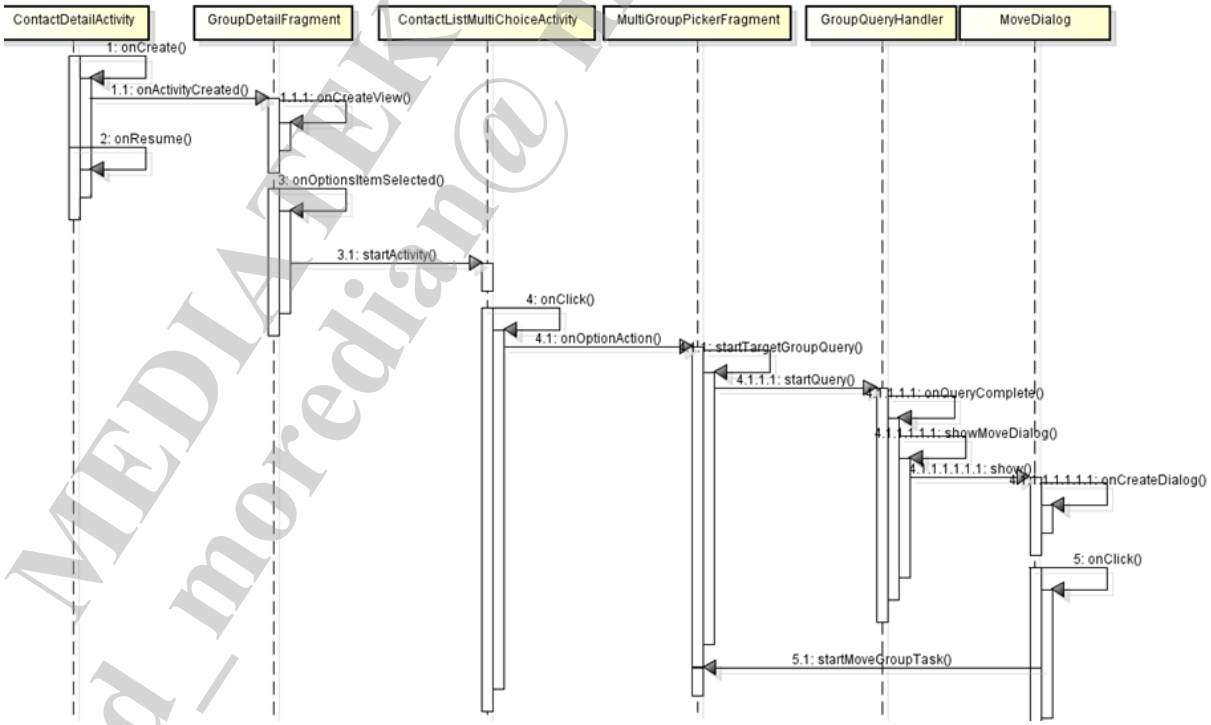
Delete a group flow as below.



7.4.3 Move Group Members

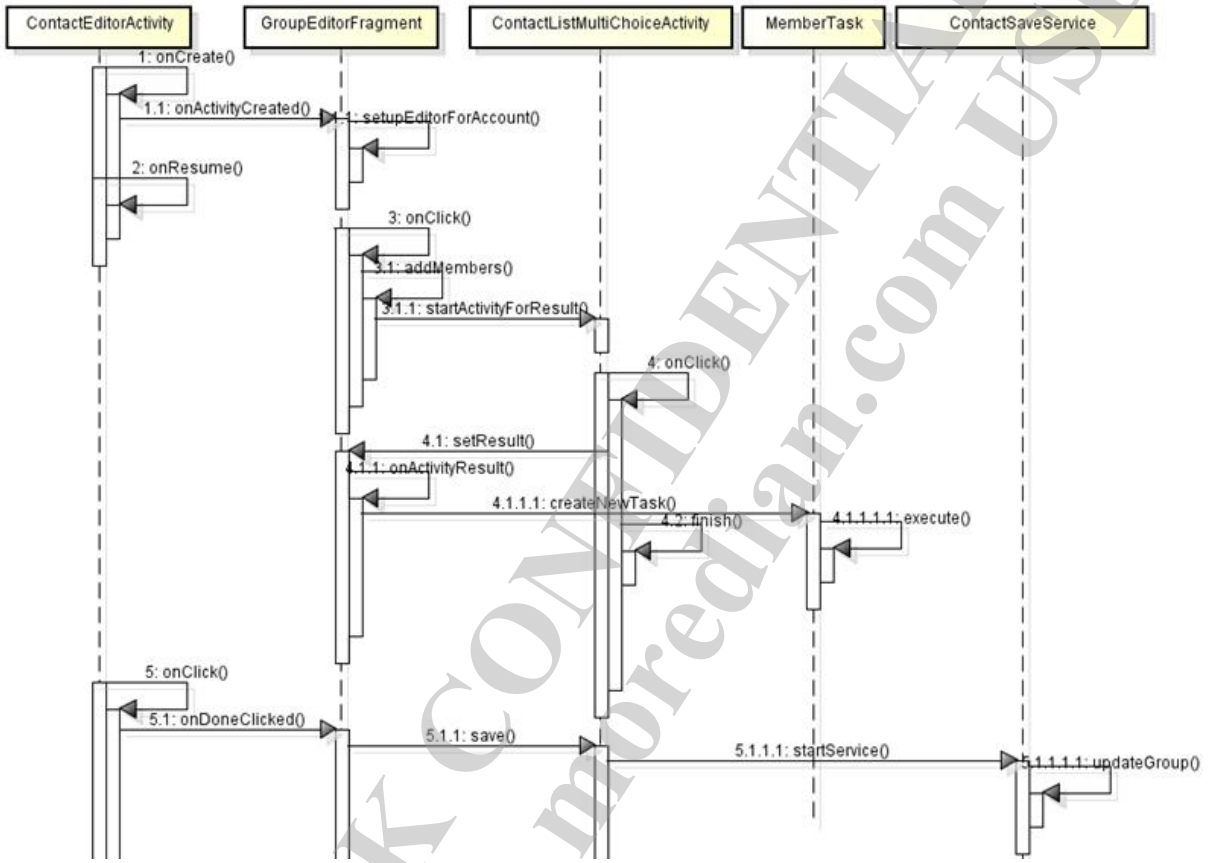
First get the id of the contacts which need to move from the target group. And then confirm which contacts in the source group need to be moved already exist in the target group. Finally, in the source group to remove the need to remove the contacts, in the target group to add the corresponding contacts.

