

C 博客

Hongyang

生命不息，奋斗不止，万事起于忽微，量变引起质变

目录视图

个人资料



鸿洋_

已关注

发私信



访问：12383581次

积分：45557

等级：BLOG > B

排名：第57名

原创：195篇

转载：0篇

译文：6篇

评论：13300条

我的微信公众号

点击直达推送文章汇总

长期为您推荐优秀博文、开源项目、视频等，进入还有好玩的等着你，欢迎扫一扫。



联系方式

新动态

给我写信

QQ群：
497438697
请勿重复加群，Thx

文章分类

【Android 5.x】 (11)

【Android 精彩案例】 (37)

【Android 源码解析】 (29)

原

Android 6.0 运行时权限处理完全解析

标签：[android](#) [6-0](#) [Permission](#)

2016-02-22 09:31

119136人阅读

评论(

分类：

【Android 5.x】 (10)

版权声明：

本文为博主原创文章，未经博主允许不得转载。

目录(?)

[+]

转载请标明出处：

<http://blog.csdn.net/lmj623565791/article/details/50709663>；

本文出自：[【张鸿洋的博客】](#)

一、概述

随着**Android 6.0**发布以及普及，我们开发者所要应对的主要就是新版本SDK带来的一些变化，首先关注的就是权限机制的变化，查看查看官网的这篇文章[http://developer.android.com/intl/zh-cn/about/versions/marshmallow/android-6.0-changes#Runtime Permissions](http://developer.android.com/intl/zh-cn/about/versions/marshmallow/android-6.0-changes#runtime-permissions)。

ok，本篇文章目的之一就是对运行时权限处理的一个介绍，以及对目前权限相关的库的一些了解。

当然非常推荐阅读官网权限相关文章：

- [Working with System Permissions](#)
- [Permissions Best Practices](#)

本文也是在上述文章基础上理解、实验以及封装。

二、运行时权限的变化及特点

对于6.0以下的权限及在安装的时候，根据权限声明产生一个权限列表，用户只有在同意之后才能完成app的安装，造成了我们默认忍受其一些不必要的权限（比如是个app都要访问通讯录、短信等）。而在6.0以后，我们可以直接安装，当app需要我们授予们可以予以拒绝（比如：单机的象棋对战，请求访问任何权限，我都是不同意的）。当然你也可以在设置界面对每个app的权限进行授权或者解除授权。

新的权限机制更好的保护了用户的隐私，Google将权限分为两类，一类是**Normal Permissions**，这类权限一般不涉及用户隐私的，比如手机震动、访问网络等；另一类是Dangerous Permission，一般是涉及到用户隐私的，需要用户进行授权，比如读取

- Normal Permissions如下

- 1
- ACCESS_LOCATION_EXTRA_COMMANDS
- 2
- ACCESS_NETWORK_STATE
- 3
- ACCESS_NOTIFICATION_POLICY
- 4
- ACCESS_WIFI_STATE
- 5
- BLUETOOTH
- 6
- BLUETOOTH_ADMIN

- 【Android 自定义控件实战】 (29)
- 【Android 自定义控件之起步】 (7)
- 【Android 快速开发】 (12)
- 【Android 原生开发游戏】 (3)
- 【Java 并发专题】 (15)
- 【android 进阶之路】 (72)
- 【Java 设计模式】 (10)
- 【Android 百度地图】 (4)
- 【html5 css3精彩案例】 (14)
- 【Android github 控件】 (10)
- 【Android 基础】 (16)
- 【Javascript 】 (9)
- 【rabbitMQ 用法】 (5)
- 【Android微知识点】 (4)

友情链接

郭霖的博客

夏安明的博客

任玉刚的博客

元斌的博客

Github开源项目的配套视频

敬佩的孔老师

foruok的订阅号程序视界

OpenCV大神shiter

专为Android程序员的导航

泡在网上的日子

博客专栏



HTML5 & CSS3 实战

文章：11篇

阅读：170784



设计模式融入生活

文章：10篇

阅读：100454



Android 精彩案例

文章：67篇

阅读：5044793

阅读排行

Android Https相关完全解析 ...

(1563738)

Android Fragment 真正的完...

(578077)

Android RecyclerView 使用...

(526500)

Android OkHttp完全解析 是...

(401615)

Android 自定义View (一)

(256769)

Android 属性动画 (Property...

(239279)

Android Fragment 真正的完...

(215070)

Android 屏幕适配方案

(200994)

Android 手把手教您自定义Vi...

(184479)

Android 沉浸式状态栏攻略 ...

(161791)

文章搜索

7	BROADCAST_STICKY
8	CHANGE_NETWORK_STATE
9	CHANGE_WIFI_MULTICAST_STATE
10	CHANGE_WIFI_STATE
11	DISABLE_KEYGUARD
12	EXPAND_STATUS_BAR
13	GET_PACKAGE_SIZE
14	INSTALL_SHORTCUT
15	INTERNET
16	KILL_BACKGROUND_PROCESSES
17	MODIFY_AUDIO_SETTINGS
18	NFC
19	READ_SYNC_SETTINGS
20	READ_SYNC_STATS
21	RECEIVE_BOOT_COMPLETED
22	REORDER_TASKS
23	REQUEST_INSTALL_PACKAGES
24	SET_ALARM
25	SET_TIME_ZONE
26	SET_WALLPAPER
27	SET_WALLPAPER_HINTS
28	TRANSMIT_IR
29	UNINSTALL_SHORTCUT
30	USE_FINGERPRINT
31	VIBRATE
32	WAKE_LOCK
33	WRITE_SYNC_SETTINGS

Dangerous Permissions:

