1 Permissions and API calls diagram

1. Calendar permissions

Reference open source software：

https://github.com/mapbox/mapbox-android-demo/tree/master/MapboxAndroidDemo

（1）Permissions and corresponding APIs

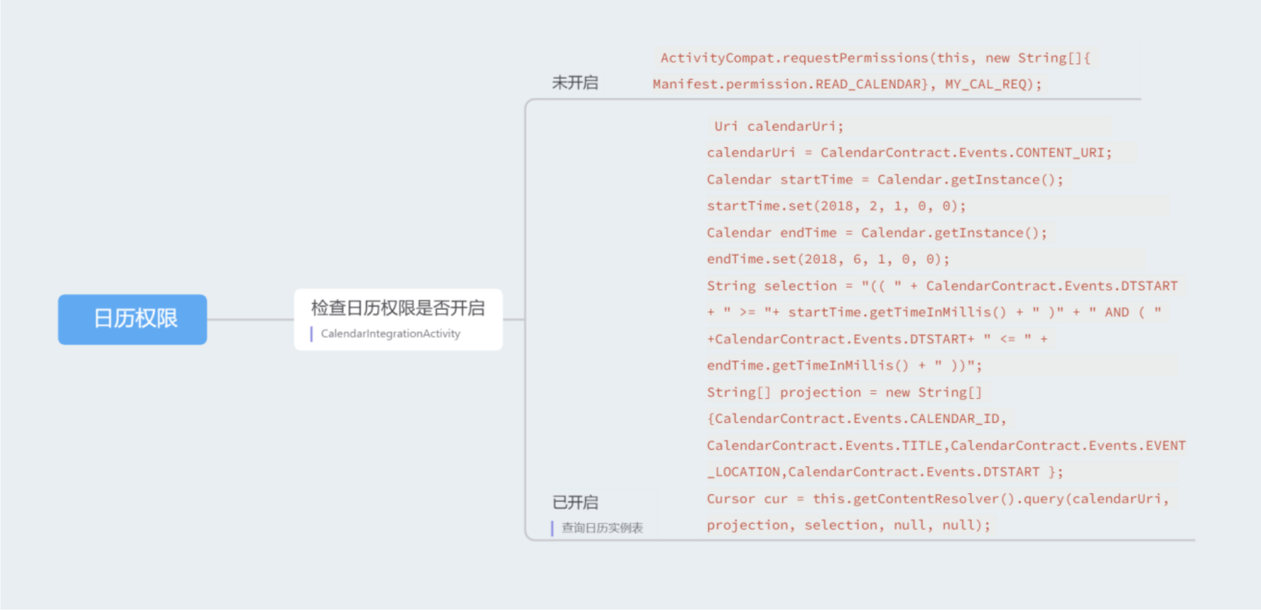
Permissions：

<uses-permission android:name="android.permission.READ_CALENDAR"></uses-permission>

API：

android.provider.CalendarContract

（2）Call diagram



```
ActivityCompat.requestPermissions(this, new String[]{
Manifest.permission.READ_CALENDAR}, MY_CAL_REQ);
```

```
Uri calendarUri;
calendarUri = CalendarContract.Events.CONTENT_URI;
Calendar startTime = Calendar.getInstance();
startTime.set(2018, 2, 1, 0, 0);
Calendar endTime = Calendar.getInstance();
endTime.set(2018, 6, 1, 0, 0);
String selection = "(( " + CalendarContract.Events.DTSTART
+ " >= "+ startTime.getTimeInMillis() + " )" + " AND ( "
+CalendarContract.Events.DTSTART+ " <= " +
endTime.getTimeInMillis() + " ))";
String[] projection = new String[]
{CalendarContract.Events.CALENDAR_ID,
CalendarContract.Events.TITLE,CalendarContract.Events.EVENT
_LOCATION,CalendarContract.Events.DTSTART };
Cursor cur = this.getContentResolver().query(calendarUri,
projection, selection, null, null);
```

2. Phone Permissions

（1）Permissions and corresponding APIs

Permissions：

<uses-permission android:name="android.permission. READ_PHONE_STATE "></uses-permission>

API：

A lot of open source software uses READ_PHONE_STATE permission, but its corresponding API is not found.