The flow as below.

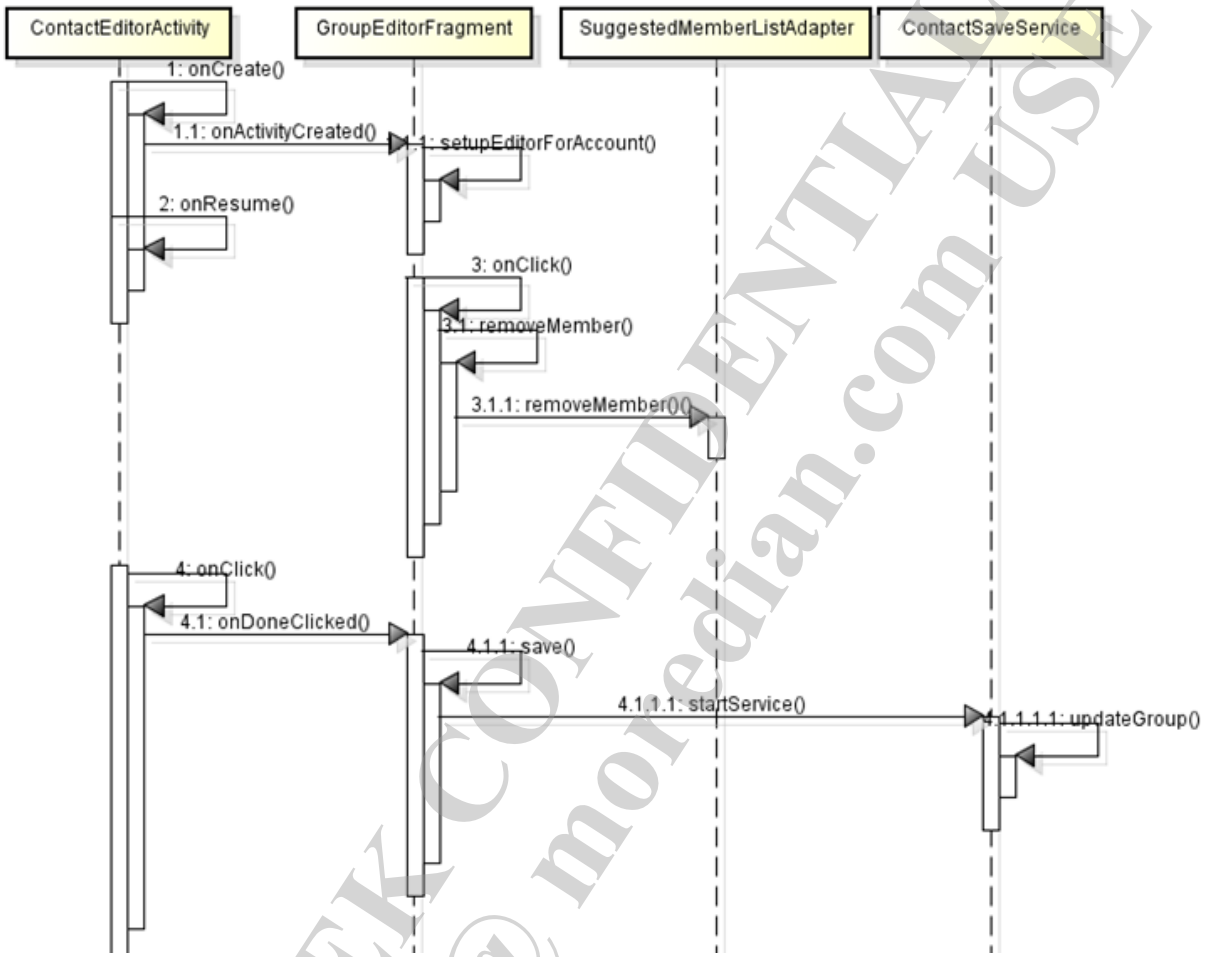


7.4.4 Add Contacts to Group

The related flow as below.



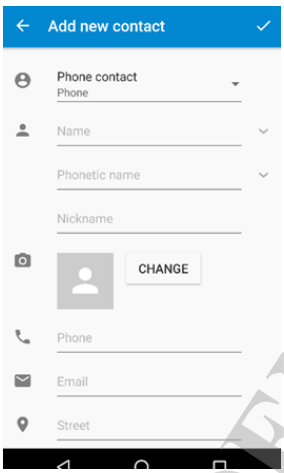
7.4.5 Remove Contacts from Group



7.5 QuickContact

7.5.1 Build QuickContact principle

In fact , the construction of QuickContact and Contacts editing is similar, you can look at the next edit, the same is the first column is an icon, such as name, number, mailbox, address, etc., the second column is specific to fill the information The



For the construction of these information, the part of the Contacts Editor UI section have some low-leavel explain. The way is the same that incluing an icon, a blank. Since the type of information stored by a contact is limited, we can prepare various types of features in advance, and then have a unified styling factory to handle. Specifically: name, number, mailbox, etc. We can use different Data Kind to represent, and later to enter the blank is a unified building. So that only need to know Data Kind, you can use the same method to build a different item.