1	group:android.permission-group.CONTACTS
2	permission:android.permission.WRITE_CONTACTS
3	permission:android.permission.GET_ACCOUNTS
4	permission:android.permission.READ_CONTACTS
5	
6	group:android.permission-group.PHONE
7	permission:android.permission.READ_CALL_LOG
8	permission:android.permission.READ_PHONE_STATE
9	permission:android.permission.CALL_PHONE
10	permission:android.permission.WRITE_CALL_LOG
11	permission:android.permission.USE_SIP
12	permission:android.permission.PROCESS_OUTGOING_CALLS
13	permission:com.android.voicemail.permission.ADD_VOICEMAIL
14	
15	group:android.permission-group.CALENDAR
16	permission:android.permission.READ_CALENDAR
17	permission:android.permission.WRITE_CALENDAR
18	
19	group:android.permission-group.CAMERA
20	permission:android.permission.CAMERA
21	
22	group:android.permission-group.SENSORS
23	permission:android.permission.BODY_SENSORS
24	
25	group:android.permission-group.LOCATION
26	permission:android.permission.ACCESS_FINE_LOCATION
27	permission:android.permission.ACCESS_COARSE_LOCATION
28	
29	group:android.permission-group.STORAGE
30	permission:android.permission.READ_EXTERNAL_STORAGE
31	permission:android.permission.WRITE_EXTERNAL_STORAGE
32	
33	group:android.permission-group.MICROPHONE
34	permission:android.permission.RECORD_AUDIO
35	
36	group:android.permission-group.SMS
37	permission:android.permission.READ_SMS
38	permission:android.permission.RECEIVE_WAP_PUSH
39	permission:android.permission.RECEIVE_MMS
40	permission:android.permission.RECEIVE_SMS
41	permission:android.permission.SEND_SMS
42	permission:android.permission.READ_CELL_BROADCASTS

快速回复

我要收藏

返回顶部

可以通过adb shell pm list permissions -d -g 进行查看。

看到上面的dangerous permissions，会发现一个问题，好像危险权限都是一组一组的，恩，没错，的确是这样的，

最新评论

Android PullToRefresh (ListView Grid...
坚持不放弃 : 为什么不自己写一个

Android Xfermode 实战 实现圆形、圆...
琦琦安卓进阶 : @sinat_33279707:使用默认
的paint

设计模式 适配器模式 以手机充电器为例
Vami杜晓静 : 手机充电器的例子很恰当

设计模式 策略模式 以角色游戏为背景
Vami杜晓静 : 讲解思路很清晰,受益匪浅

设计模式 状态模式 以自动售货机为例
Vami杜晓静 : 16年软考就考的这个,真该
早点来看看楼主分享的,

Android 热修复 Tinker接入及源码浅析
铁血男儿王刚蛋 : 这个文件夹我找不到啊

Android NestedScrolling机制完全解析 ...
chaoxionghuai : 直接 滑动 顶部 “软件介
绍” 区域, 无法滑动, 哪位解决这个问题?

Android NestedScrolling机制完全解析 ...
chaoxionghuai : 点击 顶部 “软件介绍”
区域, 无法滑动, 哪位实现了

Android EventBus源码解析 带你深入理...
mak1993 : EventBus3.0已经使用注解代替
`onEvent` 的命名方式的, 楼主的博客很有参
考价值,谢谢...

Android EventBus源码解析 带你深入理...
mak1993 : @huosange:两个 `onEventXX`
X`都会收到 `Publisher` 发布的信息,因为两...

统计

微信公众号

那么有个问题：分组对我们的权限机制有什么影响吗？

的确是有影响的，如果app运行在Android 6.x的机器上，对于授权机制是这样的。如果你申请某个危险的权限，假设你的app与某个危险权限，那么系统会立即授权，而不需要用户去点击授权。比如你的app对 `READ_CONTACTS` 已经授权了，当你的app申请

直接授权通过。此外，对于申请时弹出的dialog上面的文本说明也是对整个权限组的说明，而不是单个权限（ps:这个dialog是

三、相关API

好在运行时相关的API也比较简单，所以适配起来并不会非常痛苦。

API的讲解就跟着申请权限步骤一起了：

- 1. 在AndroidManifest文件中添加需要的权限。

这个步骤和我们之前的开发并没有什么变化，试图去申请一个没有声明的权限可能会导致程序崩溃。

- 2. 检查权限

```
1  if (ContextCompat.checkSelfPermission(thisActivity,
2      Manifest.permission.READ_CONTACTS)
3      != PackageManager.PERMISSION_GRANTED) {
4  }else{
5      //
6  }
```

这里涉及到一个API，`ContextCompat.checkSelfPermission`，主要用于检测某个权限是否已经为 `PackageManager.PERMISSION_DENIED` 或者 `PackageManager.PERMISSION_GRANTED`。当返回DENI

- 3. 申请授权

```
1  ActivityCompat.requestPermissions(thisActivity,
2      new String[] {Manifest.permission.READ_CONTACTS},
3      MY_PERMISSIONS_REQUEST_READ_CONTACTS);
```

