

计算机科学与技术学院

智记便签项目

软件设计规格说明

学 号: 2023064128

专 业: 计算机科学与技术

学 生 姓 名: 潘铮

学 号: 2023064129

专 业: 计算机科学与技术

学 生 姓 名: 季凯阳

人 员 分 工:

合 作 方 法:

任 课 教 师: 郑丽颖教授

# 目录

<b>1</b>	<b>引言</b>	<b>2</b>
1.1	编写目的	2
1.2	项目背景	2
1.3	定义、首字母缩写词和缩略语	2
1.4	参考资料	3
<b>2</b>	<b>系统架构设计</b>	<b>3</b>
2.1	系统总体架构	3
2.2	架构层次说明	4
2.2.1	表示层 ( Presentation Layer )	4
2.2.2	业务逻辑层 ( Business Logic Layer )	4
2.2.3	数据访问层 ( Data Access Layer )	4
2.3	设计模式应用	5
<b>3</b>	<b>模块详细设计</b>	<b>5</b>
3.1	便签编辑模块	5
3.1.1	模块概述	5
3.1.2	类设计	5
3.1.3	主要接口	5
3.1.4	算法设计	7
3.2	导出分享模块	7
3.2.1	模块概述	7
3.2.2	类设计	7
3.2.3	主要接口	7
3.2.4	算法设计	8
3.3	数据管理模块	9
3.3.1	模块概述	9
3.3.2	类设计	9
3.3.3	主要接口	9
<b>4</b>	<b>数据库设计</b>	<b>10</b>
4.1	数据库概要	10
4.2	数据表设计	10
4.2.1	note 表	10
4.2.2	data 表	11

4.3	数据库关系图	11
4.4	索引设计	12
4.5	数据完整性约束	12
5	接口设计	13
5.1	外部接口	13
5.1.1	ContentProvider 接口	13
5.1.2	Intent 接口	13
5.2	内部接口	13
5.2.1	模块间接口	13
6	用户界面设计	14
6.1	界面结构	14
6.2	便签列表界面	14
6.3	便签编辑界面	15
6.4	格式选择对话框	15
7	非功能性设计	16
7.1	性能设计	16
7.1.1	响应时间优化	16
7.1.2	内存优化	16
7.2	可靠性设计	16
7.2.1	异常处理	16
7.2.2	数据备份	16
7.3	安全性设计	17
7.3.1	数据安全	17
7.3.2	权限管理	17
7.4	可维护性设计	17
7.4.1	代码规范	17
7.4.2	模块化设计	17
7.5	可移植性设计	18
8	测试设计	18
8.1	单元测试	18
8.1.1	测试范围	18
8.1.2	测试框架	18
8.2	集成测试	18
8.2.1	测试场景	18

8.3	系统测试	19
8.3.1	功能测试	19
8.3.2	性能测试	19
8.3.3	兼容性测试	19
9	部署设计	19
9.1	构建配置	19
9.1.1	Gradle 配置	19
9.1.2	依赖库	20
9.2	发布流程	20
9.3	版本升级策略	20
9.3.1	数据库升级	20
9.3.2	兼容性保证	20
10	附录	21
10.1	术语表	21
10.2	设计变更记录	21

# 1 引言

## 1.1 编写目的

本文档是智记便签 (Smart Note) 项目的软件设计规格说明书 (Software Design Specification, SDD), 旨在详细描述系统的架构设计、模块设计、接口设计和数据库设计。本文档基于已完成的软件需求规格说明书 (SRS), 为开发人员提供详细的设计方案和实现指导。

本文档的预期读者包括：

- 项目开发人员：作为编码实现的依据
- 测试人员：作为测试用例设计的参考
- 项目管理人员：用于进度跟踪和质量评估
- 维护人员：作为系统维护和升级的参考文档

## 1.2 项目背景

项目名称：智记便签 (Smart Note)

项目背景：智记便签是在小米便签基础上进行功能增强和优化的移动端便签应用。本项目新增 Markdown 富文本支持和增强的导出分享功能，为用户提供更加专业和高效的笔记体验。

## 1.3 定义、首字母缩写词和缩略语

术语	定义
SDD	Software Design Description, 软件设计规格说明
SRS	Software Requirements Specification, 软件需求规格说明
MVC	Model-View-Controller, 模型-视图-控制器架构模式
DAO	Data Access Object, 数据访问对象
UML	Unified Modeling Language, 统一建模语言
API	Application Programming Interface, 应用程序编程接口
UI	User Interface, 用户界面
Markdown	一种轻量级标记语言
ContentProvider	Android 平台的数据共享机制

