1. 功能一： 修改文件夹名称
   1. **功能介绍**

**功能概述：**选中一个文件夹，读取用户输入的文件夹名称，将原有的文件夹名称进行修改，并将修改后的结果存储到本地数据库。

**操作步骤：**在主界面的右下角有一个菜单栏，用户点击按钮，选中特定文件夹，选择修改文件夹名称选项，输入拟修改的名称，点击确认按钮即可。

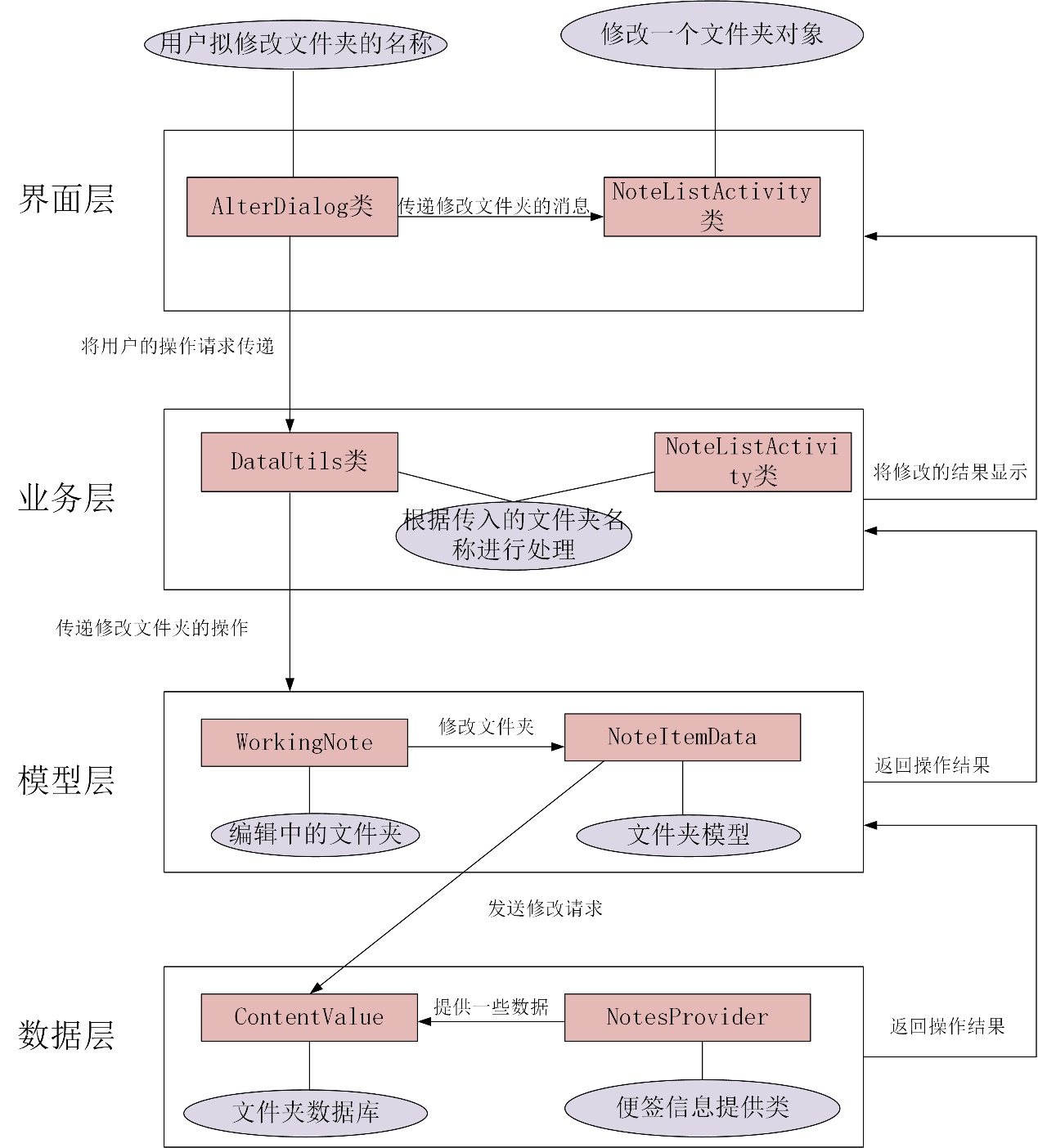
* 1. **业务请求与响应过程**

修改文件夹功能（下简称“功能”）的界面层主要由包ui中NotesListActivity类通过调用showCreateOrModifyFolderDialog()方法实现。showCreateOrModifyFolderDialog()中使用AlertDialog.Builder构建了一个对话框，显示一个包含文件夹名称输入框和确认取消按钮的对话框，并调用showSoftInput()方法显示软键盘。

功能的业务层由DataUtils类和NotesListActivity类实现。首先通过TextUtils的isEmpty()方法来判断文件夹名是否为空，其次调用checkVisibleFolderName()方法判断文件夹名是否存在，在不存在的情况下然后创建 ContentValues 对象并存储信息，最后使用 ContentResolver 的 insert()方法将文件夹中的信息插入到数据库中。

功能的模型层主要由model包中的WorkingNote类和NoteItemData类来实现，主要包括getFolderId()等方法。

功能的数据层主要由ContentValue类的对象来实现。创建一个ContentValue的对象，并调用put()方法将文件夹的名称和类型存入对象。最后使用 ContentResolver 的 insert() 方法将包含文件夹信息的 ContentValues 对象插入到数据库中。



* 1. **类的作用**
     1. **界面层相关的类**

1. **类名：NoteListActivity**

该类是小米便签的主列表界面，在该界面可以进行文件夹和便签的相关操作，例如创建文件夹，移动便签，新建便签等功能。

1. **类的精读：**NoteListActivity类的成员

|  |  |
| --- | --- |
| **成员** | **说明** |
| FOLDER\_NOTE\_LIST\_QUERY\_TOKEN | 文件夹便签表查询标识常量，传递查询信号 |