该方法是异步的，第一个参数是Context；第二个参数是需要申请的权限的字符串数组；第三个参数为requestCode，主

以从方法名 `requestPermissions` 以及第二个参数看出，是支持一次性申请多个权限的，系统会通过对话框逐一询问用户是

- 4. 处理权限申请回调

```
1  @Override
2  public void onRequestPermissionsResult(int requestCode,
3      String permissions[], int[] grantResults) {
4      switch (requestCode) {
5          case MY_PERMISSIONS_REQUEST_READ_CONTACTS: {
6              // If request is cancelled, the result arrays are empty.
7              if (grantResults.length > 0
8                  && grantResults[0] == PackageManager.PERMISSION_GRANTED) {
9
10                 // permission was granted, yay! Do the
11                 // contacts-related task you need to do.
12
13             } else {
14
15                 // permission denied, boo! Disable the
16                 // functionality that depends on this permission.
17             }
18             return;
19         }
20     }
21 }
```

ok，对于权限的申请结果，首先验证requestCode定位到你的申请，然后验证grantResults对应于申请的结果，这里的数

权限字符串数组。如果你同时申请两个权限，那么grantResults.length就为2，分别记录你两个权限的申请结果。如果



微信关注我的公众号

- 快速回复
- 我要收藏
- 返回顶部

情了~

当然，到此我们的权限申请的不走，基本介绍就如上述。不过还有个API值得提一下：

```
1 // Should we show an explanation?
2 if (ActivityCompat.shouldShowRequestPermissionRationale(thisActivity,
3     Manifest.permission.READ_CONTACTS))
4     // Show an explanation to the user *asynchronously* -- don't block
5     // this thread waiting for the user's response! After the user
6     // sees the explanation, try again to request the permission.
7
8 }
```

这个API主要用于给用户一个申请权限的解释，该方法只有在用户在上一次已经拒绝过你的这个权限申请。也就是说，用户已经框，你需要给用户一个解释，为什么要授权，则使用该方法。

那么将上述几个步骤结合到一起就是：

```
1 // Here, thisActivity is the current activity
2 if (ContextCompat.checkSelfPermission(thisActivity,
3     Manifest.permission.READ_CONTACTS)
4     != PackageManager.PERMISSION_GRANTED) {
5
6     // Should we show an explanation?
7     if (ActivityCompat.shouldShowRequestPermissionRationale(thisActivity,
8         Manifest.permission.READ_CONTACTS)) {
9
10        // Show an explanation to the user *asynchronously* -- don't block
11        // this thread waiting for the user's response! After the user
12        // sees the explanation, try again to request the permission.
13
14    } else {
15
16        // No explanation needed, we can request the permission.
17
18        ActivityCompat.requestPermissions(thisActivity,
19            new String[]{Manifest.permission.READ_CONTACTS},
20            MY_PERMISSIONS_REQUEST_READ_CONTACTS);
21
22        // MY_PERMISSIONS_REQUEST_READ_CONTACTS is an
23        // app-defined int constant. The callback method gets the
24        // result of the request.
25    }
26 }
```

微信关注我的公众号



四、简单的例子

这里写一个简单的例子，针对于运行时权限。相信大家最开始接触Android的时候，都利用Intent实验了打电话、发短信等功

我们看看直接拨打电话在Android 6.x的设备上权限需要如何处理。

当然代码非常简单：

```
1 package com.zhy.android160217;
2
3 import android.Manifest;
4 import android.content.Intent;
5 import android.content.pm.PackageManager;
6 import android.net.Uri;
7 import android.os.Bundle;
8 import android.support.v4.app.ActivityCompat;
9 import android.support.v4.content.ContextCompat;
10 import android.support.v7.app.AppCompatActivity;
11 import android.view.View;
12 import android.widget.Toast;
13
14 public class MainActivity extends AppCompatActivity
15 {
16
17     private static final int MY_PERMISSIONS_REQUEST_CALL_PHONE = 1;
18
19     @Override
```

快速回复

我要收藏

返回顶部

```
20     protected void onCreate(Bundle savedInstanceState)
21     {
22         super.onCreate(savedInstanceState);
23         setContentView(R.layout.activity_main);
24     }
25
26     public void testCall(View view)
27     {
28         if (ContextCompat.checkSelfPermission(this,
29             Manifest.permission.CALL_PHONE)
30             != PackageManager.PERMISSION_GRANTED)
31         {
32
33             ActivityCompat.requestPermissions(this,
34                 new String[] {Manifest.permission.CALL_PHONE},
35                 MY_PERMISSIONS_REQUEST_CALL_PHONE);
36         } else
37         {
38             callPhone();
39         }
40     }
41
42     public void callPhone()
43     {
44         Intent intent = new Intent(Intent.ACTION_CALL);
45         Uri data = Uri.parse("tel:" + "10086");
46         intent.setData(data);
47         startActivity(intent);
48     }
49
50     @Override
51     public void onRequestPermissionsResult(int requestCode, String[] permissions,
52         int[] grantResults) {
53
54         if (requestCode == MY_PERMISSIONS_REQUEST_CALL_PHONE)
55         {
56             if (grantResults[0] == PackageManager.PERMISSION_GRANTED)
57             {
58                 callPhone();
59             } else
60             {
61                 // Permission Denied
62                 Toast.makeText(MainActivity.this, "Permission Denied", Toast.LENGTH_SHORT).show();
63             }
64             return;
65         }
66         super.onRequestPermissionsResult(requestCode, permissions, grantResults);
67     }
68 }
```



在Android 6.x上运行是，点击testCall，即会弹出授权窗口，如何你Allow则直接拨打电话，如果Denied则Toast弹出“ Perm

例子很简单，不过需要注意的是，对于Intent这种方式，很多情况下是 不需要 授权的甚至权限都不需要的，比如：你是到拨号界
就不需要去申请权限；打开系统图库去选择照片；调用系统相机app去牌照等。更多请参考[Consider Using an Intent](#)

当然，上例也说明了并非所有的通过Intent的方式都不需要申请权限。一般情况下，你是通过Intent打开另一个应用，而用户通
只关注结果（onActivityResult），那么权限是不需要你处理的，而是由打开的app去处理。