SQLite

轻量级关系型数据库管理系统

## 1.4 参考资料

1. 智记便签软件需求规格说明书 (SRS\_ 智记便签.tex)
2. GB/T 8567-2006 计算机软件文档编制规范
3. Android 开发者文档：<https://developer.android.com/>
4. 小米便签 (MiCode Notes) 开源项目：<https://github.com/MiCode/Notes>
5. 《软件工程：实践者的研究方法》(第8版)

## 2 系统架构设计

### 2.1 系统总体架构

智记便签采用经典的 Android 应用三层架构设计，将系统分为表示层 (Presentation Layer)、业务逻辑层 (Business Logic Layer) 和数据访问层 (Data Access Layer)。图 1 展示了系统的总体架构。

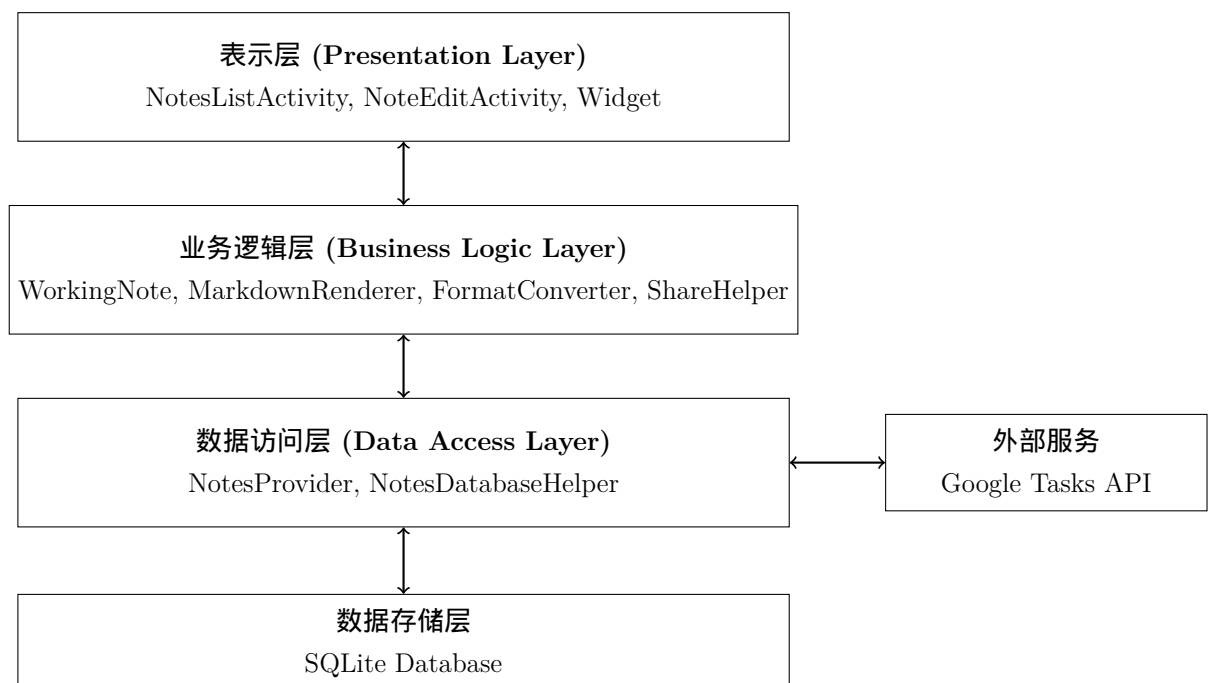


图 1: 系统总体架构图

## 2.2 架构层次说明

### 2.2.1 表示层 ( Presentation Layer )

表示层负责与用户的交互，包括显示界面和接收用户输入。主要组件包括：

- **NotesListActivity**：显示便签列表，提供便签的浏览、搜索和分类功能
- **NoteEditActivity**：便签编辑界面，支持文本输入、Markdown 编辑和预览
- **AlarmAlertActivity**：提醒通知界面
- **NoteWidget**：桌面小部件，提供快速访问和编辑功能
- **FormatDialog**：导出格式选择对话框

