# TODO

RecyclerView总结，如何分析

# 基本介绍

在线下载路径

/Users/chanson.cc/Library/Android/sdk/extras/android/m2repository/com/android/support

support-v4-24.1.1-sources.jar是包含代码的jar， 使用JD-GUI可以打开查看代码。 从24.1.1 之后的就都是aar 的库了， 无法再通过这种方式直接看源码了。 但是仍然可以解压aar， 查看里面的class.jar 来查看。

AS在线引用

compile **'com.android.support:appcompat-v7:25.4.0'**compile **'com.android.support:support-v4:25.4.0'**compile **'com.android.support:support-v13:25.4.0'**compile **'com.android.support:design:25.4.0'**

compile **'com.android.support:palette-v7:25.4.0'**compile **'com.android.support:recyclerview-v7:25.4.0'**

通过

Frameworks.support

比如系统设置

compile **'com.android.support:appcompat-v7:25.4.0'**compile **'com.android.support:support-v4:25.4.0'**compile **'com.android.support:support-v13:25.4.0'**compile **'com.android.support:design:25.4.0'**compile **'com.android.support:recyclerview-v7:25.4.0'**compile **'com.android.support:palette-v7:25.4.0'**

这里有个误区

错

*<!-- <style name="Rm500.Preference.DropdownPreferenceStyle" parent="@android:style/Preference.DropDown.Material">-->*

对

<**style name="Rm500.Preference.DropdownPreferenceStyle" parent="style/Preference.DropDown.Material"**>  
 <**item name="android:selectableItemBackground"**>@color/black\_25</**item**>  
</**style**>

分析：本质上**support的并不是系统模块，因为编译选项是**

google提供了Android Support Library package 系列的包来保证来高版本sdk开发的向下兼容性，即我们用4.x开发时，在1.6等版本上，可以使用高版本的有些特性，如fragement,ViewPager等，下面，简单说明下这几个版本间的区别：

Android Support v4: 这个包是为了照顾1.6及更高版本而设计的，这个包是使用最广泛的，eclipse新建工程时，都默认带有了。

Android Support v7: 这个包是为了考虑照顾2.1及以上版本而设计的，但不包含更低，故如果不考虑1.6,我们可以采用再加上这个包，另外注意，v7是要依赖v4这个包的，即，两个得同时被包含。Android Support v13 :这个包的设计是为了android 3.2及更高版本的，一般我们都不常用，平板开发中能用到。

Android Support v14 :这个包的设计依赖哪个版本没找到，应该是android 5.0，一般我们都不常用，可穿戴设备开发中能用到。

Android Support v17 :这个包的设计依赖android 5.0及更高版本，一般我们都不常用，电视开发中能用到。

---------------------

作者：Aquarius\_Seven

来源：CSDN

原文：https://blog.csdn.net/Aquarius\_Seven/article/details/51160299

版权声明：本文为博主原创文章，转载请附上博文链接！

## QA？

Fgh

1, Android Support V4, V7, V13是什么？

本质上就是三个java library。

2, 为什么要有support库？

如果在低版本Android平台上开发一个应用程序，而应用程序又想使用高版本才拥有的功能，就需要使用Support库。

3, 三个Support 库的区别和作用是什么？

Android Support v4 是最早（2011年4月份）实现的库。用在Android1.6 (API lever 4)或者更高版本之上。它包含了相对V4, V13大的多的功能。

例如：Fragment，NotificationCompat，LoadBroadcastManager，ViewPager，PageTabAtrip，Loader，FileProvider 等。

详细API 参考 http://developer.android.com/reference/android/support/v4/app/package-summary.html

Android Support v7: 这个包是为了考虑Android2.1(API level 7) 及以上版本而设计的，但是v7是要依赖v4这个包的，也就是如果要使用，两个包得同时 被引用。

v7支持了Action Bar。

Android Support v13:这个包的设计是为了android 3.2及更高版本的，一般我们都不常用，平板开发中能用到。

4, 如何使用？

首先要保证Android Support Library 被安装

---------------------

作者：hh2000

来源：CSDN

原文：https://blog.csdn.net/hh2000/article/details/39718623

版权声明：本文为博主原创文章，转载请附上博文链接！

# RecyclerView

Sdfg

focused====android.support.v7.widget.RecyclerView{ffe71e2 VFED.V... .F...... 0,0-1920,900 #7f11020a app:id/list}

mView====DecorView@c97202a[Settings$InputMethodAndLanguageSettingsActivity]

recoverFocusFromState

---------------------

View viewToFocus = null;

if (focusTarget == null || mChildHelper.isHidden(focusTarget.itemView)

|| !focusTarget.itemView.hasFocusable()) {

if (mChildHelper.getChildCount() > 0) {

// At this point, RV has focus and either of these conditions are true:

// 1. There's no previously focused item either because RV received focused before

// layout, or the previously focused item was removed, or RV doesn't have stable IDs

// 2. Previous focus child is hidden, or 3. Previous focused child is no longer

// focusable. In either of these cases, we make sure that RV still passes down the

// focus to one of its focusable children using a best-effort algorithm.

viewToFocus = findNextViewToFocus();///BUGFIX

}

} else {

// looks like the focused item has been replaced with another view that represents the

// same item in the adapter. Request focus on that.

viewToFocus = focusTarget.itemView;

}

if (viewToFocus != null) {

if (mState.mFocusedSubChildId != NO\_ID) {

View child = viewToFocus.findViewById(mState.mFocusedSubChildId);

if (child != null && child.isFocusable()) {

viewToFocus = child;

}

}

viewToFocus.requestFocus();

}

-------------------------------------

@Nullable

private View findNextViewToFocus() {

int startFocusSearchIndex = mState.mFocusedItemPosition != -1 ? mState.mFocusedItemPosition

: 0;

ViewHolder nextFocus;

final int itemCount = mState.getItemCount();

for (int i = startFocusSearchIndex; i < itemCount; i++) {

nextFocus = findViewHolderForAdapterPosition(i);

if (nextFocus == null) {

break;

}

if (nextFocus.itemView.hasFocusable()) {

return nextFocus.itemView;

}

}

final int limit = Math.min(itemCount, startFocusSearchIndex);

for (int i = limit - 1; i >= 0; i--) {

nextFocus = findViewHolderForAdapterPosition(i);

if (nextFocus == null) {

return null;

}

if (nextFocus.itemView.hasFocusable()) {

return nextFocus.itemView;

}

}

return null;

}

# android.support.v7.preference;

## DropDownPreference

### dropdownPreferenceStyle

**public** DropDownPreference(Context context, AttributeSet attrs) {  
 **this**(context, attrs, R.attr.***dropdownPreferenceStyle***);  
}

### createAdapter

**protected** ArrayAdapter createAdapter() {  
 **return new** ArrayAdapter<>(**mContext**, android.R.layout.***simple\_spinner\_dropdown\_item***);  
}

### android.R.layout.simple\_spinner\_dropdown\_item

**style="?android:attr/spinnerDropDownItemStyle"**

### spinnerDropDownItemStyle

them

<**item name="spinnerDropDownItemStyle"**>@style/Widget.DropDownItem.Spinner</**item**>

<**style name="Widget.DropDownItem"**>  
 <**item name="textAppearance"**>@style/TextAppearance.Widget.DropDownItem</**item**>  
 <**item name="paddingStart"**>@dimen/dropdownitem\_text\_padding\_left</**item**>  
 <**item name="paddingEnd"**>@dimen/dropdownitem\_text\_padding\_right</**item**>  
 <**item name="gravity"**>center\_vertical</**item**>  
</**style**>  
  
<**style name="Widget.DropDownItem.Spinner"**>  
 <**item name="checkMark"**>?attr/listChoiceIndicatorSingle</**item**>  
</**style**>

<**item name="listChoiceIndicatorSingle"**>@drawable/btn\_radio</**item**>

### DialogPreference

R.styleable.*DialogPreference*

*<!-- Base attributes available to DialogPreference. -->*<**declare-styleable name="DialogPreference"**>  
 *<!-- The title in the dialog. -->* <**attr name="dialogTitle" format="string"** />  
 *<!-- The message in the dialog. If a dialogLayout is provided and contains  
 a TextView with ID android:id/message, this message will be placed in there. -->* <**attr name="dialogMessage" format="string"** />  
 *<!-- The icon for the dialog. -->* <**attr name="dialogIcon" format="reference"** />  
 *<!-- The positive button text for the dialog. Set to @null to hide the positive button. -->* <**attr name="positiveButtonText" format="string"** />  
 *<!-- The negative button text for the dialog. Set to @null to hide the negative button. -->* <**attr name="negativeButtonText" format="string"** />  
 *<!-- A layout to be used as the content View for the dialog. By default, this shouldn't  
 be needed. If a custom DialogPreference is required, this should be set. For example,  
 the EditTextPreference uses a layout with an EditText as this attribute. -->* <**attr name="dialogLayout" format="reference"** />  
</**declare-styleable**>

**this**(context, attrs, TypedArrayUtils.*getAttr*(context, R.attr.*dialogPreferenceStyle*,  
 android.R.attr.***dialogPreferenceStyle***));