五、封装

（1）申请权限

虽然权限处理并不复杂，但是需要编写很多重复的代码，所以目前也有很多库对用法进行了封装，大家可以在github首页搜索：
了几个库的使用方式，发现<https://github.com/lovedise/PermissionGen>这个库据我所见相比较而言使用算是比较简单的，那
础来讲解，大家有兴趣可以多几个库的源码。

封装的代码很简单，正如大家的对上面的权限代码所见，没有特别复杂的地方。

对于申请权限的代码，原本的编写为：


```

1  if (ContextCompat.checkSelfPermission(this,
2      Manifest.permission.CALL_PHONE)
3      != PackageManager.PERMISSION_GRANTED)
4  {
5
6      ActivityCompat.requestPermissions(this,
7          new String[] {Manifest.permission.CALL_PHONE},
8          MY_PERMISSIONS_REQUEST_CALL_PHONE);
9  } else
10 {
11     callPhone();
12 }

```

大家可以看到，对于其他的权限，其实申请的逻辑是类似的；唯一不同的肯定就是参数，那么看上面的代码，我们需要3个参数: Activity|Fragment、权限字符串数组、int型申请码。

也就是说，我们只需要写个方法，接受这几个参数即可，然后逻辑是统一的。

```

1  public static void needPermission(Fragment fragment, int requestCode, String[] permissions)
2  {
3      requestPermissions(fragment, requestCode, permissions);
4  }
5
6  @TargetApi(value = Build.VERSION_CODES.M)
7  private static void requestPermissions(Object object, int requestCode, String[] permissions)
8  {
9      if (!Utils.isOverMarshmallow())
10     {
11         doExecuteSuccess(object, requestCode);
12         return;
13     }
14     List<String> deniedPermissions = Utils.findDeniedPermissions(getActivity());
15
16     if (deniedPermissions.size() > 0)
17     {
18         if (object instanceof Activity)
19         {
20             ((Activity) object).requestPermissions(deniedPermissions.toArray(new String[deniedPermissions.size()]), requestCode);
21         } else if (object instanceof Fragment)
22         {
23             ((Fragment) object).requestPermissions(deniedPermissions.toArray(new String[deniedPermissions.size()]), requestCode);
24         } else
25         {
26             throw new IllegalArgumentException(object.getClass().getName() + " is not supported");
27         }
28     } else
29     {
30         doExecuteSuccess(object, requestCode);
31     }
32 }
33 }

```

微信关注我的公众号



Utils.findDeniedPermissions 其实就是check没有授权的权限。

快速回复

我要收藏

返回顶部

```

1  #Utils
2  @TargetApi(value = Build.VERSION_CODES.M)
3  public static List<String> findDeniedPermissions(Activity activity, String... permissions)
4  {
5      List<String> denyPermissions = new ArrayList<>();
6      for (String value : permissions)
7      {
8          if (activity.checkSelfPermission(value) != PackageManager.PERMISSION_GRANTED)
9          {
10             denyPermissions.add(value);
11         }
12     }
13     return denyPermissions;
14 }

```

那么上述的逻辑就很清晰了，需要的3种参数传入，先去已经申请的权限，然后开始申请权限。

ok，我相信上面代码，大家扫一眼就可以了解了。

(2) 处理权限回调

对于回调，因为要根据是否授权去执行不同的事情，所以很多框架也需要些一连串的代码，或者和前面的申请代码耦合。不过这也是我选择它来讲解的原因。

回调主要做的事情：

- 1. 了解是否授权成功。
- 2. 根据授权情况进行回调。

对于第一条逻辑都一样；对于第二条，因为涉及到两个分支，每个分支执行不同的方法。

对于第二条，很多框架选择将两个分支的方法在申请权限的时候进行注册，然后在回调中根据requestCode进行匹配执行，不过进行对象管理。

不过这个框架采取了一种很有意思的做法，它利用注解去确定需要执行的方法，存在两个注解：

```
1 @PermissionSuccess(requestCode = 100)
2 @PermissionFail(requestCode = 100)
```

利用反射根据授权情况+requestCode即可找到注解标注的方法，然后直接执行即可。

大致的代码为：

```
1 @Override public void onRequestPermissionsResult(int requestCode, String[] permissions,
2           int[] grantResults) {
3     PermissionGen.onRequestPermissionsResult(this, requestCode, permissions,
4     }
5
6 private static void requestResult(Object obj, int requestCode, String[] permissions,
7           int[] grantResults)
8 {
9     List<String> deniedPermissions = new ArrayList<>();
10    for (int i = 0; i < grantResults.length; i++)
11    {
12        if (grantResults[i] != PackageManager.PERMISSION_GRANTED)
13        {
14            deniedPermissions.add(permissions[i]);
15        }
16    }
17
18    if (deniedPermissions.size() > 0)
19    {
20        doExecuteFail(obj, requestCode);
21    } else
22    {
23        doExecuteSuccess(obj, requestCode);
24    }
25 }
```

微信关注我的公众号



快速回复

我要收藏

返回顶部

首先根据grantResults进行判断成功还是失败，对于成功则：

```
1 private static void doExecuteSuccess(Object activity, int requestCode)
2 {
3     Method executeMethod = Utils.findMethodWithRequestCode(activity.getClass(),
4         PermissionSuccess.class, requestCode);
5
6     executeMethod(activity, executeMethod);
7 }
8
9 #Utils
10 public static <A extends Annotation> Method findMethodWithRequestCode(Class clazz, Class<A> annotation)
11 {
12     for (Method method : clazz.getDeclaredMethods())
13     {
14         if (method.isAnnotationPresent(annotation))
15         {
16             if (isEqualRequestCodeFromAnntation(method, annotation, requestCode))
17             {
18                 return method;
19             }
20         }
21     }
22 }
```

```
19         }
20     }
21 }
22 return null;
23 }
```

