应用开发UI配置框架设计方案

【2019年3月】

内部资料保密勿泄漏

## 版本记录

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **日期** | **版本号** | **修订说明** | **修订人** | **审核人** | **批准人** |
| 2019.03.12 | V1.0 |  | 唐石元 |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

**目录**

[一、 前言概述 1](#_Toc338346695)

[二、 项目背景及需求分析 2](#_Toc338346696)

[2.1 项目背景 2](#_Toc338346697)

[2.2 项目实现需求 3](#_Toc338346698)

[三、 项目设计概要 8](#_Toc338346700)

# 前言概述

本方案的宗旨在于指导项目设计人员，开发人员更好的进行项目的设计开发以及项目的后续维护和改进。本书只讲述主要功能及某些技术关键实现点，详细实现细节本书不一一讲述，如有必要请可查阅具体实现源码。

# 项目背景及需求分析

## 项目背景

为规范公司软件开发流程，提升软件开发效率。

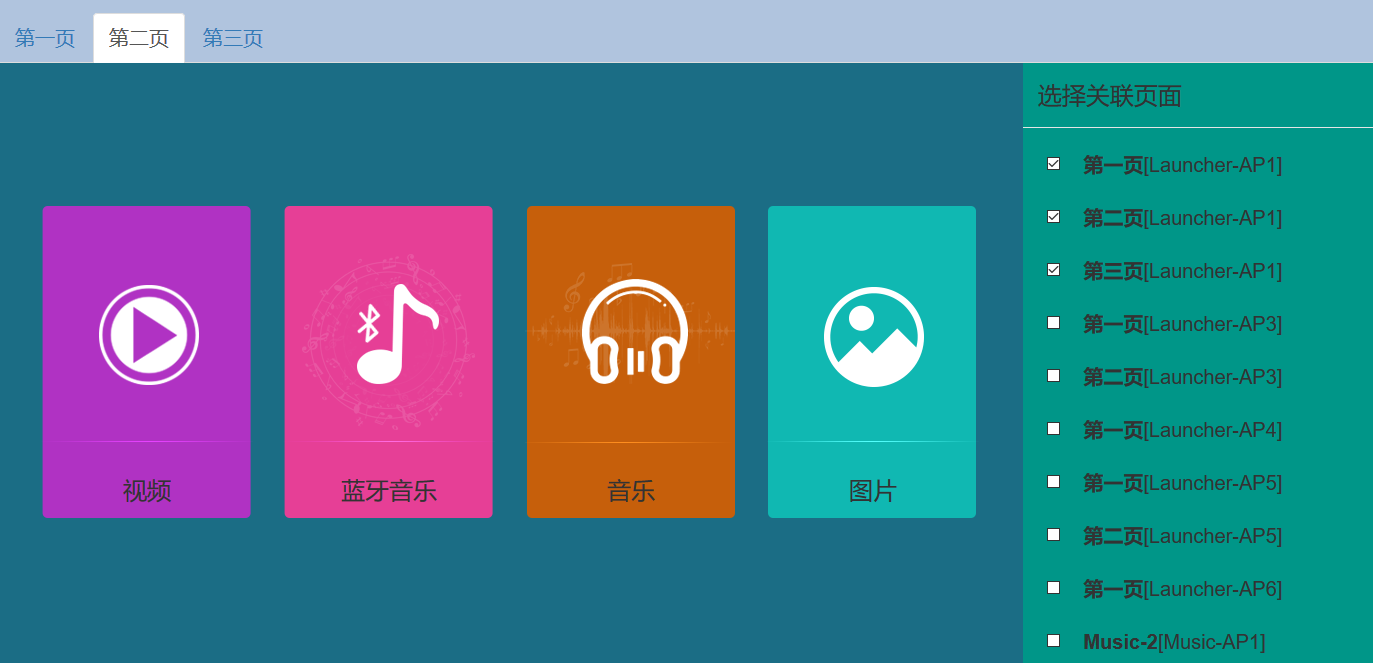
同时也为了解决车载Android平台主应用、蓝牙应用、多媒体应用、收音机等等各种应用需要经常更换UI界面的需求。

