Please note that GitHub no longer supports your web browser.
We recommend upgrading to the latest Google Chrome or Firefox.

Ignore Learn more

googlesamples / **easypermissions**

Simplify Android M system permissions https://firebaseopensource.com/projec...

#android #android-library #permissions

175 commits | 14 branches | 12 releases | 21 contributors | Apache-2.0

Branch: master ▾ | New pull request | Create new file | Upload files | Find File | Clone or download ▾

samtstern Migrate to AndroidX (#272) Latest commit 67d611b Jan 23, 2019

.github	Create ISSUE_TEMPLATE.md	Nov 27, 2017
app	Migrate to AndroidX (#272)	Jan 23, 2019
easypermissions	Migrate to AndroidX (#272)	Jan 23, 2019
gradle/wrapper	Update Gradle wrapper to 4.10.3 (#270)	Jan 9, 2019
.gitignore	Fix publishing	Aug 6, 2018
.travis.yml	Update jacoco task name	Dec 21, 2018
CONTRIBUTING.md	Fix whitespace errors and typo	Sep 14, 2016
LICENSE	Update License Information (#148)	Aug 8, 2017
README.md	Migrate to AndroidX (#272)	Jan 23, 2019
build.gradle	Migrate to AndroidX (#272)	Jan 23, 2019
gradle.properties	Migrate to AndroidX (#272)	Jan 23, 2019
gradlew	Update Gradle wrapper to 4.10.3 (#270)	Jan 9, 2019
gradlew.bat	Update Gradle wrapper to 4.10.3 (#270)	Jan 9, 2019
settings.gradle	Readying for initial release	Dec 17, 2015

README.md

EasyPermissions build passing codecov 72% Android Weekly #185

EasyPermissions is a wrapper library to simplify basic system permissions logic when targeting Android M or higher.

Installation

EasyPermissions is installed by adding the following dependency to your `build.gradle` file:

```
dependencies {
    // For developers using AndroidX in their applications
    implementation 'pub.devrel:easypermissions:3.0.0'

    // For developers using the Android Support Library
    implementation 'pub.devrel:easypermissions:2.0.1'
}
```

Usage

Basic

To begin using EasyPermissions, have your `Activity` (or `Fragment`) override the `onRequestPermissionsResult` method:

```java
public class MainActivity extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }

    @Override
    public void onRequestPermissionsResult(int requestCode, String[] permissions, int[] grantResults) {
        super.onRequestPermissionsResult(requestCode, permissions, grantResults);

        // Forward results to EasyPermissions
        EasyPermissions.onRequestPermissionsResult(requestCode, permissions, grantResults, this);
    }
}
```

Request Permissions

The example below shows how to request permissions for a method that requires both `CAMERA` and `ACCESS_FINE_LOCATION` permissions. There are a few things to note:

- Using `EasyPermissions#hasPermissions(...)` to check if the app already has the required permissions. This method can take any number of permissions as its final argument.
- Requesting permissions with `EasyPermissions#requestPermissions` . This method will request the system permissions and show the rationale string provided if necessary. The request code provided should be unique to this request, and the method can take any number of permissions as its final argument.
- Use of the `AfterPermissionGranted` annotation. This is optional, but provided for convenience. If all of the permissions in a given request are granted, *all* methods annotated with the proper request code will be executed(be sure to have an unique request code). The annotated method needs to be *void* and *without input parameters* (instead, you can use *onSaveInstanceState* in order to keep the state of your suppressed parameters). This is to simplify the common flow of needing to run the requesting method after all of its permissions have been granted. This can also be achieved by adding logic on the `onPermissionsGranted` callback.

```java
@AfterPermissionGranted(RC_CAMERA_AND_LOCATION)
private void methodRequiresTwoPermission() {
    String[] perms = {Manifest.permission.CAMERA, Manifest.permission.ACCESS_FINE_LOCATION};
    if (EasyPermissions.hasPermissions(this, perms)) {
        // Already have permission, do the thing
        // ...
    } else {
        // Do not have permissions, request them now
        EasyPermissions.requestPermissions(this, getString(R.string.camera_and_location_rationale),
                RC_CAMERA_AND_LOCATION, perms);
    }
}
```

Or for finer control over the rationale dialog, use a `PermissionRequest` :

```java
EasyPermissions.requestPermissions(
        new PermissionRequest.Builder(this, RC_CAMERA_AND_LOCATION, perms)
                .setRationale(R.string.camera_and_location_rationale)
                .setPositiveButtonText(R.string.rationale_ask_ok)
                .setNegativeButtonText(R.string.rationale_ask_cancel)
                .setTheme(R.style.my_fancy_style)
                .build());
```

Optionally, for a finer control, you can have your `Activity` / `Fragment` implement the `PermissionCallbacks` interface.

```java
public class MainActivity extends AppCompatActivity implements EasyPermissions.PermissionCallbacks {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }

    @Override
    public void onRequestPermissionsResult(int requestCode, String[] permissions, int[] grantResults) {
        super.onRequestPermissionsResult(requestCode, permissions, grantResults);

        // Forward results to EasyPermissions
        EasyPermissions.onRequestPermissionsResult(requestCode, permissions, grantResults, this);
    }

    @Override
    public void onPermissionsGranted(int requestCode, List<String> list) {
        // Some permissions have been granted
        // ...
    }

    @Override
    public void onPermissionsDenied(int requestCode, List<String> list) {
        // Some permissions have been denied
        // ...
    }
}
```

Required Permissions

In some cases your app will not function properly without certain permissions. If the user denies these permissions with the "Never Ask Again" option, you will be unable to request these permissions from the user and they must be changed in app settings. You can use the method `EasyPermissions.somePermissionPermanentlyDenied(...)` to display a dialog to the user in this situation and direct them to the system setting screen for your app:

Note: Due to a limitation in the information provided by the Android framework permissions API, the `somePermissionPermanentlyDenied` method only works after the permission has been denied and your app has received the `onPermissionsDenied` callback. Otherwise the library cannot distinguish permanent denial from the "not yet denied" case.

```
@Override
public void onPermissionsDenied(int requestCode, List<String> perms) {
    Log.d(TAG, "onPermissionsDenied:" + requestCode + ":" + perms.size());

    // (Optional) Check whether the user denied any permissions and checked "NEVER ASK AGAIN."
    // This will display a dialog directing them to enable the permission in app settings.
    if (EasyPermissions.somePermissionPermanentlyDenied(this, perms)) {
        new AppSettingsDialog.Builder(this).build().show();
    }
}

@Override
public void onActivityResult(int requestCode, int resultCode, Intent data) {
    super.onActivityResult(requestCode, resultCode, data);

    if (requestCode == AppSettingsDialog.DEFAULT_SETTINGS_REQ_CODE) {
        // Do something after user returned from app settings screen, like showing a Toast.
        Toast.makeText(this, R.string.returned_from_app_settings_to_activity, Toast.LENGTH_SHORT)
                .show();
    }
}
```

Interacting with the rationale dialog

Implement the `EasyPermissions.RationaleCallbacks` if you want to interact with the rationale dialog.

```
@Override
public void onRationaleAccepted(int requestCode) {
    // Rationale accpets to request some permissions
    // ...
}

@Override
public void onRationaleDenied(int requestCode) {
    // Rationale denied to request some permissions
    // ...
}
```

Rationale callbacks don't necessarily imply permission changes. To check for those, see the `EasyPermissions.PermissionCallbacks`.

LICENSE