根据注解和requestCode找到方法，然后反射执行即可。失败的逻辑类似，不贴代码了。

ok，到此我们的运行时权限相对于早起版本的变化、特点、以及如何处理和封装都介绍完了。

不过对于上面讲解的库，肯定有人会说：运行时反射会影响效率，没错，不过我已经在上述代码的基础上将运行时注解改成Annr了，即编译时注解，这样的话，就不存在反射损失效率的问题了。本来准备fork修改，然后PR，结果写完，改动太大，估计PR没项目了，也方便后面的做一些扩展和维护。

详见库：<https://github.com/hongyangAndroid/MPPermissions>。

六、MPPermissions用法

对外的接口和PermissionGen基本一致，因为申请只需要三个参数，抛弃了使用原本类库的单例的方式，直接一个几个静态方法

贴一个用法：

```
1 public class MainActivity extends AppCompatActivity
2 {
3
4     private Button mBtnSdcard;
5     private static final int REQUECT_CODE_SDCARD = 2;
6
7     @Override
8     protected void onCreate(Bundle savedInstanceState)
9     {
10         super.onCreate(savedInstanceState);
11         setContentView(R.layout.activity_main);
12
13         mBtnSdcard = (Button) findViewById(R.id.id_btn_sdcard);
14         mBtnSdcard.setOnClickListener(new View.OnClickListener()
15         {
16             @Override
17             public void onClick(View v)
18             {
19                 MPPermissions.requestPermissions(MainActivity.this, REQUECT_CODE_SDCARD, Manifest.permission.WRITE_EXTERNAL_STORAGE);
20             }
21         });
22     }
23
24     @Override
25     public void onRequestPermissionsResult(int requestCode, String[] permissions, int[] grantResults)
26     {
27         MPPermissions.onRequestPermissionsResult(this, requestCode, permissions, grantResults);
28         super.onRequestPermissionsResult(requestCode, permissions, grantResults);
29     }
30
31     @PermissionGrant(REQUECT_CODE_SDCARD)
32     public void requestSdcardSuccess()
33     {
34         Toast.makeText(this, "GRANT ACCESS SDCARD!", Toast.LENGTH_SHORT).show();
35     }
36
37     @PermissionDenied(REQUECT_CODE_SDCARD)
38     public void requestSdcardFailed()
39     {
40         Toast.makeText(this, "DENY ACCESS SDCARD!", Toast.LENGTH_SHORT).show();
41     }
42 }
43 }
```



- 快速回复
- 我要收藏
- 返回顶部

是不是简单明了~~对于onRequestPermissionsResult所有的Activity都是一致的，所以可以放到BaseActivity中去。此外，在I致，详见demo。

详见库：<https://github.com/hongyangAndroid/MPPermissions>.

至于为什么不直接介绍MPPermission的源码，因为主要涉及到Annotation Processor，所以以这种方式引出，后面考虑单篇博时注解的相关做法以及API的使用。

最后感谢 PermissionGen 库，祝大家元宵节快乐~！

欢迎关注我的微博：

<http://weibo.com/u/3165018720>

讨论群：[284327959](#)，欢迎入群
微信公众号：[hongyangAndroid](#)
(欢迎关注，第一时间推送博文信息)



顶

119

踩

9

微信关注我的公众号

- ▲ 上一篇 Android 一个改善的okHttp封装库
- ▼ 下一篇 为RecyclerView打造通用Adapter 让RecyclerView更加好用

我的同类文章

【Android 5.x】（ 10 ）

• Android NestedScrolling机制完全解...

2016-08-15

阅读 30269

• Android 优雅的为RecyclerView添加H...

2016-07-

• Android 沉浸式状态栏攻略 让你的状态...

2015-09-22

阅读 161789

• Android 增强版百分比布局库 为了适配...

2015-07-

参考知识库



Android知识库
30717 关注 | 2623 收录



.NET知识库
3087 关注 | 827 收录

快速回复

我要收藏

返回顶部

微信开发

18369 关

猜你在找

- 【Android APP开发】Android高...
- Android开发高级组件与框架——...
- Android入门实战教程
- 【技术公开课】Android内存泄漏...
- 2016年最新c语言教程第2天
- Android 6.0 运行时权限处理完全...
- Android 6.0 运行时权限处理完全...
- Android 6.0 运行时权限处理完全...
- Android 6.0 运行时权限处理完全...
- Android 6.0 运行时权限处理完全...

查看评论



HalmLee

13

我调用requestPermissions这个第一次弹框，拒绝了之后。就算我再调用requestPermissions，也不会弹框了。
然后这个接口shouldShowRequestPermissionRationale，我为什么永远都是返回false



hjr365708064

回复HalmLee: Error:A problem occurred configuring project 'permission-lib'> C
annot access first() element from an empty List，我下载Mpression后，加载项目出现这个问题，你出现了吗？



HalmLee

回复HalmLee: 用的小米4，android系统6.0



lucy10

133楼 20

大神，你说的那个讨论群不能加进去啊！！可以的话可以拉我进去吗啊？？
还有，你的这个方法我在搜索蓝牙的设备的时候再判断不行吗？为什么我在搜索设备的时候判断，刚刚安装APP的时候就会提示获取权限，你选择允许或是否，下次继续的时候怎么都没有提醒获取定位权限了，这个是miui的bug还是怎么回事？而且我在MIUI上根本搜不到设备，在平板上是可以搜索到的。



发光棒

132楼 20

非常实用



CAI.