### 2.2.2 业务逻辑层 ( Business Logic Layer )

业务逻辑层处理应用的核心功能和业务规则。主要组件包括：

- **WorkingNote**：便签的业务模型，封装便签的操作逻辑
- **MarkdownRenderer**：Markdown 渲染引擎，将 Markdown 文本转换为格式化显示
- **FormatConverter**：格式转换器，支持将便签转换为多种导出格式
- **ShareHelper**：分享助手，处理便签的分享逻辑
- **SyncManager**：同步管理器，处理云同步逻辑

### 2.2.3 数据访问层 ( Data Access Layer )

数据访问层提供统一的数据访问接口，封装数据库操作细节。主要组件包括：

- **NotesProvider**：ContentProvider 实现，提供标准化的数据访问接口
- **NotesDatabaseHelper**：数据库助手类，管理数据库创建和版本升级



## 2.3 设计模式应用

系统采用多种设计模式以提高代码质量和可维护性：

- MVC 模式：整体架构采用 MVC 模式，分离界面、逻辑和数据
- 单例模式：MarkdownRenderer、SyncManager 等工具类采用单例模式
- 工厂模式：FormatConverter 使用工厂模式创建不同格式的转换器
- 观察者模式：数据变化时使用 ContentObserver 通知 UI 更新
- 策略模式：不同的导出格式使用不同的转换策略

## 3 模块详细设计

### 3.1 便签编辑模块

#### 3.1.1 模块概述

便签编辑模块是系统的核心模块，负责便签的创建、编辑、保存和显示。该模块新增了 Markdown 编辑和预览功能。

#### 3.1.2 类设计

图 2 展示了便签编辑模块的类图。

#### 3.1.3 主要接口

NoteEditActivity 主要接口：

- togglePreview(): void
  - 功能：切换编辑模式和预览模式
  - 前置条件：Activity 已初始化
  - 后置条件：界面切换到相应模式
- renderMarkdown(): void
  - 功能：渲染 Markdown 文本为格式化显示
  - 前置条件：存在待渲染的 Markdown 文本
  - 后置条件：预览视图显示渲染结果

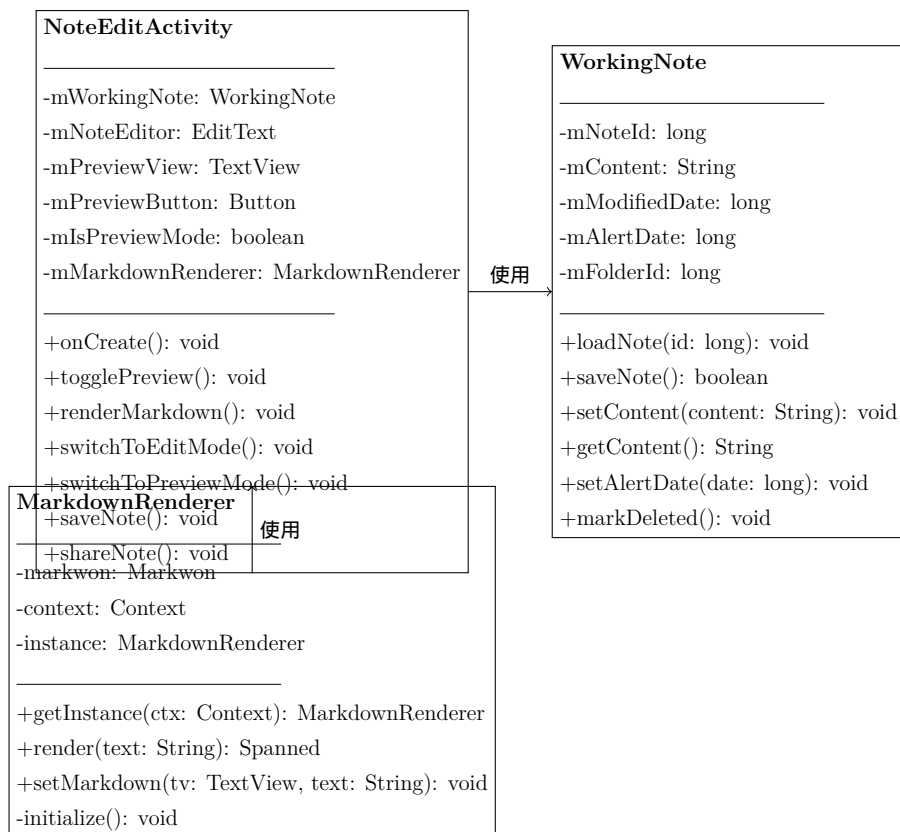


图 2: 便签编辑模块类图

- **saveNote(): void**
  - 功能：保存当前便签到数据库
  - 前置条件：WorkingNote 已初始化
  - 后置条件：便签内容保存成功