7.5.2 QuickContact event response

For the event response of each Item , the core content is to add the corresponding icon above the intent, and then add the Icon onClick event response. So when the user click on the time, you can respond to the incident through the act response and jump to the target Activity. Specifically: if the data item is an Email type, give it a corresponding Primary content description, the intent of the event, the header to be displayed, and finally the icon.

```
else if (dataItem instanceof EmailDataItem) {
    final EmailDataItem email = (EmailDataItem) dataItem;
    final String address = email.getData();
    if (!TextUtils.isEmpty(address)) {
        primaryContentDescription.append(res.getString(R.string.email_other)).append(" ");
        final Uri mailUri = Uri.fromParts(ContactsUtils.SCHEME_MAILTO, address, null);
        intent = new Intent(Intent.ACTION_SENDTO, mailUri);
        header = email.getAddress();
        entryContextMenuInfo = new EntryContextMenuInfo(header,
                res.getString(R.string.emailLabelsGroup), dataItem.getMimeType(),
                dataItem.getId(), dataItem.isSuperPrimary());
        if (email.hasKindTypeColumn(kind)) {
            text = Email.getTypeLabel(res, email.getKindTypeColumn(kind),
                    email.getLabel()).toString();
            primaryContentDescription.append(text).append(" ");
        }
        primaryContentDescription.append(header);
        icon = res.getDrawable(R.drawable.ic_email_24dp);
        iconResourceId = R.drawable.ic_email_24dp;
    }
```

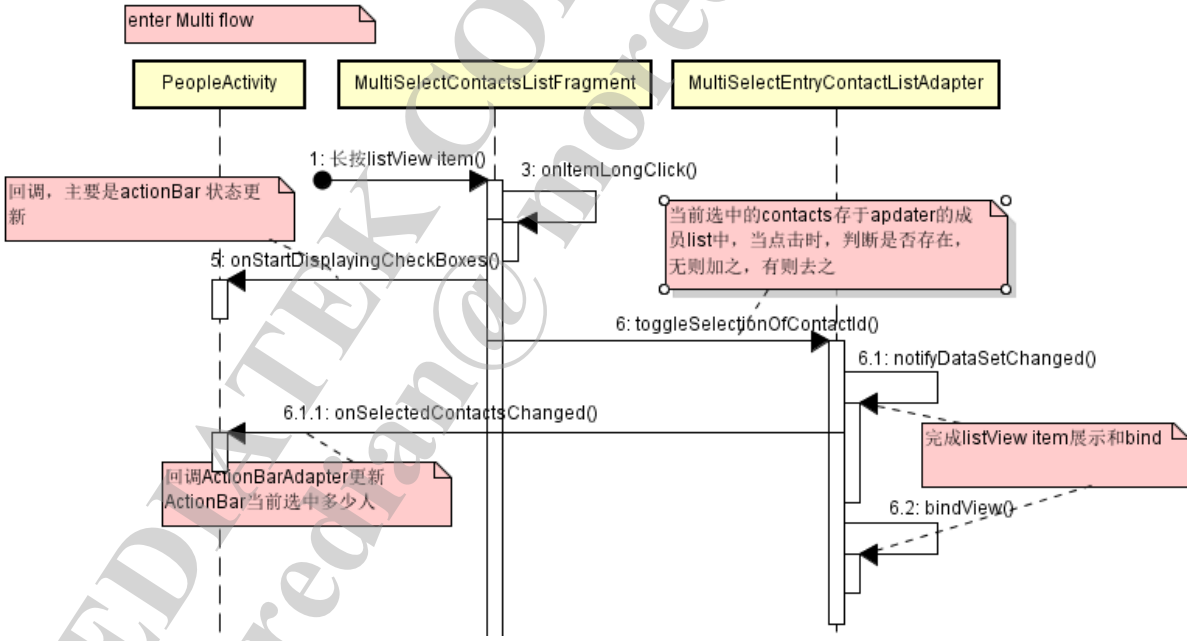
The most important method is dataItemToEntry () in QuickContactsActivity.java the main role is that obtain the data through different dataItem and build the UI which show .

7.6 Multi Operation

7.6.1 Google multiSelection

7.6.1.1 Enter to multi flow by long click

In the PeopleMain interface, click on a contact item, it will enter the multi-select interface, if not enter the multi-select mode, click on a contact item, it will jump to the contact's detail interface (quickContacts interface), so for this two different cases, judge that is based on whether the current state (can represent as checkBox is displayed or not). The controls associated with the multi checkbox are done in the MultiSelectEntry. The response to onItemClick is done by the onClickItem of the base class ContactEntryListFragment, creating the listView in the ContactEntryListFragment: onCreateView () and setting the listView to respond to Item longclick, itemClick. The listView item has a long press or click on the time, will call ContactEntryListFragment longClick or onItemClick method, from these two methods to do the distribution, the subclass only need to achieve onItemClick and onItemLongClick method . The long click on PeopleMain flow is shown below:

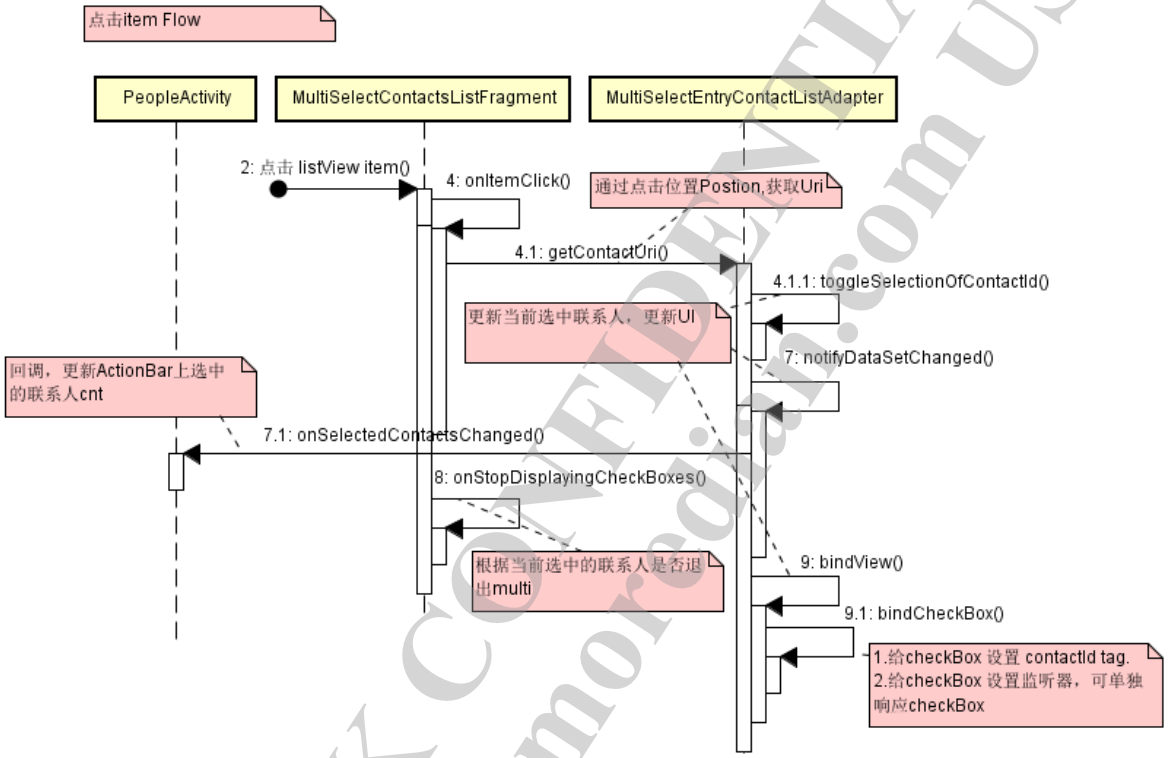


BindView method will call bindCheckBox, this method based on mDisplayCheckBoxes member variables to determine whether the current checkBox display, no enable, then all hide. The above process is mainly to update the listView interface refresh, select the contact array adapter, update the current selection number in actionBar.

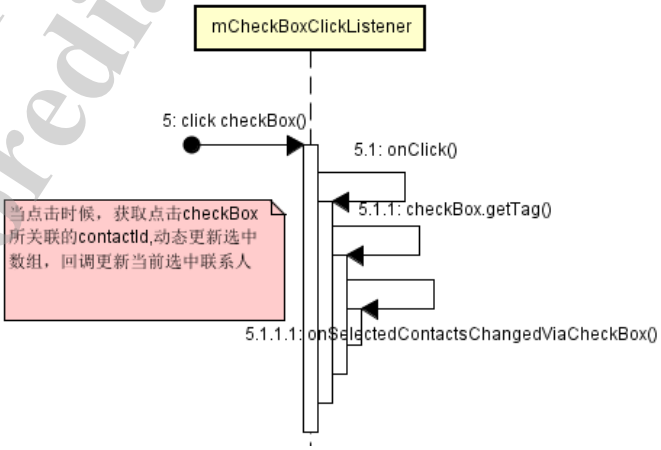
7.6.1.2 点击 listView item 和 checkBox 处理流程

click on the item In the ListView, according to the previous description, will eventually call back to MultiSelecetContactsListFragment onItemClick method, this method will click on the location of the Postion, to

find out the location of the corresponding contact Uri, to get to this contact contactsId, In the current situation (whether the multi-select state, the contact is sdn contact), meet the conditions will trigger the adapter to select the contact array and UI to refresh, if the currently selected contact is empty, then that at this time no Select the contact, exit the multi-select state, the process shown below.

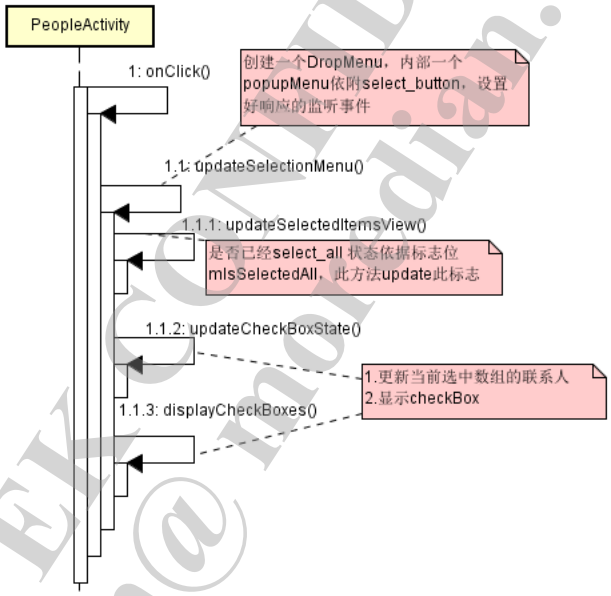


When checkbox is shown in multi Ui, checkBox is also a single click, why the corresponding response for the item and checkBox are separated? This is because google multi UI either show the ordinary contact list, do not show checkBox, or can handle long press action to enter the multi-election, show checkBox, so these two parts of the logic of unified processing, so CheckBox can click, update The In the display process bindView will check all the check box is updated.



7.6.1.3 mtk "select all" menu handle flow

In order to increase the convenience of the process of multi-selection, mtk provides a "select all" menu, you can select all by click this button for end user, in the PeopleActivity inCreate method, extract the button and set the listener, click on the button, first Will create a DropMenu, used to deal with "select all" and "deselect all", the flow of processing, the first adapter will be selected within the current size of the array and the current listView count count, if equal to the current state is selected, Otherwise it is not, and then set the mIsSelecetAll flag, and then according to this flag to select the processing branch, click on "selecet all", will call updateCheckboxState method, the method will be done within the update adapter selected array, the current all add, And then call the displayCheckBox method, will refresh the UI, and according to the current selected data, check the checkBox. This completes the full selection process. "Deselect all" process is similar, just the opposite.