我把代码写到基类里，然后在继承基类的活动中发起权限请求，运行后点击拒绝后对话框消失时闪退报错了：can not find PermissionTestActivity\$\$PermissionProxy , something when compiler.求解释，改正~希望大神能给个把权限库写在基类里使用的demo！！感激！



CAI.

回复CAI.: 有个建议致洋哥：“MPermissions.shouldShowRequestPermissionRationale(this, permissions[0], 0)”，这个方法要是能支持传入String[]的参数就更方便了。如果一次性请求多个权限，又希望每个权限无论哪个被拒绝都弹出解释的情况。目前只能由于这个方法只能传入一个权限值，所以现在想要实现上面的情况，需要用 && 符号把每个这个方法串联起来放到if的判断中，会比较麻烦。然而我的想法是希望可以一次性传入一个权限值的数组可以免去那种一长串的写法。不过写到这里，我突然又想到了一个方法，可以不用每次写一长串的判断，那就是我自己写一个方法来封装上面的方法，来传入权限数组及其他参数，判断后返回true或false，也可以比较方便的解决我提出的问题，但是，还是希望洋哥的库中能直接有这种方法~~



CAI.

Re: 20

回复CAI.: 经过自己的测试，成功和基类绑定使用了！~~原来大神不经意说的一句话“对于onRequestPermissionsResult所有的Activity都是一致的，所以可以放到BaseActivity中去”，我竟然想的太多了。我原本错误的做法是把onRequestPermissionsResult和3个带注解的都放到基类中，并想在基类中写3个对应3种情况的抽象方法让其他活动实现，结果PermissionTestActivity活动中都没有一个注解，所以就报错了，现在把3个带注解的方法放到PermissionTestActivity活动中就正常了。从这个错误做法中也获得使用大神库的启发：@注解要使用在this活动中。因为请求时传入了this这个活动，this这个活动就继承权限代理接口而成了权限代理，返回结果当然还是找这个权限代理来返回。其实在非基类活动中使用注解来确定调用的方法的代价比在基类写3种情况对应的抽象方法让子类来实现的代价要低，当然对于这个库也是无法像后者说的方式来实现（亲测），但是真心感叹大神的奇思妙想！赞！



奔跑的小黄人、

Re: 20

回复CAI.: 我也这个错误。怎么整的。囧



qq_36279344

130楼 20

在非activity中，怎么使用那个处理权限的回调方法onRequestPermissionsResult(int requestCode, String[] permissions, int[] grantResults){}呢？



阿甘的巧克力

129楼 20


如果是正式版本混淆的时候需加上
-keepclassmembers class * extends android.app.Activity {

微信关注我的公众号



- 快速回复
- 我要收藏
- 返回顶部

```
public void doFailSomething());
public void doSomething();
}
还需要过滤fragment,application,service...
如果有好的办法告知一下 --.
```



寸步不行

表示6.0和6.0.1的系统效果不一样。就是6.0.1的可以走拒绝权限后的回调，6.0不走？

128楼 2017-02-20 14:00



Lin野人

谢谢大神


127楼 2017-02-20 13:59



lopez66

谢谢，明天再试试！

126楼 2017-02-20 13:58



裂缝中的阳光dg

最近也写了一篇《探讨Android 6.0及以上新权限系统的检测与处理》，向大神学习！<http://blog.csdn.net/andrexpert/article/details/53331836>


125楼 2017-02-20 13:57



arbiter208

洋神，看了下原来那个类库的源码，感觉并不是单例模式吧。

124楼 2017-02-20 13:56




liufengxin

还有一个问题，版本sdk在23以下，但是也有可能在6.0以上的手机运行，也是需要做配置的吧，但是我ContextCompat.checkSelfPermission这个方法没有啊




裂缝中的阳光dg

回复liufengxin：请看 <http://blog.csdn.net/andrexpert/article/details/53331836>



Jay白色风车

回复liufengxin：我也疑惑这个问题呢！我的jar文件，在配置文件中targetSdkVersion = 24在引用其的项目中，targetSdkVersion = 21 此时，运行在Android6.0的手机上。肯定是要加权限的吧。网上是这么说的“Android6.0系统默认为targetSdkVersion小于23的应用默认授予了所申请的所有权限，所以如果你以前的APP设置的targetSdkVersion低于23，在运行时也不会崩溃，”



liufengxin

回复Jay白色风车：jar文件在配置文件中的targetSdkVersion>23?也就是说我的工程targetSdkVersion是小于23的，但是引入了一些比较新的jar包，也是需要做动态权限配置的是吧？这个时候的权限配置是不是只要配置用到jar包方法的地方还是所有的权限都是需要动态配置？




liufengxin

下面推荐的那个打包好的工程，是studio的吧，我要是在eclipse上想依赖要怎么做呢？？江湖救急




414、小哥

回复liufengxin：<http://blog.csdn.net/u010838785/article/details/52881386>



liufengxin

回复414、小哥：这种方法没试过，可以试试



DongDonggogo

运行时都说好，问题是怎么去找23以前项目的运行时权限在什么地方使用过呢

和小胖

Re: 2017-02-20 13:55

微信关注我的公众号





回复DongDonggogo: 那就需要看代码了。或者了解现在项目的功能。某个地方需要录音, 那么这个地方肯定是要的。应该没有一种工具, 直接来检测哪里用了这些危险权限吧。



Coder_Zhou

Re: 20

回复DongDonggogo: 能告知一下, 是怎么找的吗O__O"...