**MarkdownRenderer 主要接口：**

- **getInstance(Context): MarkdownRenderer**
  - 功能：获取 MarkdownRenderer 单例实例
  - 参数：Context - Android 上下文
  - 返回：MarkdownRenderer 实例
- **render(String): Spanned**
  - 功能：将 Markdown 文本渲染为 Spanned 对象
  - 参数：text - Markdown 格式文本
  - 返回：渲染后的 Spanned 对象

### 3.1.4 算法设计

Markdown 预览切换算法：

1. 检查当前模式（编辑/预览）
2. 如果当前为编辑模式：
  - (a) 获取 EditText 中的文本内容
  - (b) 调用 MarkdownRenderer.render() 渲染文本
  - (c) 将渲染结果设置到 TextView
  - (d) 隐藏 EditText，显示 TextView
  - (e) 更新按钮文本为”编辑”
3. 如果当前为预览模式：
  - (a) 隐藏 TextView，显示 EditText
  - (b) 更新按钮文本为”预览”

## 3.2 导出分享模块

### 3.2.1 模块概述

导出分享模块支持将便签导出为多种格式（纯文本、Markdown、HTML）并通过系统分享功能发送。

### 3.2.2 类设计

图 3展示了导出分享模块的类图。

### 3.2.3 主要接口

FormatConverter 主要接口：

- convertToPlainText(String): String
  - 功能：转换为纯文本格式
  - 参数：text - 原始文本
  - 返回：纯文本内容
- convertToHtml(String): String
  - 功能：转换为 HTML 格式

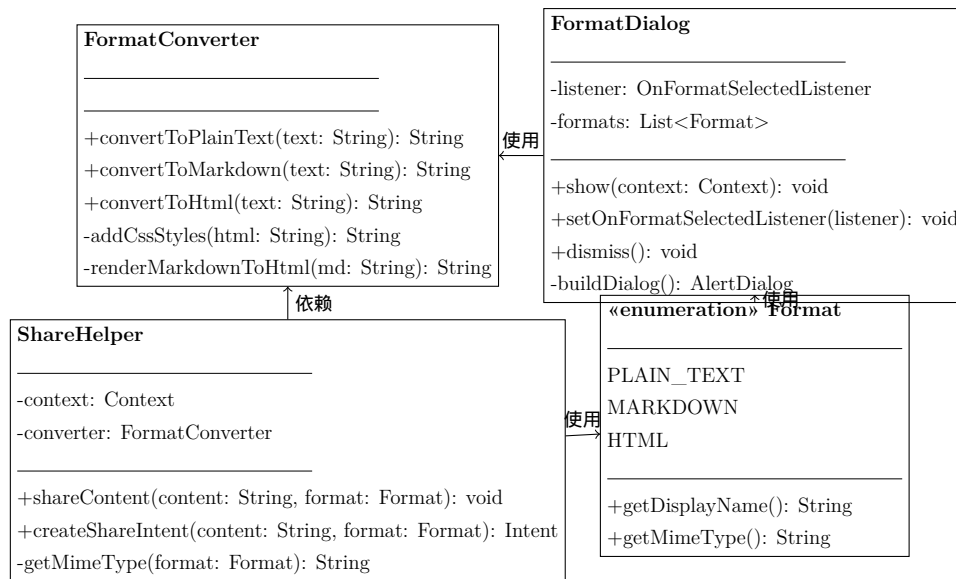


图 3: 导出分享模块类图

- 参数：text - Markdown 文本
- 返回：包含 CSS 样式的完整 HTML 文档

ShareHelper 主要接口：

- shareContent(String, Format): void
  - 功能：分享内容到其他应用
  - 参数：content - 内容文本, format - 导出格式
  - 后置条件：启动系统分享界面

### 3.2.4 算法设计

HTML 格式转换算法：

1. 检查文本是否为 Markdown 格式
2. 使用 MarkdownRenderer 将 Markdown 渲染为 HTML
3. 创建完整的 HTML 文档结构
4. 添加 CSS 样式：
  - (a) 设置字体、字号、行距
  - (b) 定义标题样式
  - (c) 定义列表样式