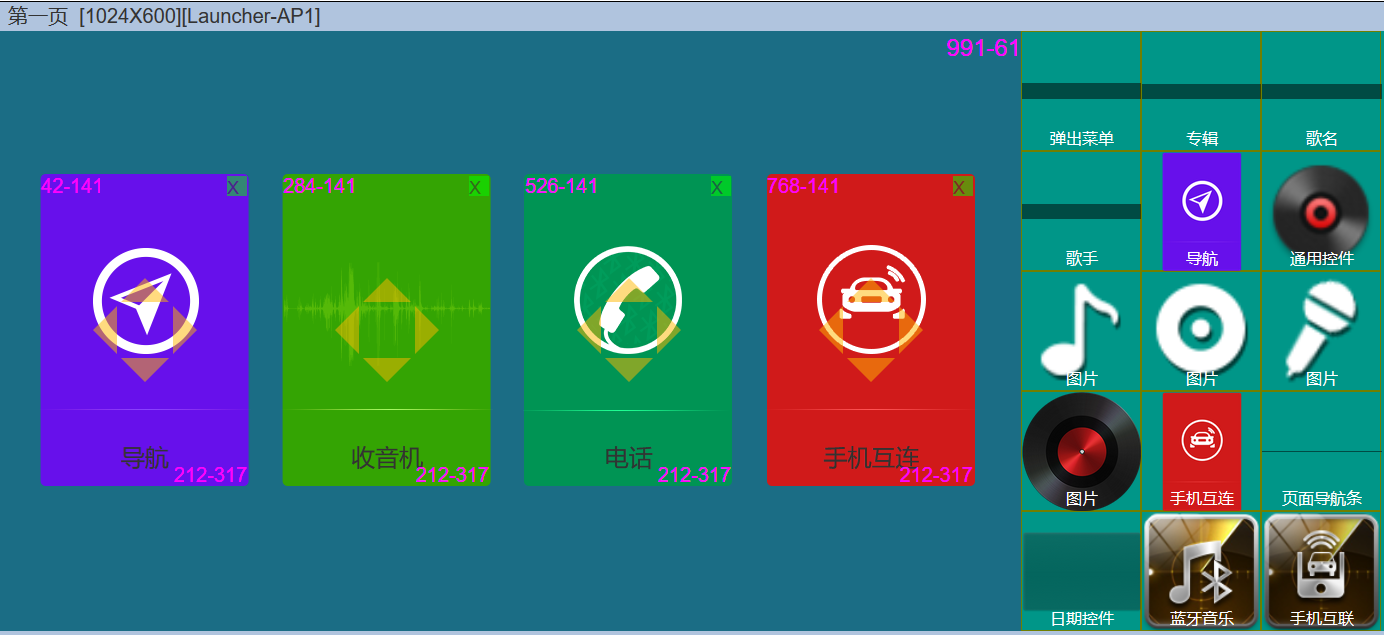
公司软件开发以此项目为框架展开，在将UI和功能分离后，可以将主要精力投入到逻辑功能模块的开发上。在遵循统一框架的基础上，能够提升公司软件开发质量和开发效率，避免重复开发和不规范开发。

