**Android 8.0 Settings流程分析与变动**

2018年03月12日 18:38:18 [angel鹿](https://me.csdn.net/qq_31012033) 阅读数：5917

 版权声明：本文为博主原创文章，未经博主允许不得转载。 https://blog.csdn.net/qq\_31012033/article/details/79522813

开！ 场!  白！ 好！ 难！ 写！

一，相比Android Settings 7.0

    如下图，在7.0的基础上，去掉了7.0新加的侧滑菜单（可能是觉得有点鸡肋吧）。多加了一级页面，把原来类别标题变成的第一级菜单的子项。在代码架构也稍加变动，并引入架构组件之LifeCycle（生命周期感知，本文不作介绍）。



二，第一级菜单的加载

浏览源码，大多数我们从程序的AndroidManifest.xml入手，这次也不列外。

packages\apps\Settings\AndroidManifest.xml:

1. <activity-alias android:name="Settings"
2. android:taskAffinity="com.android.settings"
3. android:label="@string/settings\_label\_launcher"
4. android:launchMode="singleTask"
5. android:targetActivity="Settings">
6. <intent-filter>
7. <action android:name="android.intent.action.MAIN" />
8. <category android:name="android.intent.category.DEFAULT" />
9. <category android:name="android.intent.category.LAUNCHER" />
10. </intent-filter>
11. <meta-data android:name="android.app.shortcuts" android:resource="@xml/shortcuts"/>
12. </activity-alias>

找到<category android:name="android.intent.category.LAUNCHER" />所属的类，Settings.java。但打开Settings.java类看，除了大量静态类继承SettingsActivity，就没什么东西了。那再去父类SettingsActivity.java找找。

packages\apps\Settings\src\com\android\settings\SettingsActivity.java:

首先当然是onCreate(）->

1. @Override
2. protected void onCreate(Bundle savedState) {
3. super.onCreate(savedState);
4. long startTime = System.currentTimeMillis();
5. *//工厂类实现方法com.android.settings.overlay.FeatureFactoryImpl.java*
6. final FeatureFactory factory = FeatureFactory.getFactory(this);
7. *//获取菜单信息的工厂类，实现类为[Dashboard](https://www.baidu.com/s?wd=Dashboard&tn=24004469_oem_dg&rsv_dl=gh_pl_sl_csd" \t "_blank)FeatureProviderImpl.java*
8. mDashboardFeatureProvider = factory.getDashboardFeatureProvider(this);
9. mMetricsFeatureProvider = factory.getMetricsFeatureProvider();
10. *// 从intent信息中获取<meta-data/>标签名为"com.android.settings.FRAGMENT\_CLASS"的值（下文用于加载Fragment的类名）*
11. getMetaData();
12. ... ...
13. *//获取上面getMetaData()得到的类名*
14. final String initialFragmentName = intent.getStringExtra(EXTRA\_SHOW\_FRAGMENT);
15. *//是否为快捷进入方式（如从其它的应用进入Settings的某个设置项）*
16. mIsShortcut = isShortCutIntent(intent) || isLikeShortCutIntent(intent) ||
17. intent.getBooleanExtra(EXTRA\_SHOW\_FRAGMENT\_AS\_SHORTCUT, false);
18. ... ...
19. *//当前类是否为Settings.class，即进入方式为点击launcher上的图标进入的主设置界面*
20. mIsShowingDashboard = className.equals(Settings.class.getName());
21. ... ...
22. setContentView(mIsShowingDashboard ?
23. R.layout.settings\_main\_dashboard : R.layout.settings\_main\_prefs);
24. mContent = findViewById(R.id.main\_content);
25. getFragmentManager().addOnBackStackChangedListener(this);
26. if (savedState != null) {
27. ... ...
28. } else {
29. *//加载布局*
30. launchSettingFragment(initialFragmentName, isSubSettings, intent);
31. }
33. ... ...
34. }

launchSettingFragment()->

1. @VisibleForTesting
2. void launchSettingFragment(String initialFragmentName, boolean isSubSettings, Intent intent) {
3. if (!mIsShowingDashboard && initialFragmentName != null) {
4. ... ...
5. switchToFragment(initialFragmentName, initialArguments, true, false,
6. mInitialTitleResId, mInitialTitle, false);
7. } else {
8. *// Show search icon as up affordance if we are displaying the main Dashboard*
9. mDisplayHomeAsUpEnabled = true;
10. mInitialTitleResId = R.string.dashboard\_title;
11. *//进入主页走的这里，替换目标Fragment*
12. switchToFragment(DashboardSummary.class.getName(), null */\* args \*/*, false, false,
13. mInitialTitleResId, mInitialTitle, false);
14. }
15. }

