**桌面配置**

**编 写 人：钟其鸿**

版本历史

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 版本号 | 作者 | 操作 | 日期 | 说明 |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

目录

[1 概述 3](#_Toc404672931)

[2 类图，框架 4](#_Toc404672932)

[3 代码分析 4](#_Toc404672933)

### 1 概述

这个文档主要说明default\_worskpace.xml文件使用方式。由于这文件涉及到了Workspace和Hotseat两部分的配置，所以这里所说的“桌面”指的是这两个模块。

### 2 类图，框架

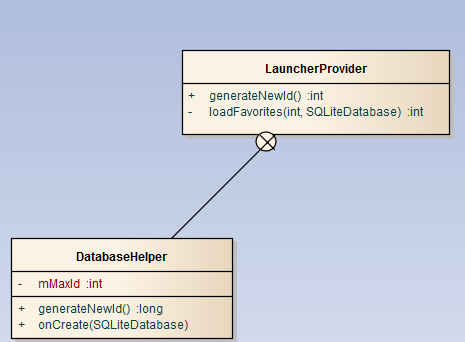
文件结构路径：

配置文件：default\_workspace.xml

解析文件：LauncherProvider.java

属性文件：attrs.xml

数据库：favorites.db



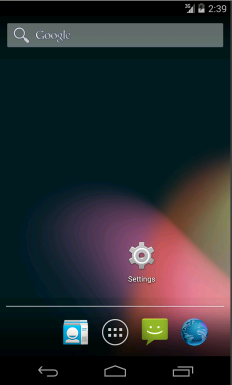
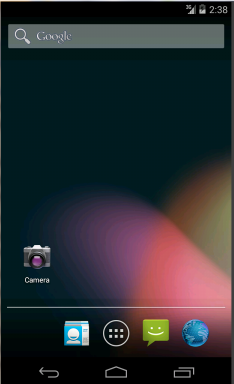
### 3 代码分析

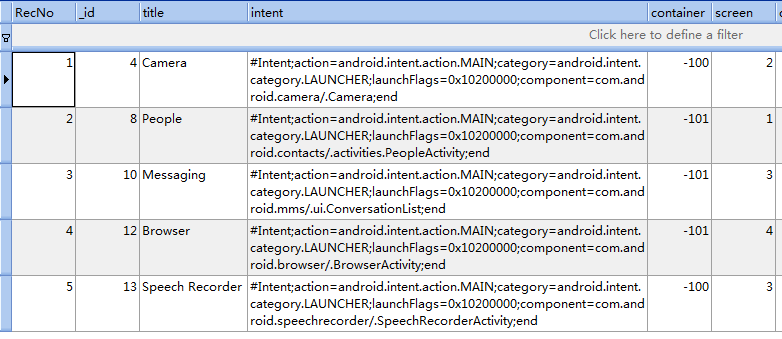
通过上面的类图可以看出，默认配置信息保存在了default\_workspace.xml和数据favorites.db文件中。default\_workspace.xml文件只在安装软件后第一次运行的时候或者清除数据后才会被读取使用。favorites.db文件在第一次运行或者清除数据后也会被置空。

这两个文件是什么关系呢？

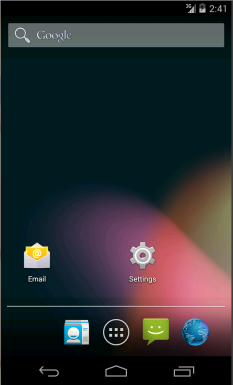
实际上，桌面显示的内容全部都是根据favorites.db数据库中读取的。如果第一次安装打开，Launcher2首先读取的是default\_workspace.xml的内容然后显示在桌面，因为数据库这时候是空的，这种方式也能保证加载显示的速度。当桌面初始化完成后，或者用户对桌面的元素作了改变Launcher2会修改数据库中的记录。后续所有的显示数据都来自数据库，也就是说default\_workspace.xml只会被使用一次，数据库则是数据永久保存和读取的地方。