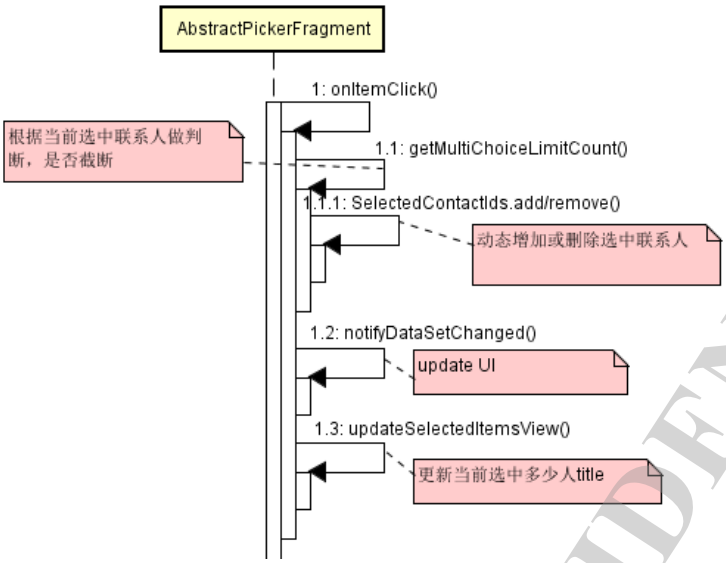


7.6.2 Mtk multiSelection

7.6.2.1 select all or disslect all

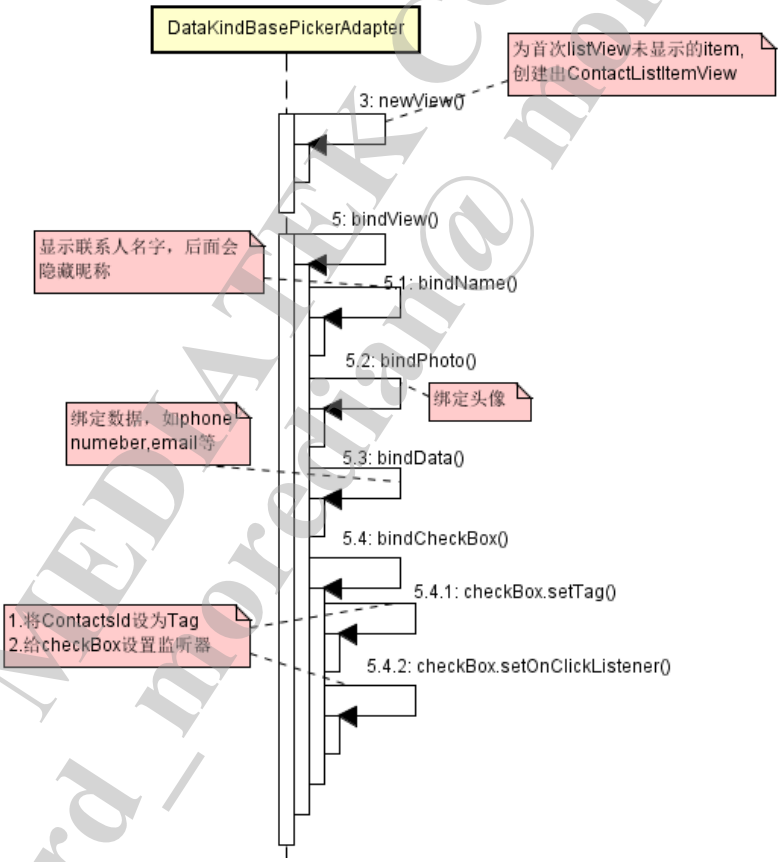
After the interface appears, you can choose to click the item or checkBox to select the contact, you can also click on the "select all" button to select and clear all contacts, select the contact storage location and google multi are consistent, will be selected contacts Id Into the corresponding adapter list, select the number of people are selected by adapter in the selected list.

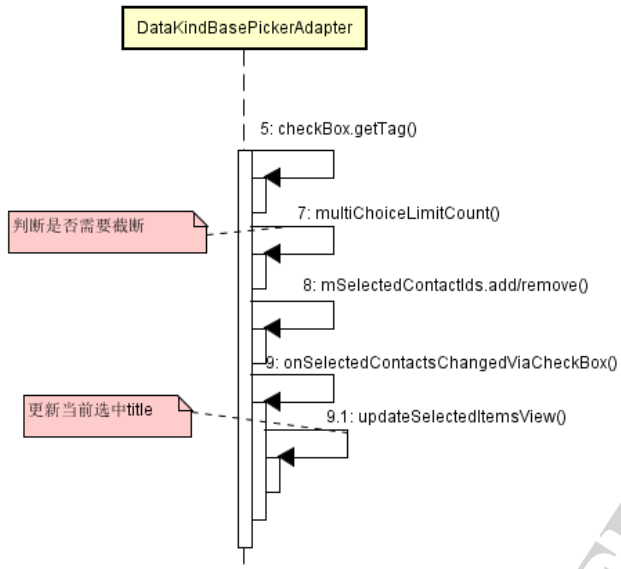
In the mtk multi, the click item is through the AbstractPickerFragment: onItemClick method response, in this method , dynamically modify the corresponding Adapter in the selected list, in addition, the method also has a limited selection of contacts (not more than 3,500 people)



7.6.2.2 multi select by click checkbox

Mtk multi DataKindBasePickerAdapter in the newView to create View, bindView is responsible for binding the data in the View, bindView process will call bindCheckBox to set the CheckBox listener, so when View is finished, give the CheckBox has been set to respond Of the listener, the setting process as shown below.



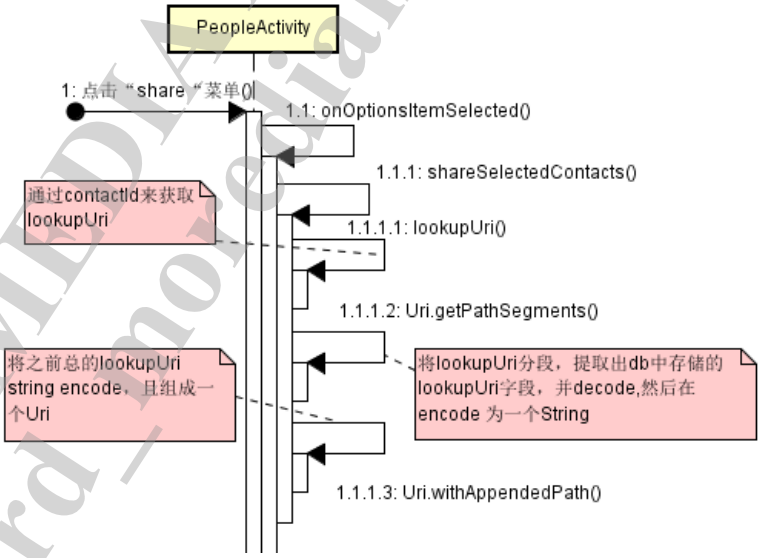


As the checkbox has been bound to the listener, so when click CheckBox, the direct response to the listener logic, as shown above. First through the getTag method, access to this CheckBox associated contactsId ,, and then determine whether to meet the restrictions required to dynamically add or delete selected list array, and finally update the selected number of information.