| FOLDER\_LIST\_QUERY\_TOKEN | 文件夹表查询标识常量，传递查询信号 |
| MENU\_FOLDER\_DELETE | 菜单文件夹删除标识常量，传递删除信号 |
| MENU\_FOLDER\_VIEW | 菜单文件夹查看标识常量，传递查看信号 |
| MENU\_FOLDER\_CHANGE\_NAME | 菜单文件夹改名标识常量，传递改名信号 |
| PREFERENCE\_ADD\_INTRODUCTION | 访问偏好设置字符串键，应用的介绍或配置信息 |
| ListEditState | 一个枚举类型成员，表示便签列表的编辑状态，包括便签列表、子文件夹和通话记录文件夹 |
| mState | 一个类成员变量，表示当前的列表编辑状态 |
| mBackgroundQueryHandler | 异步查询类对象，负责异步线程访问数据库取出相应的便签数据 |
| mNotesListAdapter | 便签列表的适配器，用于填充便签数据到界面 |
| mNotesListView | 便签列表的ListView控件，以列表的形式展示具体数据内容，并且根据数据的长度自适应屏幕显示 |
| mAddNewNote | 一个按钮类对象，用于添加新便签 |
| mDispatch | 一个布尔值，用于跟踪事件的分发状态 |
| mOriginY | 一个整数变量，用于记录Y轴坐标的值 |
| mDispatchY | 一个整数变量，用于跟踪Y轴坐标的值 |
| mTitleBar | 一个文本视图，用于显示标题栏的文本内容 |
| mCurrentFolderId | 一个长整数变量，表示当前所在文件夹的ID |
| mContentResolver | 一个ContentResolver对象，用于访问应用数据 |
| mModeCallBack | 一个回调对象，用于处理特定模式下的操作 |
| TAG | 一个用于日志记录的标签 |
| NOTES\_LISTVIEW\_SCROLL\_RATE | 一个整数常量，表示便签列表的滚动速率 |
| mFocusNoteDataItem | 一个便签数据项，用于跟踪当前焦点的便签 |
| NORMAL\_SELECTION | 一个用于数据库查询的字符串，表示普通数据筛选条件 |
| ROOT\_FOLDER\_SELECTION | 一个用于数据库查询的字符串，表示根文件夹的筛选条件 |
| REQUEST\_CODE\_OPEN\_NODE | 请求打开结点标识码常量 |
| REQUEST\_CODE\_NEW\_NODE | 请求新建结点标识码常量 |