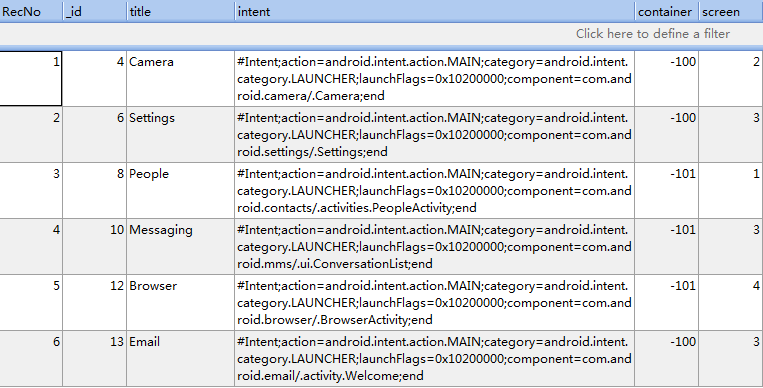
下面的图片是Launcher在nexus上第一次安装运行的效果和数据库的内容展示：





在桌面添加一个Email快捷键后，桌面显示效果和数据库改变后的内容：



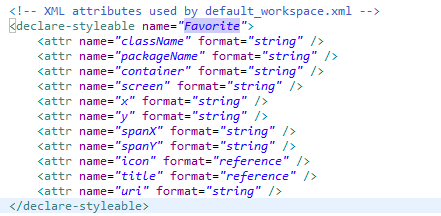


default\_workspace.xml 部分内容展示：



这里定义了一个快捷键和一个插件，他们都显示在桌面的第二个页面。由于手机Rom版本的问题，上面的手机屏幕没有显示出时钟这个插件。如果没有指定container，表示默认加载workspace上。为了规范化，后面大家必须把所有的属性都填充好。