7.7 Share/VCard

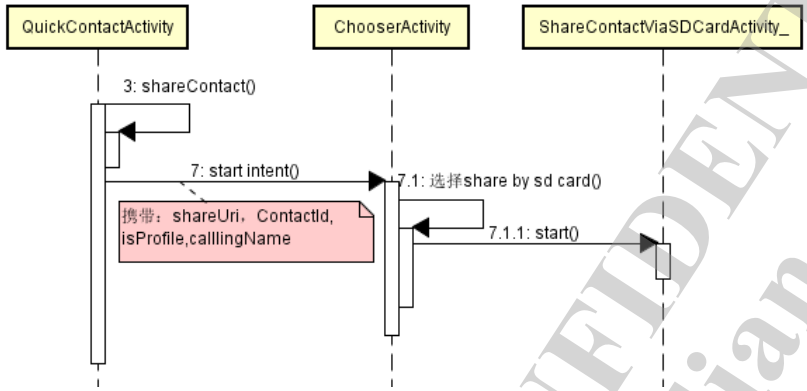
7.7.1 share multi contacts flow in PeopleMain

After clicking on the share menu, the process as shown below, first from the multiAdpater to get the currently selected ContactsIds, and then through ContactsId to find out contactUri, but also through contactUri to get the contact lookupUri, and finally all the LookupUri encode look as a string, and finally spliced into a total Uri, where the need to pay attention to the code after the encoding process.



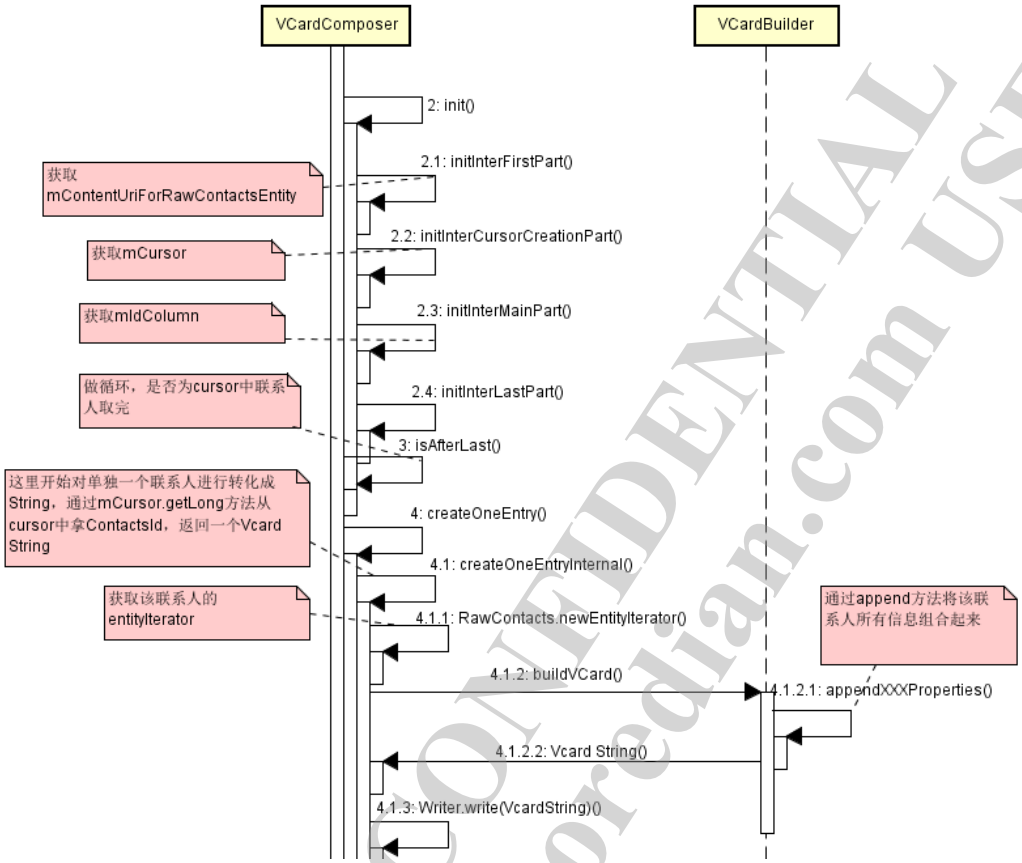
7.7.2 QuickContactsActivity share flow

In the peopleMain interface, click on a contact, as shown below, will be transferred to the contact details (QuickContactsActiivty), click the menu "'share" to share, when entering the detail interface, will get all the information of the contact, Click "share", will send an intent to go out to carry the contact's contactId, shareUri, if the profile contact, will carry a flag description.



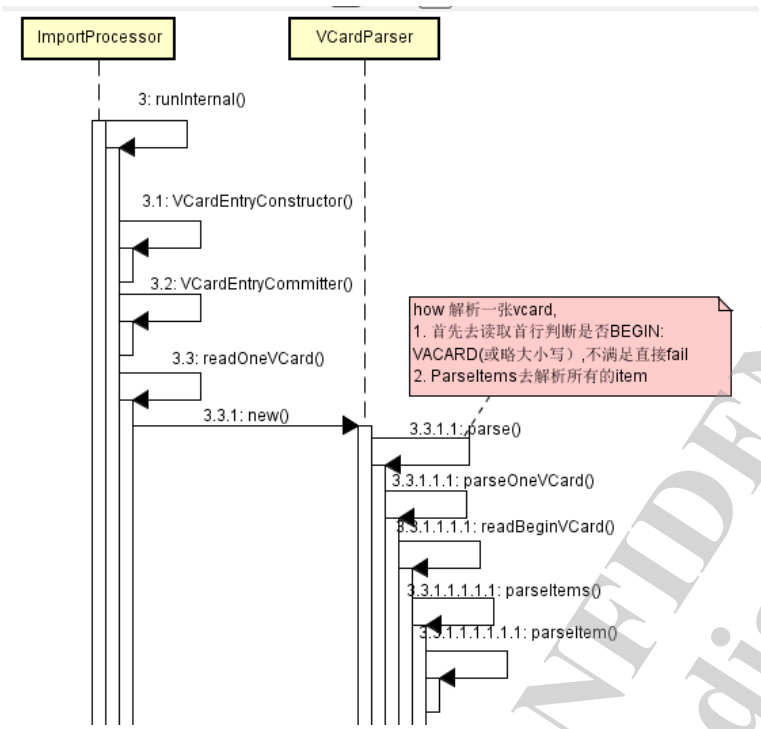
7.7.3 export Vcard

Vcard export vcf file, through the VcardComposer to complete, import and export implementation, are completed through the Vcard framework. Vcard framework is a tool, from its input and output point of view, the input for the destination file, need to export the contact for the vcf file, the entire export is the use of ExportProcessor this thread driven vcard framework to complete the processor's runInternal method Handle the ExportRequest request. The flow chart is shown below.