NoteListActivity类的方法

|  |  |  |  |
| --- | --- | --- | --- |
| **方法** | **参数解释** | **方法作用** | **实现过程** |
| onCreate(Bundle savedInstanceState) | savedInstanceState 是一个Bundle对象，用于保存Activity的状态信息 | Activity的生命周期方法，用于初始化Activity，设置布局文件，调用initResources()方法初始化资源 | 首先调用了父类onCreate方法，然后通过setContentView方法设置了布局为R.layout.note\_list |
| onActivityResult(int requestCode, int resultCode, Intent data) | requestCode 是请求码，用于标识请求的来源resultCode是结果码，表示操作是否成功；data是包含返回数据的Intent对象 | 用于处理从其他Activity返回的结果，例如打开便签或创建新便签后的操作 | 处理Activity返回结果的方法，当其他Activity完成操作后返回结果时会调用此方法；否则调用父类onActivityResult方法来处理其他情况 |
| setAppInfoFromRawRes() | / | 初始化应用程序的介绍信息 | 首先获取SharedPreferences对象，检查是否已经插入过介绍信息，如果没有插入过则执行以下操作：从应用的原始资源（R.raw.introduction）中读取介绍信息；创建一个新的便签并将读取的介绍信息设置为便签内容；保存便签，并将插入介绍信息的标志设置为true，以避免重复插入 |
| onStart() | / | Activity的生命周期方法，在Activity即将可见时调用 | 调用startAsyncNotesListQuery()方法开始异步查询便签列表数据 |
| initResources() | / | 初始化Activity的资源和视图 | 初始化了mContentResolver、mBackgroundQueryHandler、mCurrentFolderId等成员变量，以及便签列表的视图（mNotesListView）和适配器（mNotesListAdapter）等 |
| updateMenu() | / | 更新多选模式下的操作菜单 | selectedCount 表示已选择的项的数量，通过更新下拉菜单的标题和全选/取消全选菜单项的状态 |
| onPrepareActionMode(ActionMode mode, Menu menu) | ActionMode mode：表示当前的多选模式操作栏；Menu menu：表示多选操作栏的菜单 | 操作模式即将显示时调用的方法，预加载 | 不加载 |
| onActionItemClicked(ActionMode mode, MenuItem item) | ActionMode mode：表示当前的多选模式操作栏；MenuItem item：表示被点击的菜单项 | 处理操作模式下菜单项的点击事件 | 不加载 |
| onDestroyActionMode(ActionMode mode) | ActionMode mode：表示当前的多选模式操作栏 | 在操作模式结束时调用的方法；用于清理多选模式相关的设置，恢复正常模式 | 清楚索引，修改相关设置 |
| finishActionMode() | / | 结束操作模式 | 调用mActionMode.finish() |
| onItemCheckedStateChanged(ActionMode mode, int position, long id, boolean checked) | ActionMode mode：表示当前的多选模式操作栏；int position：表示被选中或取消选中的便签项的位置；long id：表示被选中或取消选中的便签项的ID；boolean checked：表示便签项是否被选中 | 用于更新已选择项的状态和更新菜单 | 调用NitesListAdapter.setCheckedItem() |
| onMenuItemClick(MenuItem item) | MenuItem item：表示被点击的菜单项 | 处理下拉菜单中的菜单项点击事件 | 根据菜单项的ID执行不同的操作，如删除或移动已选择的便签 |
| startAsyncNotesListQuery() | / | 异步查询并加载便签列表数据 | 使用 mBackgroundQueryHandler 发起异步查询，根据当前文件夹ID来加载相应的便签数据 |