default\_workspace.xml 的属性规范是在属性文件attrs.xml文件中定义的：



其中，

ClassName : 快捷键（插件）的类名

PackageName：快捷键（插件）的包名

Container: 所属容器的id，这个id会在后面介绍

Screen： 所在的屏幕

X：x方向的起始点。注意，这个x并不是具体的坐标，而是宫格的起始单元。

Y：y方向的其实点

SpanX: 占用x方向的宫格数目

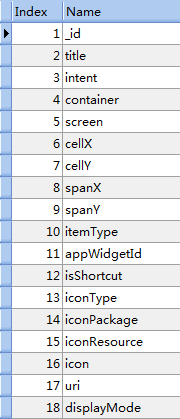
SpanY：占用y方向的宫格数目

Icon： 图片

Title：名称

Uri：外部访问地址

Favorities.db数据库中的属性：



数据库中的属性要比xml文件中定义的多。

itemType: 桌面元素的类型。比如是快捷键，时钟，文件夹，搜索条

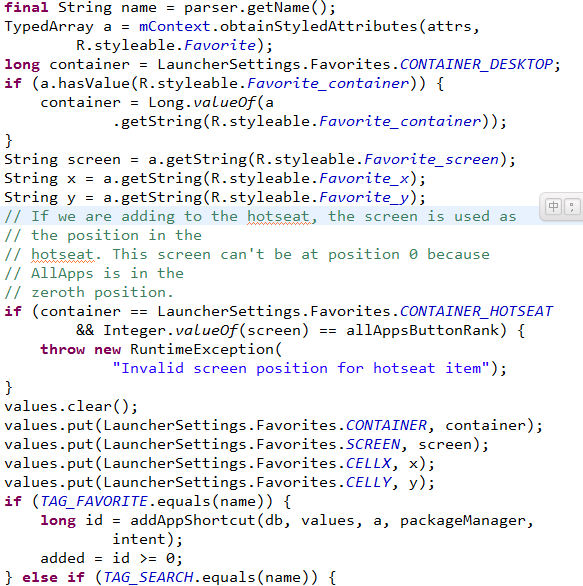
appWidgetId：widget的id

isShortcut：是否是快捷键

iconType:图片的类型。Bitmap，png，jpg…

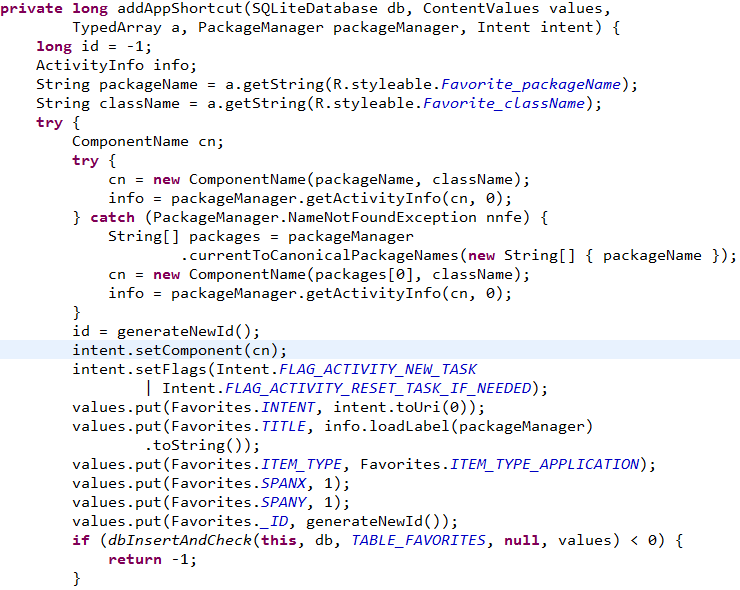
LauncherProvider.java:

private int loadFavorites(SQLiteDatabase db, int workspaceResourceId)：



**那段英文注释很重要：如果父容器是hotseat，那么这里的screen表示的不是第几页，而是position。如果配置的位置和抽屉入口的位置重复了，那么会抛出异常。**

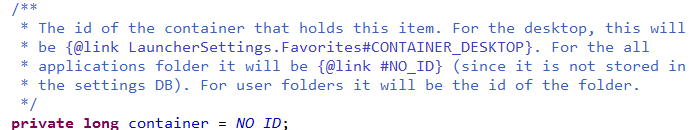
这里以添加一个普通应用入口为例子，说明Launcher是怎么创建这个应用信息的，其他类似文件夹，插件等原理类似不重复说明了。



这个函数的最后是把这个应用信息写到了数据库中。至于Launcher是如何利用这个写入的ItemInfo生成一个View然后添加到桌面的请看《Launcher启动流程详解》.docx。

附录：数据库中的container属性是怎么生成的？

Launcher对container的定义是：

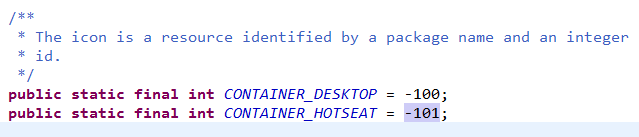


这个属性是指保存这个元素（item）的容器的id。如果这个元素保存在desktop，那么这id会被赋值成CONTAINER\_DESKTOP, 如果这元素是抽屉中的内容，那么id会被赋值成NO\_ID(因为它没有被保存在数据库中)。对于文件夹，这个id会被赋值成文件所在的“id”。

也就是说每个容器都有一个自身的编号。比如workspace的编号是：CONTAINER\_DESKTOP，

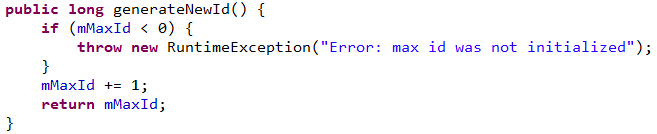
Hotseat的编号是CONTAINER\_HOTSEAT。这两id是在LauncherSettings.java中定义的：

LauncherSettings.java



那么文件夹的编号是怎么生成的呢？

LauncherProvider.java有一个内部类DatabaseHelper.java。这个内部类有一个函数叫做



每次读取到一个配置，都会调用这个函数获得一个元素的id，然后把这个id保存到数据库中的字段”\_id”中。其中mMaxId是在构造函数中初始化的，初始化的方式是读取数据库中所有记录中”\_id”字段的最大的那个值。