7.7.4 import Vcard

Import means that the Vcf file into the device, mainly through the Framework in the VcardParse to be responsible for analysis, and export the same, the import is to use ImportProcessor to do. ImportVcardActivity in the constructImportRequest method will go to a pre-resolution Vcard file. Formal analysis is through the ImportProcessor runInternal method, the use of VcardParser to resolve, the flow chart as shown below.



7.8 Multi-Window

7.8.1 Enable/disable Multi-window feature for app

For some app that does not want to support the Multi-window function, simply set the <activity> or <application> node properties android: resizeableActivity = ["true" | "false"]. For example, ContactsApp supports multi-window.

Releate file: packages/apps/Contacts/AndroidManifest.xml

multi-window

```
87      <!-- The main Contacts activity with the contact list, favorites, and groups. -->
88      <activity android:name=".activities.PeopleActivity"
89          android:label="@string/launcherActivityLabel"
90          android:theme="@style/PeopleTheme"
91          android:clearTaskOnLaunch="true"
92          android:launchMode="singleTop"
93          android:resizeableActivity="true"//true表示该App 支持multi-window模式，false表示不支持
94      >
```

7.8.2 Life-cycle

Enter the multi-window life cycle callback as follows:

OnMultiWindowModeChanged (true) -> onPause -> onSaveInstanceState -> onStop -> onDestroy -> onCreate -> onStart -> onRestoreInstanceState -> onResume -> onPause

First, the MultiWindows mode callback, the incoming parameter is true, that the current in the multi-window, and then the normal destruction and reconstruction, after the completion of reconstruction, onResume, due to the current App into the multi-window, did not get the focus So it will call once onPause ..

If you click on an App at this time, will be in this callback onResume, then click on a window with the other app, then the app will callback onPause. That is, multi-window, the switch App will trigger onPause. This point need to pay attention, such as your App is in the past pause in the pause video playback, then switch to other App, it will call back to this method, which pause the video, so the user experience is not very good, so you can put onStop Do it.

Exit multi-window callback:

OnSaveInstanceState -> onStop -> onDestroy -> onCreate -> onStart -> onRestoreInstanceState -> onResume -> onPause -> onMultiWindowModeChanged (false) -> onResume

App will change due to configuration, resulting in destruction, and then rebuild, and finally callback to onPause -> onMultiWindowModeChanged, the parameter is false, the last onResume.

7.9 RunTime Permission

7.9.1 Contacts Permission encapsulation

How to work for Contacts App?

The google design of Contacts App is that it will start RequestPermissionActivity to request permission when need request permission. when user granted,it will reCreate origin Activity. reCreate action is very import. it will bring up some bug.

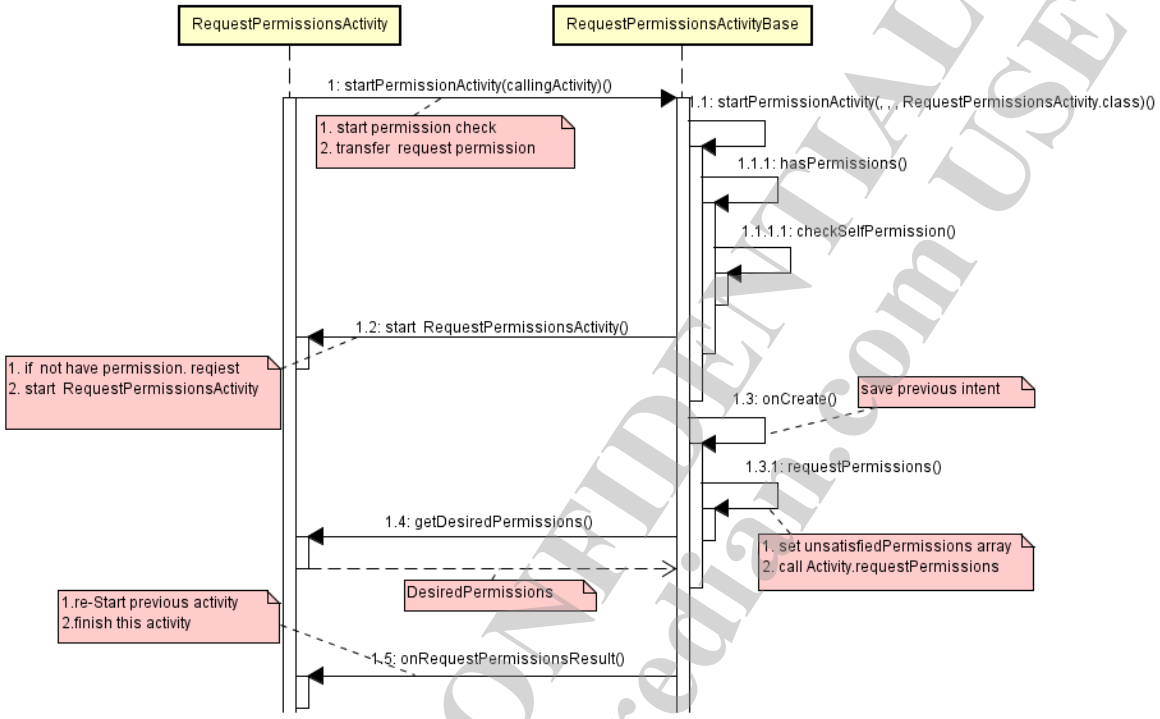
The inheritance structure:

RequestPermissionActivityBase. this is a base abstract Activity, it hold PreviousActiviyt Intent which be used for re-Create origin activity. the base class provide base function,like hasPermission, startPermissionActivity,requestPermission,

RequestPermissionActivity. This class override getRequiredPermissions, getDesiredPermission and receive request result .

RequestImportVcardPermissionActivity. This class should be used for import/Export function. because of request permission is diff with other activity, so it should separate part.