| showFolderListMenu(Cursor cursor) | cursor 是包含文件夹数据的光标对象 | 用于显示文件夹列表的对话框，并允许用户选择将便签移动到哪个文件夹 | 调用 DataUtils.batchMoveToFolder 方法将选定的便签移动到所选文件夹 |
| createNewNote() | / | 创建新的便签 | 创建一个新的便签编辑界面的 Intent，并指定当前文件夹ID |
| batchDelete() | / | 批量删除便签或将它们移动到回收站 | 异步执行的任务，根据同步模式，直接删除便签 |
| deleteFolder(long folderId) | folderId 表示要删除的文件夹的ID | 删除文件夹及其相关操作 | 根据同步模式，直接删除文件夹 |
| openNode(NoteItemData data) | data是一个 NoteItemData对象，表示要打开的便签数据 | 打开便签 | 创建一个 Intent 以查看指定便签的内容 |
| openFolder(NoteItemData data) | 同上 | 打开文件夹 | 根据所选文件夹的ID切换当前文件夹，加载该文件夹的便签列表 |
| onClick(View v) | v是触发点击事件的视图 | 处理点击事件 | 调用 createNewNote() 方法创建新便签 |
| showSoftInput() | / | 显示软键盘 | 调用相关方法 |
| hideSoftInput(View view) | view 是一个视图对象，用于获取窗口令牌以隐藏软键盘 | 隐藏软键盘 | 调用相关方法 |
| **showCreateOrModifyFolderDialog(boolean create)** | **create 是一个布尔值，表示是创建文件夹还是修改文件夹** | **显示创建或修改文件夹的对话框** | **弹出一个弹窗，将输入参数传入函数** |
| onBackPressed() | / | 处理返回按钮的点击事件 | 进入switch-case，根据相关条件执行相关操作 |
| updateWidget(int appWidgetId, int appWidgetType) | appWidgetId 表示小部件的ID，appWidgetType 表示小部件的类型 | 更新小部件 | 根据小部件的类型（2x2 或 4x4），构建适当的 Intent 并发送广播以更新小部件 |
| onContextMenuClosed(Menu menu) | menu表示菜单对象 | 当上下文菜单关闭时的回调 | 移除长按文件夹后的上下文菜单监听器 |
| onContextItemSelected(MenuItem item) | item表示菜单项目对象 | 处理上下文菜单中的选项 | 根据用户的选择，执行不同的操作，如查看文件夹、删除文件夹或更改文件夹名称 |
| onPrepareOptionsMenu(Menu menu) | menu表示菜单对象 | 在准备选项菜单时的回调 | 根据当前应用状态（便签列表、子文件夹、通话记录文件夹），设置不同的选项菜单 |
| onOptionsItemSelected(MenuItem item) | item表示菜单项目对象 | 处理选项菜单中的选项选择 | 根据用户选择执行不同的操作，如创建新文件夹、导出便签到文本文件、启动同步或设置 |
| onSearchRequested() | / | 处理搜索请求 | 启动搜索操作，用于搜索便签 |
| exportNoteToText() | / | 将便签导出为文本文件 | 使用 BackupUtils 类执行导出操作，并根据结果显示相应的对话框提示 |
| isSyncMode() | / | 检查是否启用了同步模式 | 根据用户的设置检查是否配置了同步帐户 |
| startPreferenceActivity() | / | 启动应用的设置界面 | 跳转到设置界面，允许用户配置应用的各种参数 |
| startQueryDestinationFolders() | / | 开始查询可作为目标的文件夹列表 | 构建查询，获取可以移动便签的文件夹列表，并在查询完成后显示选择目标文件夹的对话框 |
| onItemLongClick(AdapterView<?> parent, View view, int position, long id) | parent：表示触发长按事件的父级适配器视图; view：表示用户长按的特定列表项视图; position：表示被长按的列表项在适配器中的位置; id：表示被长按的列表项的唯一标识符 | 处理长按列表项的事件 | 如果被长按的列表项是文件夹（Notes.TYPE\_FOLDER），则将文件夹上下文菜单创建监听器设置为 mFolderOnCreateContextMenuListener。这样，当用户点击该文件夹时，将弹出文件夹上下文菜单 |