## 项目实现需求

### WEB端通过网页拖动图片方式配置UI界面。

如下列Web页面显Launcher配置截图，可随意配置图片文字等内容，拖动位置单位精确到像素以满足任意设计需要。





### Android端根据JSON数据解析生成UI界面。

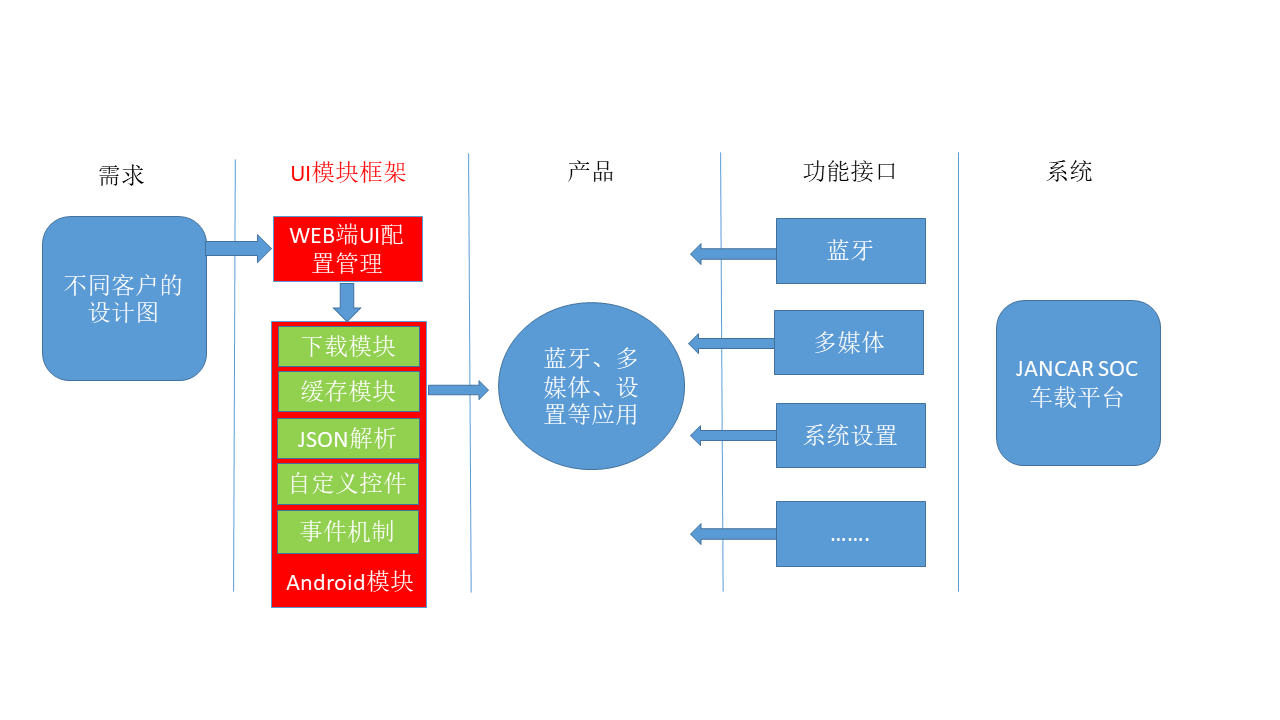
Android端通过访问Web服务端提供的接口，获取配置Json文本以及所关联包含的图片、文字信息内容等资源，最终解析生成应用UI界面。



