集成方式

　　在app的build.gradle文件里：

dependencies {

compile('com.github.hotchemi:permissionsdispatcher: 2.4.0') {

// if you don't use android.app.Fragment you can exclude support for them

exclude module: "support-v13"

}

annotationProcessor 'com.github.hotchemi:permissionsdispatcher-processor:2.4.0'

}

**@RuntimePermissions注解**：这是必须使用的注解，用于标注在你想要申请权限的Activity或者Fragment上，如demo里面的PermissionsDispatcherActivity：

@RuntimePermissions

public class PermissionsDispatcherActivity extends AppCompatActivity implements View.OnClickListener {

}

* **@NeedsPermission注解**：这也是必须使用的注解，用于标注在你要获取权限的方法，注解括号里面有参数，传入想要申请的权限。**也就是说你获取了相应的权限之后就会执行这个方法**：

//获取多个权限

@NeedsPermission({Manifest.permission.WRITE\_EXTERNAL\_STORAGE, Manifest.permission.RECORD\_AUDIO})

public void getMulti() {

Toast.makeText(this, "getMulti", Toast.LENGTH\_SHORT).show();

}

**@OnShowRationale注解**：这个不是必须的注解，用于标注申请权限前需要执行的方法，注解   
括号里面有参数，传入想要申请的权限，而且这个方法还要传入一个PermissionRequest对象，这个对象有两种方法：proceed()让权限请求继续，cancel()让请求中断。**也就是说，这个方法会拦截你发出的请求，这个方法用于告诉用户你接下来申请的权限是干嘛的，说服用户给你权限。**

@OnShowRationale({Manifest.permission.WRITE\_EXTERNAL\_STORAGE, Manifest.permission.RECORD\_AUDIO})

//给用户解释要请求什么权限，为什么需要此权限

void showRationale(final PermissionRequest request) {

new AlertDialog.Builder(this)

.setMessage("使用此功能需要WRITE\_EXTERNAL\_STORAGE和RECORD\_AUDIO权限，下一步将继续请求权限")

.setPositiveButton("下一步", new DialogInterface.OnClickListener() {

@Override

public void onClick(DialogInterface dialog, int which) {

request.proceed();//继续执行请求

}

}).setNegativeButton("取消", new DialogInterface.OnClickListener() {

@Override

public void onClick(DialogInterface dialog, int which) {

request.cancel();//取消执行请求

}

})

.show();

}

* **@OnPermissionDenied注解：**这个也不是必须的注解，用于标注如果权限请求失败，但是用户没有勾选不再询问的时候执行的方法，注解括号里面有参数，传入想要申请的权限。**也就是说，我们可以在这个方法做申请权限失败之后的处理，如像用户解释为什么要申请，或者重新申请操作等。**

@OnPermissionDenied({Manifest.permission.WRITE\_EXTERNAL\_STORAGE, Manifest.permission.RECORD\_AUDIO})//一旦用户拒绝了

public void multiDenied() {

Toast.makeText(this, "已拒绝一个或以上权限", Toast.LENGTH\_SHORT).show();

}

**@OnNeverAskAgain注解：**这个也不是必须的注解，用于标注如果权限请求失败,而且用户勾选不再询问的时候执行的方法，注解括号里面有参数，传入想要申请的权限。**也就是说，我们可以在这个方法做申请权限失败并选择不再询问之后的处理。例如，可以告诉作者想开启权限的就从手机设置里面开启。**

@OnNeverAskAgain({Manifest.permission.WRITE\_EXTERNAL\_STORAGE, Manifest.permission.RECORD\_AUDIO})//用户选择的不再询问

public void multiNeverAsk() {

Toast.makeText(this, "已拒绝一个或以上权限，并不再询问", Toast.LENGTH\_SHORT).show();

}

注意，这些注解的方法都不能是private

　　使用PermissionsDispatcher除了要实现注解之外，还要重写Activity的onRequestPermissionsResult()方法，在里面让一个PermissionsDispatcher执行回调。这个PermissionsDispatcher是什么来的呢？

@Override

public void onRequestPermissionsResult(int requestCode, @NonNull String[] permissions, @NonNull int[] grantResults) {

super.onRequestPermissionsResult(requestCode, permissions, grantResults);