Alpha58

120楼 20

解决我的问题!



linwujia

119楼 20

问一下, 在自定义View的时候, 需要获取未接电话, 这时需要权限, 怎么申请运行时权限? 这个运行时权限是只针对Activity的吗?



花-开-花-谢

118楼 20

我的是galaxy c7 是不是系统已经帮我做处理权限了



花-开-花-谢

117楼 20

我的是6.01系统, 为什么没弹出授权窗口, 感觉和6.0之前一样, 没有多大区别



Burgessb

回复花-开-花-谢: 我也没弹出来



KumuiP

大神, jar包里面的代码是<23直接调取成功逻辑的。但是现在很多23以下的手机可以对权限进行管理可以随时关闭, 这样的话。就不准了吧。



蒙奇D路飞--

回复KumuiP: 对, 这种情况很正常, 所以权限也是个头痛的问题, 没有兼容6.0和6.0以下的权限库, 处理太麻烦了, 请问兄台是如何处理的



裂缝中的阳光dg

Re: 20

回复蒙奇D路飞--: 请看 <http://blog.csdn.net/andrexpert/article/details/53331836>



有道理

115楼 20

当弹出窗口点击拒绝时, onRequestPermissionsResult回调返回的grantResults也是PackageManager.PERMISSION_GRANTED呢?



hjiangshujing

这个没有demo 吗, 不太能看明白

快速回复

14楼 20

我要收藏

返回顶部



cxjlong

113楼 20

引入后运行为什么报这个错啊?? Caused by: java.lang.RuntimeException: can not find SplashActivity_\$\$PermissionProxy, something when compiler.



hezejiao

Re: 20

回复cxjlong: 我也是报这个错, 请问你解决啦吗?



yxning2012

112楼 20

然而国内这个环境, 很多都有自己的安全管理软件, 用那些代码请求权限为0, 但是实际已经被安全软件禁止了

倔强的包子

111楼 20



微信关注我的公众号



你好 问一下如何同时处理多个权限。我现在需要文件下载，在获取到WRITE_EXTERNAL_STORAGE这个权限后还需要READ_PHONE_STATE.



xiehoushilianna

Re: 20

回复倔强的包子：这两个是一个权限组里面的，你只要申请了一个，另一个就有了；如果要申请多个权限，可以在requestPermissions(String [] permissions,int code)的权限数组中添加多个权限



倔强的包子

110楼 20

你好 问一下如何同时处理多个权限。我现在需要文件下载，在获取到WRITE_EXTERNAL_STORAGE这个权限后还需要READ_PHONE_STATE.



裂缝中的阳光dg

Re: 20

回复倔强的包子：<http://blog.csdn.net/andexpert/article/details/53331836>



倔强的包子

109楼 20

你好 问一下如何同时处理多个权限。我现在需要文件下载，在获取到WRITE_EXTERNAL_STORAGE这个权限后还需要READ_PHONE_STATE.



qq_29598851

108楼 20

getActivity(object) 这是什么方法



xm447975373

鸿神，你这jar包里面没有shouldShowRequestPermissionRationale()这个方法吧！一直包找不到这个方法！



xiexin52k

回复xm447975373：我也发现这个问题.



LouisZhoun

106楼 20

大神源码分享一份呢，谢谢。



coolzpw

105楼 20

我错了。。。-- 有提示 log弄了刷选 我晕了



coolzpw

104楼 20

大神请教下，我用华为Mate7 系统已经是6.0了 运行上面代码就会崩，而且没Log- 怎么破



wytiger666

shouldShowRequestPermissionRationale大神这个方法讲得有问题吧,应该在返回false的时候解释,不是返回true的时候解释,不要想当然哦哦.

shouldShowRequestPermissionRationale()返回值 的说明:

1. 第一次请求权限时，用户拒绝了，下一次：shouldShowRequestPermissionRationale() 返回 true，应该显示一些为什么需要这个权限的说明
2. 第二次请求权限时，用户拒绝了，并选择了“不在提醒”的选项时：shouldShowRequestPermissionRationale() 返回 false
3. 设备的策略禁止当前应用获取这个权限的授权：shouldShowRequestPermissionRationale() 返回 false



qq_29986637

102楼 20

鸿神，我用eclipse开发的 没有ContextCompat.checkSelfPermission这个方法呢，该怎么解决啊？v4包的原因吗？看了好多方法也没解决 求回复



假装是AA

Re: 20

微信关注我的公众号



快速回复

03楼 20

我要收藏

返回顶部



101楼 20

回复u011192000: 补充下 测试了下, 魅族系统5.1不管是权限...



Re: 20



100楼 20



99樓 20

回复Lucky jhf: 我也出现了这个问题用eclipse 开发的 你解决了吗

微信关注我的公众号



97楼 20



96樓 20

95楼 20

 快速回复

C

94楼 20

☆ 我要收藏

[^ 返回顶部](#)

93楼 20

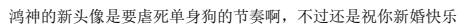
1. [https://www.youtube.com/watch?v=...](#)
 2. [https://www.youtube.com/watch?v=...](#)
 3. [https://www.youtube.com/watch?v=...](#)
 4. [https://www.youtube.com/watch?v=...](#)
 5. [https://www.youtube.com/watch?v=...](#)
 6. [https://www.youtube.com/watch?v=...](#)
 7. [https://www.youtube.com/watch?v=...](#)
 8. [https://www.youtube.com/watch?v=...](#)
 9. [https://www.youtube.com/watch?v=...](#)
 10. [https://www.youtube.com/watch?v=...](#)

92楼 20



91楼 20

90樓 20



不能这样啊，你答应我的，让我给你生猴子的！



我觉得你们俩确实是商量好的！
新婚快乐！郭神&张哥！哈哈哈



博主,请问<<空空狐>>这个软件是你编写的吗?