(d) 定义代码块样式

5. 返回完整 HTML 文档

3.3 数据管理模块

3.3.1 模块概述

数据管理模块负责所有数据的持久化存储和访问，通过 ContentProvider 提供统一的数据访问接口。

3.3.2 类设计

图 4展示了数据管理模块的类图。

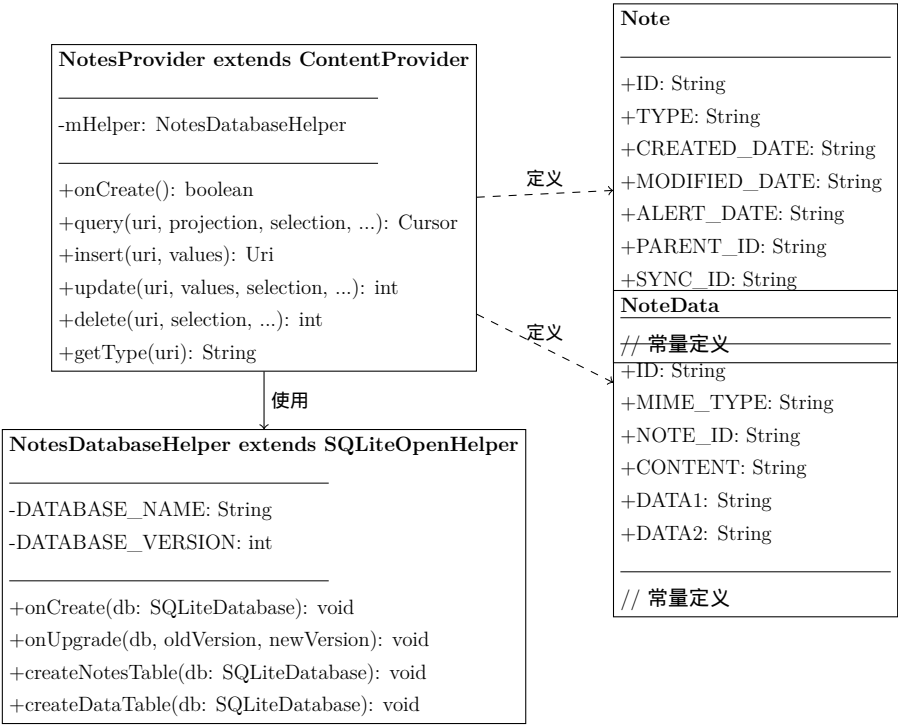


图 4: 数据管理模块类图

3.3.3 主要接口

NotesProvider 主要接口：

- query(Uri, String[], String, String[], String): Cursor
  - 功能：查询数据库数据
  - 参数：uri - 查询 URI, projection - 查询列, selection - WHERE 子句

- 返回：查询结果 Cursor
- insert(Uri, ContentValues): Uri
  - 功能：插入新数据
  - 参数：uri - 插入 URI, values - 数据值
  - 返回：新插入数据的 URI
- update(Uri, ContentValues, String, String[]): int
  - 功能：更新数据
  - 参数：uri - 更新 URI, values - 新数据值, selection - WHERE 子句
  - 返回：受影响的行数

## 4 数据库设计

### 4.1 数据库概要

智记便签使用 SQLite 数据库进行本地数据存储。数据库名称为 notes.db，当前版本为 3。数据库包含两个核心数据表：note 表和 data 表。

### 4.2 数据表设计

#### 4.2.1 note 表

note 表存储便签的元数据信息。表结构如下：

字段名	类型	约束	说明
_id	INTEGER	PRIMARY KEY	便签唯一标识符，自增
type	INTEGER	NOT NULL	便签类型（0= 普通便签, 1= 文件夹）
created_date	INTEGER	NOT NULL	创建时间（Unix 时间戳，毫秒）
modified_date	INTEGER	NOT NULL	最后修改时间（Unix 时间戳，毫秒）
sync_id	INTEGER		云同步 ID，与 Google Tasks 关联

字段名	类型	约束	说明
parent_id	INTEGER	DEFAULT 0	父文件夹 ID , 0 表示根目录
alerted_date	INTEGER		提醒时间 ( Unix 时间戳 , 毫秒 )
widget_id	INTEGER		关联的桌面小部件 ID
widget_type	INTEGER		小部件类型
bg_color_id	INTEGER		背景颜色 ID
has_attachment	INTEGER	DEFAULT 0	是否有附件 ( 预留字段 )
notes_count	INTEGER	DEFAULT 0	文件夹包含的便签数量
version	INTEGER	DEFAULT 0	版本号 , 用于同步冲突解决