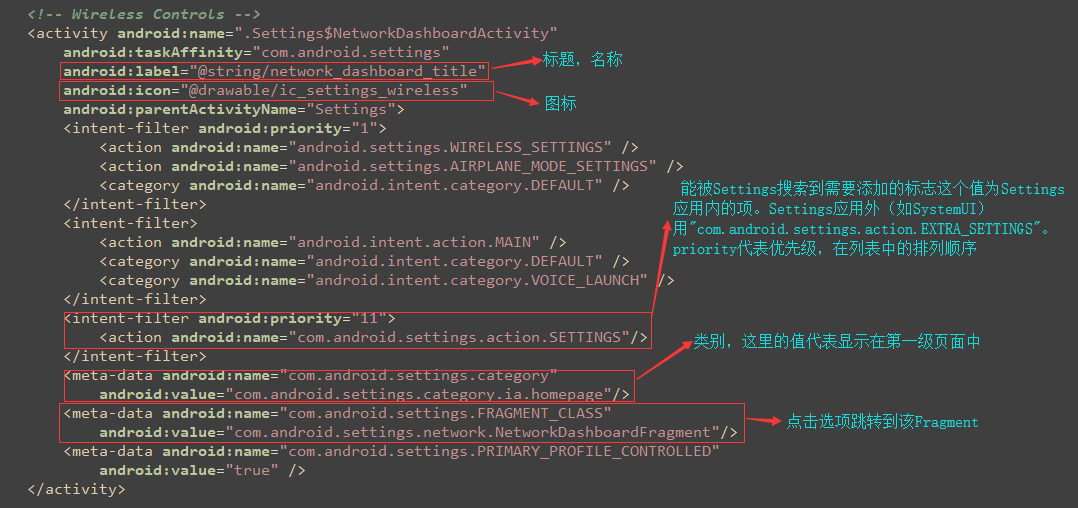
继续，我们看看DashboardSummary.java，对于它我们主要是想知道它的数据加载，它是怎么加载自己的子项的。

packages\apps\Settings\src\com\android\settings\dashboard\DashboardSummary.java:

对子项的数据获取在updateCategoryAndSuggestion(）中得到实现。

1. @VisibleForTesting
2. void updateCategoryAndSuggestion(List<Tile> suggestions) {
3. final Activity activity = getActivity();
4. if (activity == null) {
5. return;
6. }
7. */\*根据"com.android.settings.category"的值查询子项数据，这里的值为"com.android.settings.category.ia.homepage"。*
8. *具体获取办法追踪到frameworks\base\packages\SettingsLib\src\com\android\settingslib\drawer\TileUtils.java中。*
9. *通过PackageManager查询所有在AndroidManifest.xml中定义<meta-data/>中含有该值的类。注意：会过滤掉非系统级应用的数据！*
10. *有兴趣的自行研究，这里不深究。\*/*
11. final DashboardCategory category = mDashboardFeatureProvider.getTilesForCategory(
12. CategoryKey.CATEGORY\_HOMEPAGE);
13. if (category == null) {
14. return;
15. }
16. mSummaryLoader.updateSummaryToCache(category);
17. if (suggestions != null) {
18. mAdapter.setCategoriesAndSuggestions(category, suggestions);
19. } else {
20. *//数据的绑定在[适配器](https://www.baidu.com/s?wd=%E9%80%82%E9%85%8D%E5%99%A8&tn=24004469_oem_dg&rsv_dl=gh_pl_sl_csd" \t "_blank)中,->packages\apps\Settings\src\com\android\settings\dashboard\DashboardAdapter.java*
21. mAdapter.setCategory(category);
22. }
23. }

对于第一级菜单的加载。在AndroidManifest.xml中的配置如下列图：



三，第二级菜单的加载

    以上我们知道第一级菜单是完全动态的加载，但二级菜单则是动态加载和静态xml布局文件，就拿“系统”这项为例。

packages\apps\Settings\AndroidManifest.xml:

1. <activity android:name=".Settings$SystemDashboardActivity"
2. android:label="@string/header\_category\_system"
3. android:icon="@drawable/ic\_settings\_about">
4. <intent-filter android:priority="-1">
5. <action android:name="com.android.settings.action.SETTINGS"/>
6. </intent-filter>
7. <meta-data android:name="com.android.settings.category"
8. android:value="com.android.settings.category.ia.homepage"/>
9. <meta-data android:name="com.android.settings.FRAGMENT\_CLASS"
10. android:value="com.android.settings.system.SystemDashboardFragment"/>
11. <meta-data android:name="com.android.settings.summary"
12. android:resource="@string/system\_dashboard\_summary"/>
13. </activity>