# 项目设计概要

## 基本说明

### 基本关系简图



### 后台源码目录说明

/application

.../admin //**后台管理入口**

…/api //**网络访问接口**

…/auth //**RABC权限管理模块**

…/flyui /**/UI配置管理模块**

…/home //**公司网站管理后台**

…/index //**公司网站**

…/command.php

…/common.php

…/config.php

**…/database.php //\*数据库配置文件**

…/route.php

…/tags.php

/config //一些配置文件，可自行定义

/downfile //给客服下载的文件放在这里，自定义，可修改

/extend //自定义目录，放了一些可重用的公共函数

/public //网站公共资源,js css img等

/runtime //thinkphp运行需要目录，注意需要写权限

/thinkphp

/uploads //图片上传保存目录

/vendor //thinkphp扩展模块

#阅读源码需要具备PHP,,html,css,js,jQuery等基础知识。

### Android端源码概述

#### 主要类说明

**com.flyzebra.flyui.chache.UpdataVersion.java**

功能如下：

1、从后台接口获取Json数据，并下载Json中指定的图片资源。  
2、将Json数据和图片资源缓存到本地  
3、可从assets中读取并解析数据。

**com.flyzebra.flyui.view.themeview.ThemeView.java //模板**

**com.flyzebra.flyui.view.themeview.PagesViewPager.java**

**com.flyzebra.flyui.view.pageview.SimplePageView.java //单页**

**com.flyzebra.flyui.view.cellview.\*CellView.java //单个控件**

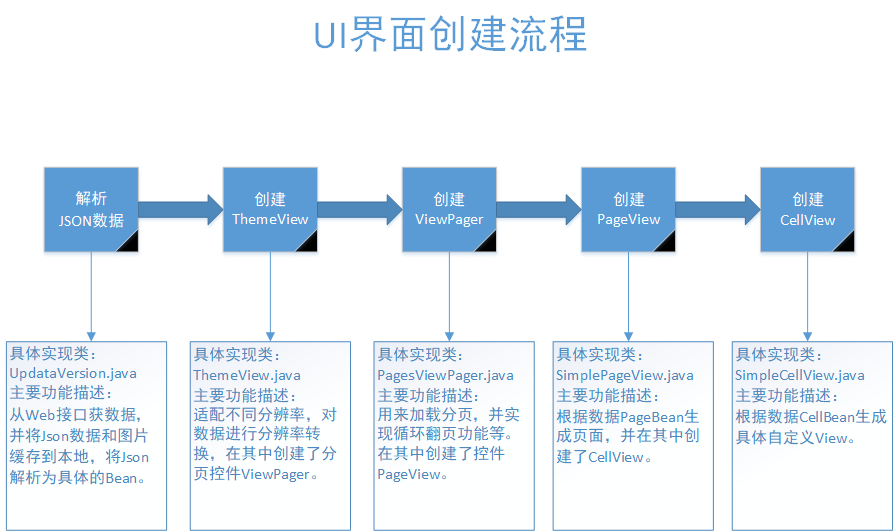
功能如下：

1、析解JSON数据，创建UI界面，ThemeView.java对应模板，PagesViewPager继承自ViewPage用来加载页面(SimplePageView.java), SimplePageView.java对应页面，\*CellView对应各个自定义View。

2、ThemeView.java中的方法matchResolution()会以fitCenter的形式转换适配不同的分辨率。

3、ThemeView.java对应模板，SimplePageView.java对应页面，\*CellView.java对应到具体自定义控件。\*CellView.java的生成加入了工厂方法设计模式，以后的主要工作也在于添加创建\*CellView.java，对功能进行扩展。

#### UI创建流程图



## 后台接口

### 后台接口安全策略

用户访问接口模块为API，目前接口访问安全策略为GET无限制，POST、PUT、DELETE需要登陆后才可操作。具体代码如下(Base.php)：今后可视需求做进一步修改。

**if**(Request::*instance*()->isGet()){  
}**else if**(Request::*instance*()->isPost()){  
 **if** (!session(**'userid'**)) {  
 $this->error(**"没有访问权限!"**);  
 }  
}**else if**(Request::*instance*()->isPut()){  
 **if** (!session(**'userid'**)) {  
 $this->error(**"没有访问权限!"**);  
 }  
}**else if**(Request::*instance*()->isDelete()){  
 **if** (!session(**'userid'**)) {  
 $this->error(**"没有访问权限!"**);  
 }  
}**else**{  
 **if** (!session(**'userid'**)) {  
 $this->error(**"没有访问权限!"**);  
 }  
}

### 后台接口简要说明

**请求地址：/api/app?type=launcher&themeName=Launcher-AP1&version=V1.0.0&acode**

**参数说明：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **参数** | **中文名称** | **参数类型** | **Y/可选** | **参数说明** |
| type | 应用类型 | 字符 | Y | 应用类型 |
| themeName | 模板名称 | 字符 | Y | 模板名称 |
| version | 版本号 | 字符 | Y | 应用版本号 |
| acode | 区域 | 字符 | N | 地理区域标识 |
| userId | 用户标识 | 字符 | N | 用户ID |
| token | 授权码 | 字符 | N | 授权码 |

如下接口地址：

/api/cell

/api/page

/api/celltype

/api/theme

对应cell,page,celltype,theme数据表，遵循Restful

如下接口地址：

/api/pagecell

/api/cellpagecell

/api/themetopcell

结合中间表，返回页面和cell的关联位置关系。

## 网页端设计

### 图片上传功能

图片上传功能将采用图片MD5值做为唯一区分标志，并根据MD5值生成图片URL，避免重复上传，Android客服端根据URL设计图片缓存，提升内存分配的合理性。

Web端将图片上传功能封装为单一控件模块，JS源码为flyimage.js，完成图片上传功能。

如下图：





点击灰色区域会自动跳转到文件选择界面，选择图片即完成图片上传，同时会返回图片地址和图片尺寸等数据 。点击X按钮设计为点击事件，会返回回调方法，具体删除操作需在回调中实现。