我反编译看到有zhy的包名,里面是你之前的autoLayout和百分比布局.

恩,我想问一下,对于空空狐里面的大量图片是如何缓存的?只是LruCache和DiskLruCache吗?

还有就是用户发布物品的图片时,图片是如何缩放并且保证不失真然后上传到服务器的?

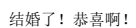
这俩问题是我正在做的demo里面的两个难点,您能给解答一下吗?



666666666666666666.结婚了。恭喜，恭喜



我怎么没弹出授权框, `grantResults.length > 0`
`&& grantResults[0] == PackageManager.PERMISSION_GRANTED`这个直接执行了`false`



翔哥结婚了吗？恭喜！恭喜！恭喜！



我猜你指定是跟着老郭的头像来的，哈哈，一语中的有木有。



为头像点个赞 $O(n_n)$ 哈哈!



引用“breeze_wf”的评论: [\[quote=u013023556\]鸿洋大神、写一篇关于Retrofit+RxJava整合开发的网络...](#)



上礼拜看到换的头像，惊呆了，不过感觉还是之前的头像比较亲民，这个头像伤害太高了，直击心灵



头像啊，就在那儿虐我们吧， $O(\geq \square \leq)O$



头像给单身狗带来1w点伤害



[^ 返回顶部](#)



sinat_33694144

75楼 2017-02-20 14:00

哇，大神结婚啦



wangdong20

74楼 2017-02-20 13:59

鸿洋大神和郭大神换头像是商量好的吧，我差点找不到你们了



逆水当行舟

73楼 2017-02-20 13:58

给头像点赞



乐派天使

72楼 2017-02-20 13:57

洋哥，你和郭哥是不是商量好的，先后晒晒你们的幸福照啊？恭喜！赶紧整出两只猴子吧，哈哈



Aloha_QoQ

71楼 2017-02-20 13:56

鸿神新头像好评。。祝福。。



大脸铭哥

70楼 2017-02-20 13:55

哈哈，鸿洋大神结婚啦。。。恭喜恭喜，幸福快乐



duzzi

69楼 2017-02-20 13:54

大神换头像了啊，祝幸福



Nice_gh

68楼 2017-02-20 13:53

昨天看到这个就好了~~~弄了好久



花-开-花-谢

67楼 2017-02-20 13:52

亲 怎么让 Android 程序一直后台运行，像微信一样有好几个进程 正在运行.....



雪域光芒

66楼 2017-02-20 13:51

像魅族的flyme之类的系统不是原生的系统，5.0的设备也是在要使用的时候提示授权，ContextCompat.checkSelfPermission(this,Manifest.permission.CALL_PHONE)!=PackageManager.PERMISSION_GRANTED这个方法根本就没用，这个时候权限管理，我想知道有什么方法可以检查当前是否拥有某一项权限？



步行vs天下

Re: 2017-02-20 13:50

回复雪域光芒：我的魅蓝note2就出现了这样的问题，5.1的系统需要动态分配权限，郁闷了！一直找不到好的解决方法



雪域光芒

Re: 2017-02-20 13:50

回复雪域光芒：补充下 测试了下，魅族系统5.1不管是权限拒绝还是允许都是ContextCompat.checkSelfPermission(this,Manifest.permission.CALL_PHONE)== PackageManager.PERMISSION_GRANTED，只不过当拒绝时打不出去电话，鸿洋大神能否指点下



人心太闪烁y

65楼 2017-02-20 13:49

为这种付出精神点赞

查看更多评论

发表评论

用户名： xthy_

评论内容：

快速回复

我要收藏

返回顶部

提交

* 以上用户言论只代表其个人观点，不代表CSDN网站的观点或立场

核心技术类目

- [全部主题](#)[Hadoop](#)[AWS](#)[移动游戏](#)[Java](#)[Android](#)[iOS](#)[Swift](#)[智能硬件](#)[Docker](#)[OpenStack](#)[VP](#)[IE10](#)[Eclipse](#)[CRM](#)[JavaScript](#)[数据库](#)[Ubuntu](#)[NFC](#)[WAP](#)[jQuery](#)[BI](#)[HTML5](#)[Spring](#)[Apac](#)[HTML](#)[SDK](#)[IIS](#)[Fedora](#)[XML](#)[LBS](#)[Unity](#)[Splashtop](#)[UML](#)[components](#)[Windows Mobile](#)[Ra](#)[Cassandra](#)[CloudStack](#)[FTC](#)[coremail](#)[OPhone](#)[CouchBase](#)[云计算](#)[iOS6](#)[Rackspace](#)[Web App](#)[S](#)[Compuware](#)[大数据](#)[apttech](#)[Perl](#)[Tornado](#)[Ruby](#)[Hibernate](#)[ThinkPHP](#)[HBase](#)[Pure](#)[Solr](#)[An](#)[Cloud Foundry](#)[Redis](#)[Scala](#)[Django](#)[Bootstrap](#)

公司简介 | 招贤纳士 | 广告服务 | 联系方式 | 版权声明 | 法律顾问 | 问题报告 | 合作伙伴 | 论坛反馈

网站客服 杂志客服 微博客服

webmaster@csdn.net 400-600-2320

北京创新乐知信息技术有限公司 版权所有 | 江苏知之为计算机有限公司 | 江苏乐知网络技术有限公司

京 ICP 证 09002463 号 | Copyright © 1999-2016, CSDN.NET, All Rights Reserved

微信关注我的公众号



快速回复

我要收藏

返回顶部

http://blog.csdn.net/Imj623565791/article/details/50709663

17/17