* + 1. **业务层相关的类**
  1. **类名：**DataUtils

1. **类的作用：**

一个数据处理的工具类，能够包含了一系列用于处理笔记数据的静态方法。这些方法涵盖了笔记的批量删除、移动到文件夹、获取文件夹的笔记部件信息、检查文件夹名是否存在等功能。

(**2) 类的精读：**

表1 类的成员

|  |  |
| --- | --- |
| **成员** | **说明** |
| TAG | public static final String TAG = "DataUtils"; |

表2 类中的方法

|  |  |  |  |
| --- | --- | --- | --- |
| **方法** | **参数解释** | **方法作用** | **实现过程** |
| getUserFolderCount | ContentResolver:  查询便签内容  （Notes.CONTENT\_NOTE\_URI）获取游标（Cursor） | 获取用户文件夹的数量 | 查询便签内容获取游标  查询时指定了特定条件：  选择字段 "COUNT(\*)"，表示查询的结果是文件夹的数量。  如果游标不为空（则移动游标到第一行  尝试从游标中获取第一列的整数值，即文件夹的数量最后，无论是否发生异常，都需要关闭游标  返回获取到的文件夹数量。 |
| batchDeleteNotes | HashSet<Long>：id列表  ContentResolver: 一个内容解析器  ContentUris.  withAppendedId():用于构建便签的特定URI。  operationList操作列表 | 根据传入的便签 ID 集合进行批量删除操作 | 遍历 ids 集合中的每个便签ID，根据ID构建一个 对象，指定要删除的具体便签。  利用内容提供者操作类执行与内容提供者相关的操作，例如插入、更新、删除等。  返回结果数组 results 中的第一个结果判断操作是否成功。如果结果为空或第一个结果为空，表示删除失败。  如果删除操作成功，则返回 true，否则返回 false。 |
| moveNoteToFoler | resolver：用于与应用程序的数据提供者进行通信。  id：要移动的便签的唯一标识符。  srcFolderId：便签当前所在的文件夹的唯一标识符。  desFolderId：要移动到的目标文件夹的唯一标识符。 | 将指定的便签移动到不同的文件夹中 | 创建一个对象，用于存储要更新的便签的新值。之后更新便签的属性值。  将更新后的 values 对象应用于指定的便签。指定要更新的便签的位置。 |
| batchMoveToFolder | resolver：用于交互。  ids：包含要移动的便签的 ID。  folderId：要移动到的目标文件夹的 ID。 | 将一组便签移动到指定文件夹 | 对于 ids 中的每个便签 ID，创建一个Operation对象，并设置更新的 URI 和要修改的列的值。这里的列包括父文件夹ID和本地修改标志。  将创建的Operation对象添加到 operationList 中。  使用 resolver 对象调用 applyBatch() 方法，将 operationList 中的操作应用到 ContentProvider 中。  根据返回的结果进行判断：如果 results 为空、长度为 0 或第一个元素为空，则记录一条日志并返回 false。  如果没有抛出异常，则返回 true。否则，记录异常信息并返回 false。 |

* + 1. **模型层相关的类**

**1、 类名：WorkingNote**

1. **类的作用：**

定义创建、加载、保存和管理便签一些属性的具体操作，包括文本内容、提醒时间、背景颜色等属性。允许用户创建新的便签、加载已有的便签，以及保存对便签的修改。

1. **类的精读：**

表1 类的成员

|  |  |
| --- | --- |
| 成员 | 说明 |
| mNote | Note对象，用于管理便签的元数据和内容 |
| mNoteId | 便签的唯一标识符 |
| mContent | 便签的文本内容 |
| mMode | 便签的模式 |
| mBgColorId | 便签的背景颜色ID |
| mWidgetId | 小部件（Widget）的ID |
| mWidgetType | 小部件类型 |
| mFolderId | 便签所属的文件夹ID |
| mContext | Android应用程序的上下文 |
| TAG | 标记当前便签状态 |
| mIsDeleted | 表示便签是否已被标记为删除 |
| mNoteSettingStatusListener | 便签设置状态的监听器 |
| DATA\_PROJECTION | 查询数据时使用的投影 |
| NOTE\_PROJECTION | 查询便签时使用的投影 |
| NoteSettingChangedListener | 定义便签设置改变监听器的接口 |