代码调用示例如下：

**$**(**'.FlyImageInput'**).flyinput({  
 **ajaxurl**: **"{:url('/api/imagefile')}"**,  
 **autoup**: **true**,  
 **showView**: **true**,  
 **postData**: **false**,  
 **imageUrl**: **""**,  
 **imageWidth**: **""**,  
 **imageHeight**: **""**,  
 **width**: **"100%"**,  
 **height**: **"300px"**}).bind(**"success"**, **function** (e, data) {  
 **$**(**'#imageurl'**).val(data.saveName);  
}).bind(**"del"**, **function** (e, data) {  
 **$**(**'#imageurl'**).val(**""**);  
});

参数说明：

**ajaxurl**: **undefined**,//实现图片上传的接口，具体实现看源码  
**autoup**: **true**,//是否自动上传图片，默认自动上传  
**showView**: **true**,//是否显示预览，不显示为input控件  
**postData**: **false**,//是否生成上传url,图片宽高等input控件  
**imageUrl**: **""**,//默认显示图片  
**imageWidth**: **""**,//默认图片宽度  
**imageHeight**: **""**,//默认图片高度  
**width**: **"100%"**,//控件宽度  
**height**: **"240px"**//控件高度

回调：

bind(**"success"**, **function** (e, data))//上传成功回调

data.savaName//上传成功后返回的图片地址

data.width//上传图片的宽度

data.height//上传图片的高度

bind(**"del"**, **function** (e, data))//删除按钮回调

data.num组件cell标识

data.subnum组件cell.images 标识

两个参数用于指定哪个控件执行了删除操作  
如表达不足请结合源码理解。

### UI拖动布局界面

Web端将拖动布局功能封装为单一控件模块，JS源码位于flyscreen.js，由拖动布置区域，控件选择区域，选项控制区域三个区域组成。

如下所示：左上为拖动布置区域，左下为选项控制区域，右边为控件选择区域。

功能描述：

解析Json数据显示界面，并实现拖拽功能。



拖动区域调用示例代码：

**$**(**'.flyscreen'**).screen({  
 **width**: **"{$item.width}"**,  
 **height**: **"{$item.height}"**,  
 **pageId**: **"{$item.pageId}"**,  
 **url**: **"{:url('/api/pagecell')}"**,  
 **defimgurl**: **"\_\_RES\_\_/flyui/img/default\_cellmenu.png"  
 showdelete**: **true**,  
 **showadjust**: **true**,  
 **moveevent**: **true**});  
参数说明：

**width**: 1024,//控件宽度  
**height**: 600,//控件高度  
**pageId**: **undefined**,//页面ID  
**url**: **undefined**,//中间表接口地址  
**defimgurl**: **""**,//默认显示图片  
**showdelete**: **true**,//是否显示DEL标签  
**showadjust**: **true**,//是否显示位置调整标签  
**moveevent**: **true**//是否可拖动

方法说明

**$**(**'.flyscreen'**).screen(**'refresh'**, **pageId**);//刷新界面

**$**(**'.flyscreen'**).screen(**"showdelete"**, **true**);//显示/隐藏DEL标签  
**$**(**'.flyscreen'**).screen(**"showadjust"**, **true**);//显示/隐藏位置调整标签  
  
控件选择区域调用示例代码：

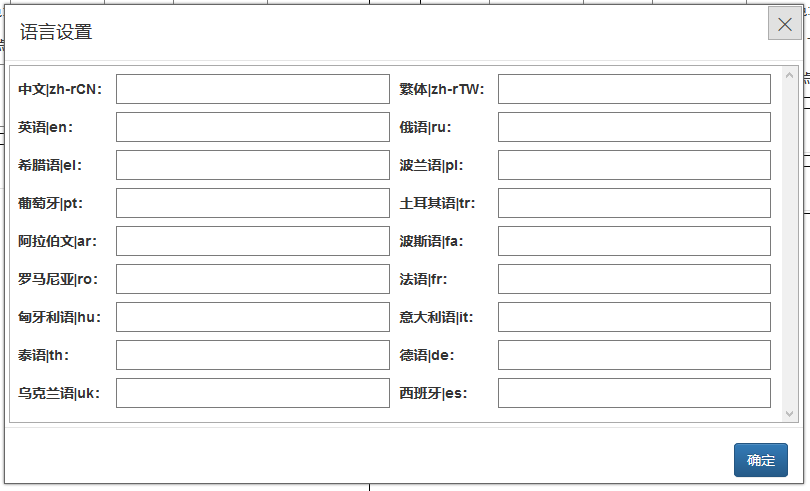
**$**(**'.flymenu'**).menu({  
 **width**: 380,  
 **height**: **"{$item.height}"**,  
 **url**: **"{:url('/api/cell')}"**,  
 **defimgurl**: **"\_\_RES\_\_/flyui/img/default\_cellmenu.png"**,  
 **childw**:120,  
 **childh**:120  
});