#### 4.2.2 data 表

data 表存储便签的具体内容数据。表结构如下：

字段名	类型	约束	说明
_id	INTEGER	PRIMARY KEY	数据唯一标识符 , 自增
mime_type	TEXT	NOT NULL	数据 MIME 类型 ( 如 text/note )
note_id	INTEGER	NOT NULL	关联的便签 ID ( 外键 )
content	TEXT	NOT NULL	实际内容 ( Markdown 原文 )
data1	INTEGER		扩展字段 1 ( 如联系人 ID )
data2	INTEGER		扩展字段 2
data3	TEXT		扩展字段 3 ( 如联系人姓名 )
data4	TEXT		扩展字段 4
data5	TEXT		扩展字段 5

### 4.3 数据库关系图

图 5 展示了数据库的实体关系图。

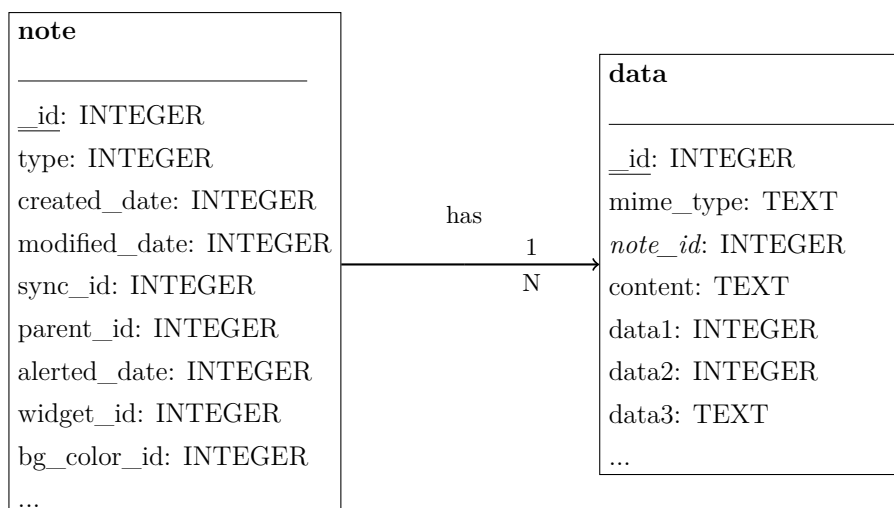


图 5: 数据库实体关系图

## 4.4 索引设计

为提高查询性能，在以下字段上建立索引：

- **note 表：**
  - parent\_id 索引：加速按文件夹查询便签
  - modified\_date 索引：加速按修改时间排序
  - alerted\_date 索引：加速查询待提醒的便签
- **data 表：**
  - note\_id 索引：加速根据便签 ID 查询内容
  - mime\_type 索引：加速按类型查询数据

## 4.5 数据完整性约束

- **主键约束：**\_\_id 字段为主键，确保唯一性
- **外键约束：**data 表的 note\_id 引用 note 表的 \_\_id，级联删除
- **非空约束：**关键字段（type, created\_date, modified\_date, content 等）不允许为空
- **默认值约束：**parent\_id 默认为 0，has\_attachment 默认为 0



## 5 接口设计

### 5.1 外部接口

#### 5.1.1 ContentProvider 接口

NotesProvider 实现了标准的 ContentProvider 接口，供外部应用或组件访问便签数据。

URI 定义：

URI	说明
content://com.xiaomi.notes/note	访问所有便签
content://com.xiaomi.notes/note/ <del>noteId</del>	访问特定 ID 的便签
content://com.xiaomi.notes/data	访问所有数据
content://com.xiaomi.notes/data/ <del>noteId</del>	访问特定 ID 的数据
content://com.xiaomi.notes/note/ <del>noteId</del> /data	访问特定便签的所有数据

MIME 类型：

- vnd.android.cursor.dir/note：便签集合
- vnd.android.cursor.item/note：单个便签
- vnd.android.cursor.dir/data：数据集合
- vnd.android.cursor.item/data：单条数据

#### 5.1.2 Intent 接口

分享 Intent：

- Action: ACTION\_SEND
- Type: 根据导出格式确定（text/plain, text/markdown, text/html）
- Extra: EXTRA\_TEXT（内容），EXTRA\_SUBJECT（标题）