表2 类中的方法

|  |  |  |  |
| --- | --- | --- | --- |
| **方法** | **参数解释** | **方法作用** | **实现过程** |
| setBgClorId(int id) | Id:用来修改当前颜色的值 | 修改当前便签背景颜色的ID值 | 先检查选择器是否可见，如果可见并且触摸操作不在选择器范围内则将其设置为不可见，并返回true来拦截触摸事件。 |
| getBgcolorId() | 无 | 返回当前的背景颜色的ID值 | Return直接返回 |
| getTitleBgResId() | 无 | 返回当前的标题颜色的ID值 | 通过背景颜色ID值来调用NoteBgResource内的方法来匹配返回标题的ID值 |
| WorkingNote(Context context, long folderId) | context:Android应用程序的上下文。  folderId:笔记所属文件夹的ID。 | 创建一个新的笔记对象 | 实例化对象 |
| getBgcolorId() | 无 | 返回当前的背景颜色的ID值 | Return直接返回 |
| getBgColorResId(): | 无 | 获取笔记背景颜色资源ID | Return直接返回 |
| getCheckListMode() | 无 | 获取笔记模式 | Return直接返回 |
| getModifiedDate() | 无 | 获取笔记修改日期 | Return直接返回 |
| getAlertDate() | 无 | 获取提醒日期 | Return直接返回 |
| getContent() | 无 | 获取笔记内容 | Return直接返回 |
| getNoteId() | 无 | 获取笔记ID | Return直接返回 |
| getWidgetType() | 无 | 获取笔记对应的小部件类型 | Return直接返回 |
| getTitleBgResId() | 无 | 获取笔记标题背景资源ID | Return直接返回 |
| createEmptyNote(Context context, long folderId, int widgetId, int widgetType, int defaultBgColorId) | context: Android 应用程序的上下文。  folderId: 笔记所属文件夹的ID。  widgetId: 笔记对应的小部件的ID。  widgetType: 笔记对应的小部件类型。  defaultBgColorId: 笔记的默认背景颜色ID。 | 创建一个空白笔记 | 通过调用一系列set方法，设置笔记的初始情况 |
| getWidgetType() | 无 | 获取笔记对应的小部件类型 | Return直接返回 |

* 1. **类名：NoteItemData**

1. **类的作用：表示便签项的数据。该类通过从数据库查询结果的 Cursor 中获取相应的列数据来初始化实例。**
2. **类的精读：**

表1 类的成员

|  |  |
| --- | --- |
| 成员 | 说明 |
| mId | 便签项的ID |
| mId | 提醒时间 |
| mBgColorId | 背景颜色ID |
| mCreatedDate | 创建时间 |
| mHasAttachment | 是否有附件 |
| mModifiedDate | 最近修改时间 |
| mNotesCount | 笔记数量 |
| mParentId | 父文件夹ID |
| mSnippet | 便签内容的摘要 |
| mType | 便签类型 |
| mWidgetId | Widget的ID |
| mWidgetType | Widget类型 |
| mName | 联系人名字 |
| mPhoneNumber | 联系人电话号码 |