参数说明：

**width**: 380,//控件宽度  
**height**: 600,//控件高度  
**url**: **undefined**,//获取Cell的接口地址  
**defimgurl**: **"./default\_cellmenu.png"**,//默认显示图片  
**childw**: 120,//子项Cell显示区域宽度  
**childh**: 120子项Cell显示区域高度

//需要扩展功能：搜索查找功能

### 多国语言及事件添加



如以上图片所示，此控件用来设置多国语言和设置事件等，将对应ID做为Key值转换成Json字符串。

调用示例如下：

**$**(**'#languageDialog'**).flyjsonedit({  
 **title**: **"语言设置"**,  
 **width**: **"800px"**,  
 **height**: **"480px"**,  
 **inputIds**: [**"en"**, **"zh\_rCN"**, **"zh\_rTW"**, **"ru"**, **"el"**, **"pl"**, **"tr"**, **"ar"**, **"fa"**, **"ro"**, **"fr"**, **"hu"**, **"it"**, **"th"**, **"de"**, **"uk"**, **"es"**, **"pt"**]  
}).bind(**"confirm"**, **function** (e, data) {  
 **if** (**"{$id}"** > 0) {  
 *postForm*(**'{$id}'**, **true**);  
 }  
});

其调用需要配合HTML中的代码，InputIDs对应Html中的input 控件id

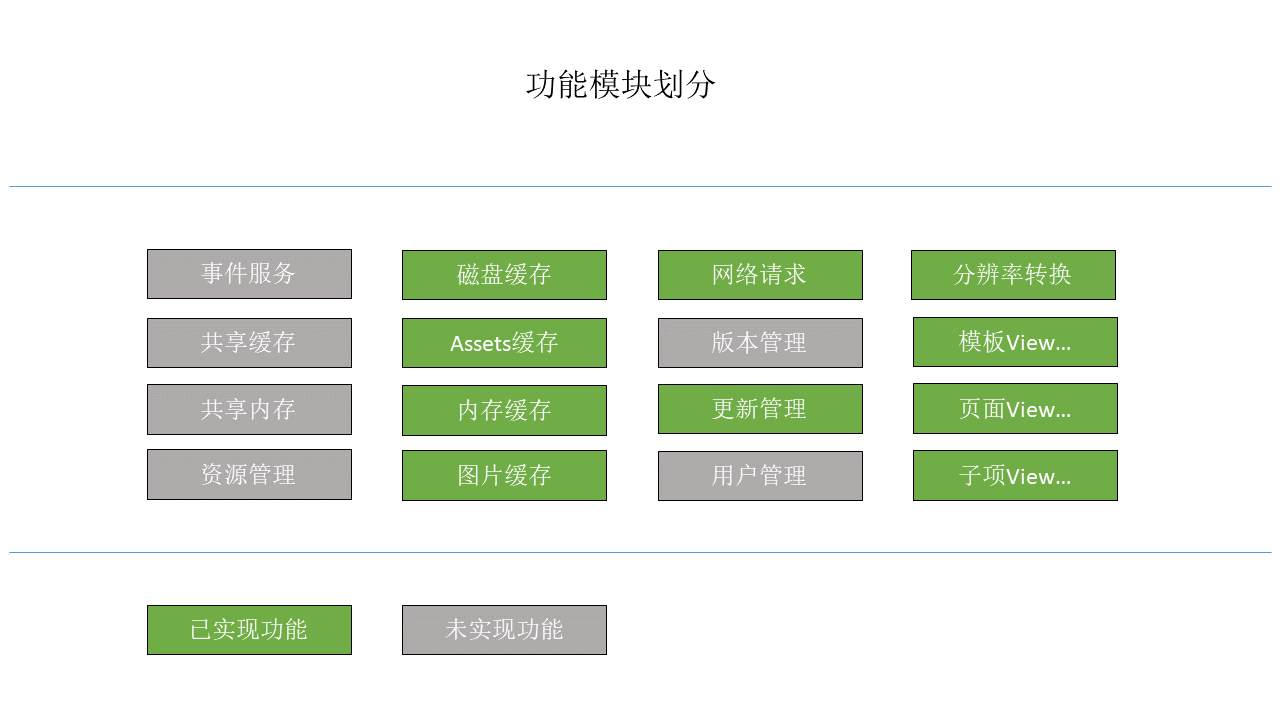
*<!--语言编辑-->*<**div id="languageDialog" class="FlyJsonEdit"**>  
 <**div style="margin**:4**px**;**padding**:4**px**;**width**:**calc**(100% - 8**px**);;**height**: 358**px**;**border**:**solid**;**border-width**:1**px**;**border-color**:**#AAAAAA**;;**overflow-y**:**scroll**;**"**>  
 <**div style="margin**:4**px**;**height**:30**px**;**width**:**calc**(50% - 8**px**);**float**: **left"**>  
 <**label style="width**: 94**px**;**"**>中文|zh-rCN：</**label**>  
 <**input id="zh\_rCN" style="width**:**calc**(100% - 100**px**);**height**:30**px**;**" type="text"**>  
 </**div**>  
 …此处省略其它语言…  
 </**div**>  
</**div**>