### 5.2 内部接口

#### 5.2.1 模块间接口

表示层 → 业务逻辑层：

- WorkingNote.loadNote(long noteId)：加载便签

- `WorkingNote.saveNote()`：保存便签
- `MarkdownRenderer.render(String text)`：渲染 Markdown
- `FormatConverter.convertToHtml(String text)`：转换为 HTML

业务逻辑层 → 数据访问层：

- `ContentResolver.query(...)`：查询数据
- `ContentResolver.insert(...)`：插入数据
- `ContentResolver.update(...)`：更新数据
- `ContentResolver.delete(...)`：删除数据

## 6 用户界面设计

### 6.1 界面结构

图 6展示了应用的界面导航结构。

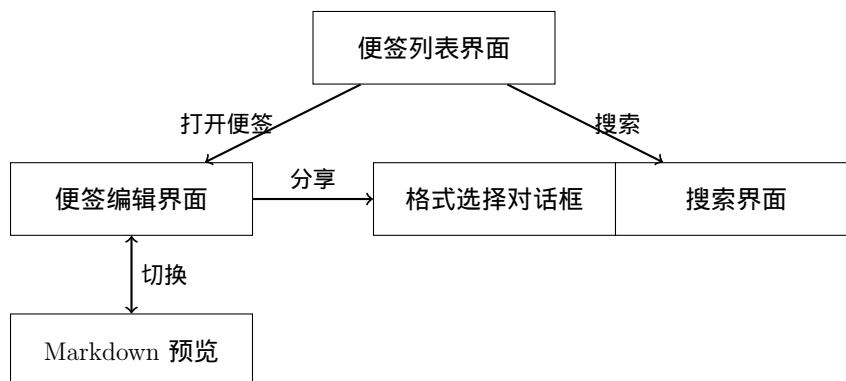


图 6: 用户界面导航结构图

### 6.2 便签列表界面

界面组成：

- 标题栏：显示应用名称”智记便签”，包含搜索图标
- 便签列表：ListView 或 RecyclerView 显示便签，支持滑动
- 浮动按钮：右下角的”+”按钮，用于创建新便签

- 侧边栏（可选）：显示文件夹列表

列表项设计：

- 第一行：便签标题或首行内容（粗体）
- 第二行：便签预览内容（灰色，最多显示 2 行）
- 第三行：修改时间、提醒图标（如有）
- 左侧：背景颜色标识条

## 6.3 便签编辑界面

编辑模式布局：

- 标题栏：返回按钮、便签标题（可编辑）、菜单按钮
- 编辑区域：全屏 `EditText`，支持多行文本输入
- 底部工具栏：
  - 预览按钮：切换到预览模式
  - 分享按钮：打开分享对话框
  - 提醒按钮：设置提醒时间
  - 更多按钮：其他操作（删除、移动等）

预览模式布局：

- 标题栏：同编辑模式
- 预览区域：`TextView` 显示渲染后的 Markdown 内容
- 底部工具栏：编辑按钮（返回编辑模式）、分享按钮

## 6.4 格式选择对话框

对话框设计：

- 标题：“选择导出格式”
- 选项列表：
  - 纯文本（说明：移除所有格式标记）
  - Markdown（说明：保持原始 Markdown 格式）
  - HTML（说明：转换为网页格式）
- 底部按钮：取消、确定

## 7 非功能性设计

### 7.1 性能设计

#### 7.1.1 响应时间优化

- 异步加载：使用 `AsyncQueryHandler` 进行数据库查询，避免阻塞 UI 线程
- 懒加载：列表采用分页加载，每次加载 20-30 条便签
- 缓存策略：Markdown 渲染结果缓存，避免重复渲染
- 索引优化：在常用查询字段上建立索引，提高查询速度

#### 7.1.2 内存优化

- 及时释放：使用完 `Cursor` 后及时关闭，避免内存泄漏
- 对象池：频繁创建的对象使用对象池技术
- 图片压缩：如果支持图片附件，需进行适当压缩
- 弱引用：对大对象使用 `WeakReference`

### 7.2 可靠性设计

#### 7.2.1 异常处理

- 数据库异常：捕获 `SQLiteException`，记录日志并提示用户
- 网络异常：云同步失败时保留本地数据，稍后重试
- 渲染异常：Markdown 渲染失败时显示原文
- 崩溃恢复：记录应用状态，崩溃后恢复现场