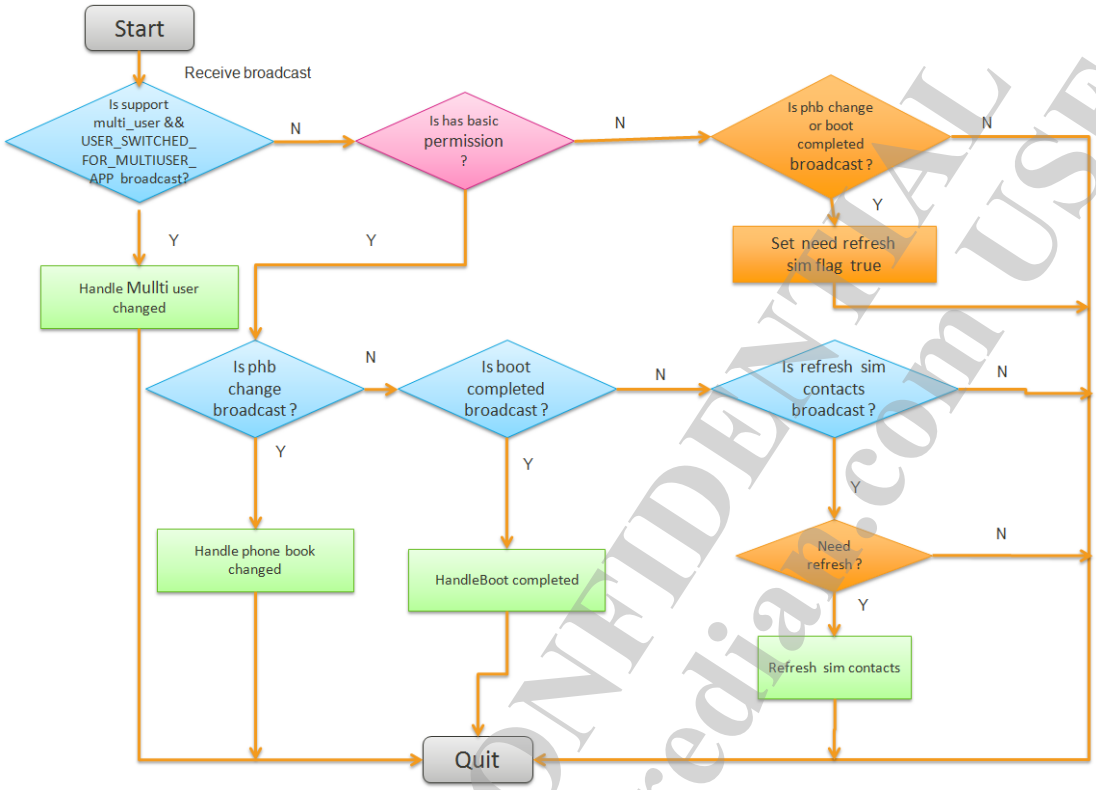
7.9.2 RequestPermisionsActivity work flow



7.10 SimProcessor

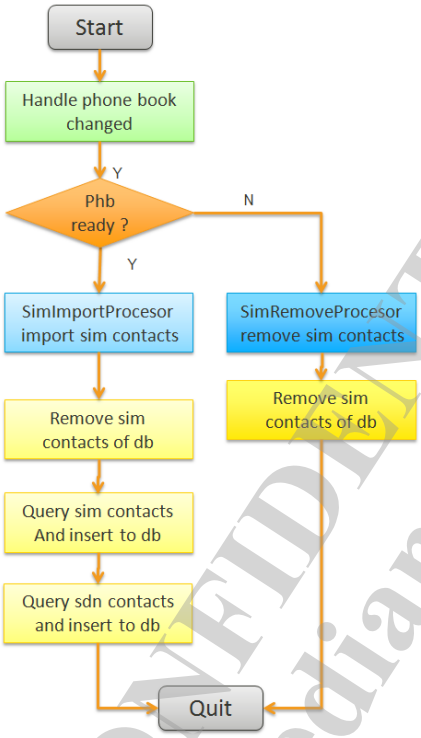
7.10.1 Start flow

When received different broadcast , it will start to do different things. The handle flow chart is below.



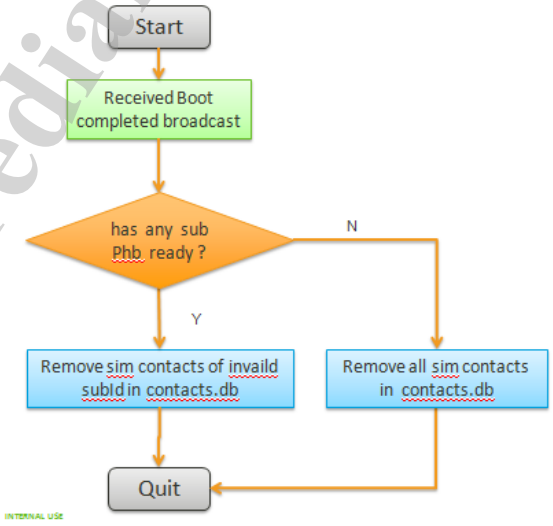
7.10.2 Handle phone book change

when received phb state changed broadcast within granted permission state. SimProcessor will start to handle load or remove sim contacts. the detail flow chat is below.



7.10.3 Handle boot completed

When device finished boot. it will send boot completed broadcast. Once SimProcessor received the broadcast, it will handle to remove sim contacts. there is one concern is that the received order between boot completed and phb change ready broadcast. SimProcessor will load sim contacts to contacts db when received phb change and phb state is ready. and then SImProcessor received boot completed broadcast, it just will remove sim contacts of invaild subId. other flow is more common that boot completed broadcast it camed before phb ready change broadcast, because of any sim card is unavailable. so SimProcessor just removed all sim contacts in this case. the two handle flow chat is below.



7.10.4 Refesh sim contacts

SimProcessor maintain a NEED_REFRESH_SIM_CONTACTS Flag value which is boolean type. the default is false. when no permission, it will be marked ture.when Contacts App is granted permission. Contacts will send out a REFRESH_SIM_CONTACTS broadcast. Then SimProcessor will handle to refresh sim contacts. peopleActivity:onCreate method send this broadcast.