表2 类中的方法

|  |  |  |  |
| --- | --- | --- | --- |
| **方法** | **参数解释** | **方法作用** | **实现过程** |
| NoteItemData(Context context, Cursor cursor) | Context:文本  Cursor：游标 | 构造函数，从数据库查询结果的Cursor中初始化便签项数据。 | 调用相关方法，实现数据库的查询和数据的初始化。 |
| isOneFollowingFolder() | 无 | 判断是否是一个便签项后面跟随一个文件夹 | 调用相关方法，实现是否跟随一个文件夹的判断 |
| isMultiFollowingFolder() | 无 | 判断是否是一个便签项后面跟随多个文件夹 | 调用相关方法，实现是否跟随多个文件夹的判断 |
| isLast() | 无 | 判断是否是列表中的最后一个项 | 调用相关方法，实现对是否是列表中的最后一个项的判断 |
| getCallName() | 无 | 获取联系人名字 | Return直接返回 |
| isFirst() | 无 | 判断是否是列表中的第一个项 | 调用相关方法，实现是否是列表中的第一个项的判断 |
| isSingle() | 无 | 判断列表是否只有一个项 | 调用相关方法，实现列表是否只有一个项的判断 |
| getId() | 无 | 获取便签项的ID | Return直接返回 |
| getAlertDate() | 无 | 获取提醒日期 | Return直接返回 |
| getCreatedDate() | 无 | 获取创建时间 | Return直接返回 |
| hasAttachment() | 无 | 判断是否有附件 | 调用相关方法，实现对是否有附件的判断 |
| getModifiedDate() | 无 | 获取最近修改时间 | Return直接返回 |
| getBgColorId() | 无 | 获取背景颜色ID | Return直接返回 |
| getWidgetType() | 无 | 获取父文件夹ID | Return直接返回 |

* + 1. **数据层相关的类**

**1、类名：**ContentValues

1. **类的作用：此类用于存储 ContentResolver 可以处理的一组值。**
2. **类的精读：**

表1 类的成员

|  |  |
| --- | --- |
| **成员** | **说明** |
| mValues | HashMap对象，用以存储键值对 |

表2 类中的方法

|  |  |  |  |
| --- | --- | --- | --- |
| **方法** | **参数解释** | **方法作用** | **实现过程** |
| **put(String key, Byte value)** | Key：键  Value 值 | 向 ContentValues 中添加一个 byte 类型的键值对 | 直接调用HashMap中的put方法插入对应键值对 |
| **put(String key, Short value)** | Key：键  Value 值 | 向 ContentValues 中添加一个 **Short** 类型的键值对 | 直接调用HashMap中的put方法插入对应键值对 |
| **put(String key, Integer value)** | Key：键  Value 值 | 向 ContentValues 中添加一个 **Integer** 类型的键值对 | 直接调用HashMap中的put方法插入对应键值对 |
| **put(String key, Long value)** | Key：键  Value 值 | 向 ContentValues 中添加一个 **Long** 类型的键值对 | 直接调用HashMap中的put方法插入对应键值对 |
| **put(String key, Float value)** | Key：键  Value 值 | 向 ContentValues 中添加一个 **Float** 类型的键值对 | 直接调用HashMap中的put方法插入对应键值对 |
| **put(String key,** **Double value)** | Key：键  Value 值 | 向 ContentValues 中添加一个 **Double** 类型的键值对 | 直接调用HashMap中的put方法插入对应键值对 |
| **put(String key, String value)** | Key：键  Value 值 | 向 ContentValues 中添加一个 **String** 类型的键值对 | 直接调用HashMap中的put方法插入对应键值对 |
| **put(String key, byte[] value)** | Key：键  Value 值 | 向 ContentValues 中添加一个 byte数组 类型的键值对 | 直接调用HashMap中的put方法插入对应键值对 |
| **putNull(String key)** | Key：键 | 向 ContentValues 中添加一个 空值的键值对 | 直接调用HashMap中的put方法插入对应键值对 |
| **get(String key)**: | Key：键 | 根据键获取对应的值 | 直接调用HashMap中的get方法 |
| **containsKey(String key)** | Key：键 | 检查是否包含指定键 | 直接调用HashMap中的contains方法 |
| **remove(String key)** | Key：键 | 根据键移除键值对 | 直接调用HashMap中的remove方法 |
| **clear()** |  | 清空 ContentValues 中的所有键值对。 | 直接调用HashMap中的clear方法 |