#### 7.2.2 数据备份

- 自动保存：编辑过程中定时自动保存
- 云同步：支持与 Google Tasks 同步备份
- 导出功能：支持批量导出便签
- 数据恢复：提供数据恢复功能

## 7.3 安全性设计

### 7.3.1 数据安全

- 权限控制：数据库文件存储在应用私有目录
- 加密传输：云同步使用 HTTPS 协议
- 敏感数据：用户凭证使用 Android KeyStore 存储
- 防注入：使用参数化查询防止 SQL 注入

### 7.3.2 权限管理

- 最小权限原则：仅申请必要权限
- 运行时权限：Android 6.0+ 使用运行时权限请求
- 权限说明：向用户解释权限用途

## 7.4 可维护性设计

### 7.4.1 代码规范

- 命名规范：遵循 Java 命名约定
- 注释规范：类、方法、关键逻辑添加注释
- 代码格式：统一代码格式化风格
- 版本控制：使用 Git 进行版本管理

### 7.4.2 模块化设计

- 低耦合：模块间通过接口交互
- 高内聚：模块内部功能相关性强
- 可替换：核心组件可替换（如 Markdown 引擎）
- 可扩展：预留扩展点，便于添加新功能

## 7.5 可移植性设计

- API 兼容：使用 Support Library 保证向后兼容
- 屏幕适配：支持多种屏幕尺寸和分辨率
- 平台特性：避免使用特定厂商的私有 API
- 国际化：支持多语言（预留设计）

## 8 测试设计

### 8.1 单元测试

#### 8.1.1 测试范围

对以下核心类进行单元测试：

- WorkingNote：测试便签操作逻辑
- MarkdownRenderer：测试 Markdown 渲染功能
- FormatConverter：测试格式转换功能
- NotesDatabaseHelper：测试数据库操作

#### 8.1.2 测试框架

- 使用 JUnit 4 作为测试框架
- 使用 Mockito 进行模拟对象测试
- 使用 Robolectric 进行 Android 组件测试

### 8.2 集成测试

#### 8.2.1 测试场景

- 数据流测试：UI → 业务逻辑 → 数据访问的完整流程
- 模块交互测试：编辑模块与数据模块的交互
- 异步操作测试：数据库异步查询的正确性

## 8.3 系统测试

### 8.3.1 功能测试

测试所有功能需求的实现：

- Markdown 编辑和预览功能
- 多格式导出分享功能
- 原有便签管理功能
- 云同步功能
- 提醒功能

### 8.3.2 性能测试

- 测试界面响应时间
- 测试 Markdown 渲染性能
- 测试大量数据下的性能表现
- 测试内存占用情况

### 8.3.3 兼容性测试

- 在不同 Android 版本上测试 ( 4.0, 5.0, 6.0, 7.0, 8.0, 9.0, 10.0+ )
- 在不同屏幕尺寸设备上测试
- 测试横竖屏切换

## 9 部署设计

### 9.1 构建配置

#### 9.1.1 Gradle 配置

- `minSdkVersion`: 14 (Android 4.0)
- `targetSdkVersion`: 29 (Android 10.0)
- `versionCode`: 递增的整数版本号
- `versionName`: 语义化版本号 ( 如 1.0.0 )

### 9.1.2 依赖库

- Android Support Library v7
- Markwon Markdown 渲染库 v4.x
- Google Play Services (云同步功能)

## 9.2 发布流程

1. 代码审查和测试
2. 更新版本号
3. 生成签名 APK
4. 执行完整测试
5. 上传到应用商店
6. 发布更新说明

## 9.3 版本升级策略

### 9.3.1 数据库升级

在 `NotesDatabaseHelper.onUpgrade()` 中处理数据库升级：

- 保留用户数据
- 添加新表或新字段
- 数据迁移（如需要）
- 更新索引

### 9.3.2 兼容性保证

- 新版本可以读取旧版本数据
- 提供数据导入导出功能
- 平滑的用户体验过渡



## 10 附录

### 10.1 术语表

术语	定义
Activity	Android 四大组件之一，代表一个界面
ContentProvider	Android 四大组件之一，提供数据访问接口
Cursor	Android 中的数据库查询结果集
AsyncQueryHandler	Android 提供的异步数据查询工具类
Markwon	开源的 Android Markdown 渲染库
Spanned	Android 中带格式的字符串类型
Intent	Android 中的消息传递对象
SQLiteOpenHelper	Android 提供的数据库管理助手类

### 10.2 设计变更记录

版本	日期	变更人	变更内容
1.0	2024-XX-XX	潘铮、季凯阳	初始版本