SystemDashboardFragment.java继承DashboardFragment.java。我们主要观察这个类。

packages\apps\Settings\src\com\android\settings\dashboard\DashboardFragment.java:

**1,静态加载部分：**

静态加载部分的实现方法为displayResourceTiles()->

1. */\*\**
2. *\* Displays resource based tiles.*
3. *\*/*
4. private void displayResourceTiles() {
5. *//获取xml布局文件的id（DashboardFragment.java实现该方法）*
6. final int resId = getPreferenceScreenResId();
7. if (resId <= 0) {
8. return;
9. }
10. addPreferencesFromResource(resId);
11. final PreferenceScreen screen = getPreferenceScreen();
12. */\*\* 实现布局文件中的子项控件的业务逻辑*
13. *\* DashboardFragment.java的子类实现getPreferenceControllers（）方法，该方法加载继承于AbstractPreferenceController.java的实现业务逻辑类*
14. *\*/*
15. Collection<AbstractPreferenceController> controllers = mPreferenceControllers.values();
16. for (AbstractPreferenceController controller : controllers) {
17. controller.displayPreference(screen);
18. }
19. }

**2，动态加载部分：**

动态加载部分的实现方法refreshDashboardTiles(）->

1. */\*\**
2. *\* Refresh preference items backed by DashboardCategory.*
3. *\*/*
4. @VisibleForTesting(otherwise = VisibleForTesting.PRIVATE)
5. void refreshDashboardTiles(final String TAG) {
6. final PreferenceScreen screen = getPreferenceScreen();
7. */\* 获取子项*
8. *\* getCategoryKey()从DashboardFragmentRegistry.PARENT\_TO\_CATEGORY\_KEY\_MAP中获取Category值。*
9. *\* 该值通过类名获取*
10. *\* 存：PARENT\_TO\_CATEGORY\_KEY\_MAP.put(SystemDashboardFragment.class.getName(), CategoryKey.CATEGORY\_SYSTEM);*
11. *\* CATEGORY\_SYSTEM = "com.android.settings.category.ia.system";*
12. *\*/*
13. final DashboardCategory category =
14. mDashboardFeatureProvider.getTilesForCategory(getCategoryKey());
15. ... ...
16. *// Install dashboard tiles.*
17. for (Tile tile : tiles) {
18. ... ...
19. if (mDashboardTilePrefKeys.contains(key)) {
20. ... ...
21. } else {
22. *// Don't have this key, add it.*
23. final Preference pref = new Preference(getPrefContext());
24. */\*在这里进行绑定，加载*
25. *\*packages\apps\Settings\src\com\android\settings\dashboard\DashboardFeatureProviderImpl.java->bindPreferenceToTile()*
26. *\*/*
27. mDashboardFeatureProvider.bindPreferenceToTile(getActivity(), getMetricsCategory(),
28. pref, tile, key, mPlaceholderPreferenceController.getOrder());
29. mProgressiveDisclosureMixin.addPreference(screen, pref);
30. mDashboardTilePrefKeys.add(key);
31. }
32. remove.remove(key);
33. }
34. *// Finally remove tiles that are gone.*
35. for (String key : remove) {
36. mDashboardTilePrefKeys.remove(key);
37. mProgressiveDisclosureMixin.removePreference(screen, key);
38. }
39. mSummaryLoader.setListening(true);
40. }

该文的Settings加载流程就差不多到这里了。

四，顺便说说

下拉菜单栏时长按设置图标进入设置，在系统项里面会多一个[《系统界面调节工具》](https://www.baidu.com/s?wd=%E3%80%8A%E7%B3%BB%E7%BB%9F%E7%95%8C%E9%9D%A2%E8%B0%83%E8%8A%82%E5%B7%A5%E5%85%B7%E3%80%8B&tn=24004469_oem_dg&rsv_dl=gh_pl_sl_csd" \t "_blank)。那么这是怎么显示和隐藏的了？

frameworks\base\packages\SystemUI\src\com\android\systemui\tuner\TunerService.java

->setTunerEnabled():

1. public static final void setTunerEnabled(Context context, boolean enabled) {
2. *//隐藏应用图标，隐藏某个组件启动也可以使用该方法*
3. userContext(context).getPackageManager().setComponentEnabledSetting(
4. new ComponentName(context, TunerActivity.class),
5. enabled ? PackageManager.COMPONENT\_ENABLED\_STATE\_ENABLED
6. : PackageManager.COMPONENT\_ENABLED\_STATE\_DISABLED,
7. PackageManager.DONT\_KILL\_APP);
8. }