### 其它功能

其它细节不一一概述，具体实现细节结合源码来理解。

## Android端设计

### 功能模块划分



### 模块功能简述

**事件服务**：做为一个独立的进程服务，提供消息转发，消息数据缓存，结合系统需要开发。

**共享缓存**：所有应用需要的资源，文字和图片，通过该模块实现统一管理，避免重复占用存储空间。

**共享内存**：设计理念为，当需要从磁盘文件获取数据时，可以不需要每次都从磁盘获取，可以直接从内存获取，其中对于可能存在需要修改的资源进行内存COPY。

**资源管理**：对于某些资源的控制，例如声音输出，声音录入等设备的使用，我们需要把这些资源管理起来，分配使用队列，记录使用状态。

**磁盘缓存**：基于开源项目DiskLruCache实现，可设置最大缓存，超过最大缓存后，会依据LRU算法删除使用次数最少的数据内容。

实现代码参考DiskCache.java。

**Assets缓存**：此模块的提供的主要功能为可将磁盘缓存数据打包入原始APK。开发APK时，我们可以把磁盘数据复制到assets中，当无法联网获取数据或磁盘中没有缓存数据时，可以加载assets中的数据显示UI。目前只做了简单读取实现，代码位于UpdataVersion.java中，文本通过流读取，图片则转换url依赖Glide加载。今后进一步扩展可逐步将其提取为独立的模块，对功能进行扩展。

**内存缓存**：使用LruCache实现，扩展实现请参考BitmapMemoryCache.java.

**图片缓存**：图片缓存实现了磁盘和内存双缓存。参考BitmapDiskCache.java，BitmapMemoryCache.java，DoubleBitmapCache.java。

**网络请求**：对开源项目okhttp进行了封装，实现源码FlyOkHttp.java。此外还实现一个图片下载功能，源码文件HttpDownFile.java。

**版本管理**：根据系统软硬件版本，使用区域，使用用户群体，将软件版本跟这些关联起来，并提供所有软硬件的当前版本信息。

**更新管理**：对软件UI界面及某些功能进行升级更新。目前只实现了UI更新。参考文件UpdataVersion.java。

完整的设计为，根据管理模块提供的信息，从对应的网络请求接口获取需要更新的内容，下载内容进行校验后再对软件进行更新，同时删除不必要的老版本资源。

**用户管理**：（如有必要，如果没有自己的软件生态建设想法没有必要）用户注册登陆，用户行为采集，用户界面和功能定制，用户数据远程保存等等。可以参考手机的用户功能。

**分辨率转换:**目前只于themeView.java中实现了UI fitCenter转换，建议此功能今后扩展逐步提取为独立模块，处理各种分辨率屏幕适配的问题。  
**以下为UI自定义View：**

**模板View…**:对应于一个项目的UI界面。

**页面View…**:对应用于一页UI界面。

**子项View**…:对于一页中的一个自定义View。

### 其它功能