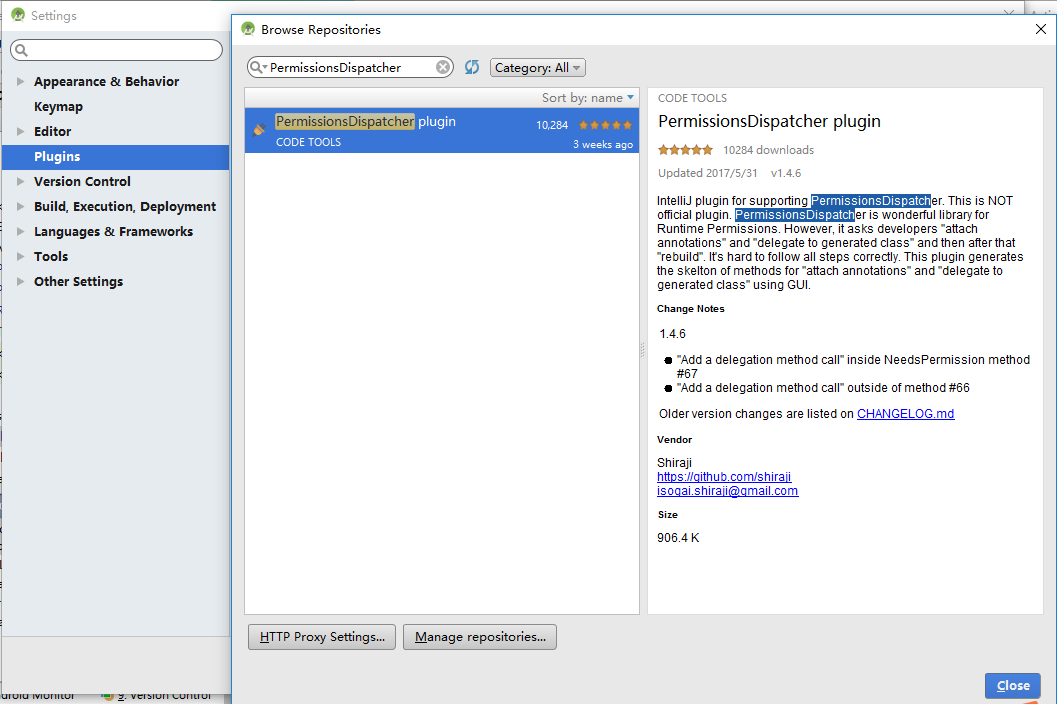
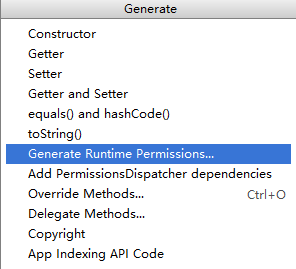
PermissionsDispatcherActivityPermissionsDispatcher.onRequestPermissionsResult(this, requestCode, grantResults);

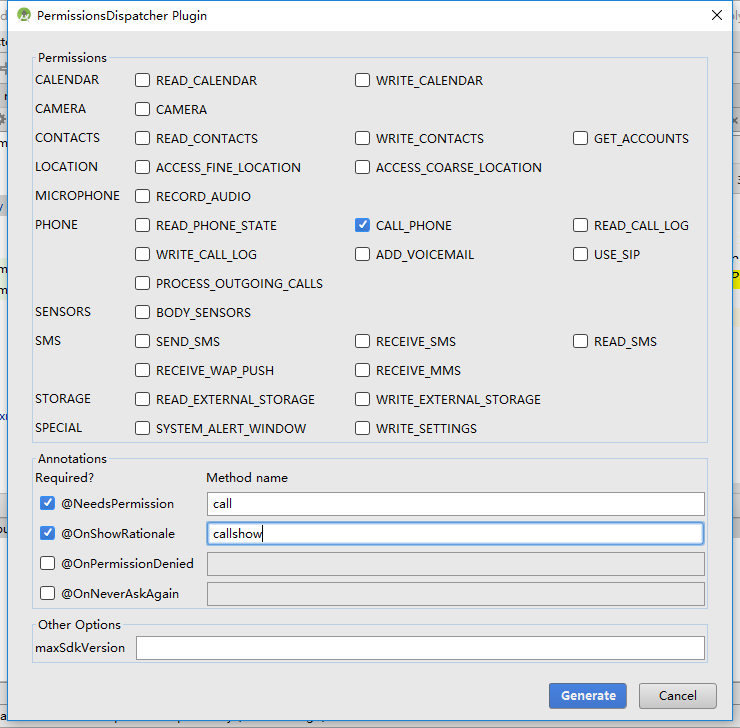
}

申请权限：

MainActivityPermissionsDispatcher.needPermissionWithCheck(this);

### 插件

　　觉得这么多注解要自己一个一个弄不够方便，PermissionsDispatcher还在AndroidStudio做了插件，只要在setting设置里的plugins界面里搜索PermissionsDispatcher就可以安装了，安装完重启一下就能使用：   
   
- 在所需的Activity或者Fragment的代码里面右键，选择Generate，然后就可以选择Generate Runtime Permissions…（生成动态权限的生成）或者下面的Add PermissionsDispatcher dependencies（添加PermissionsDispatcher依赖）   


* 点击Generate Runtime Permissions…出现如下界面，输入方法名字就能生成，很简单粗暴：   
  

生成了这个，如果你没onRequestPermissionsResult和@RuntimePermissions的话也会帮你加上：

@NeedsPermission(Manifest.permission.CALL\_PHONE)

void call() {

}

@OnShowRationale(Manifest.permission.CALL\_PHONE)

void callshow(final PermissionRequest request) {

}