（2）Call diagram

3. Camera Permissions

Reference open source software：

https://github.com/masuvern/barinsta

（1）Permissions and corresponding APIs

Permissions：

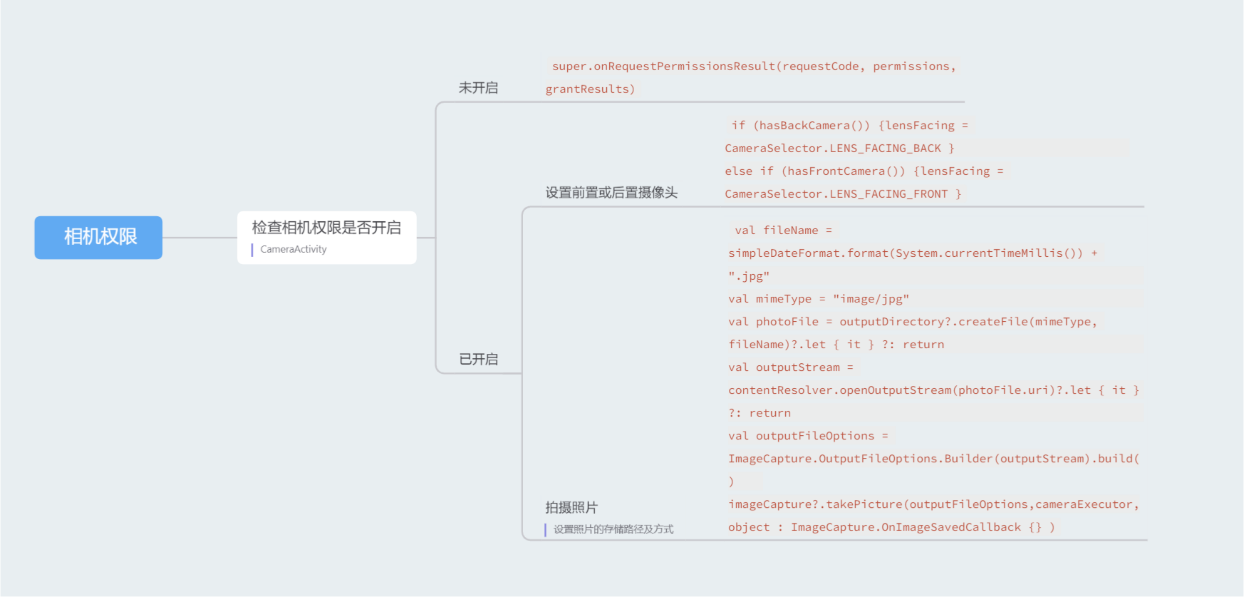
<uses-permission android:name="android.permission.CAMERA" ></uses-permission>

API：

import androidx.camera.core.*

import androidx.camera.lifecycle.ProcessCameraProvider

（2） Call diagram



```
super.onRequestPermissionsResult(requestCode, permissions,
grantResults)
```

```
if (hasBackCamera()) {lensFacing =
CameraSelector.LENS_FACING_BACK }
else if (hasFrontCamera()) {lensFacing =
CameraSelector.LENS_FACING_FRONT }
```

```
val fileName =
simpleDateFormat.format(System.currentTimeMillis()) +
".jpg"
val mimeType = "image/jpg"
val photoFile = outputDirectory?.createFile(mimeType,
fileName)?.let { it } ?: return
val outputStream =
contentResolver.openOutputStream(photoFile.uri)?.let { it }
?: return
val outputFileOptions =
ImageCapture.OutputFileOptions.Builder(outputStream).build(
)
imageCapture?.takePicture(outputFileOptions,cameraExecutor,
object : ImageCapture.OnImageSavedCallback {} )
```

4. Get contact permissions

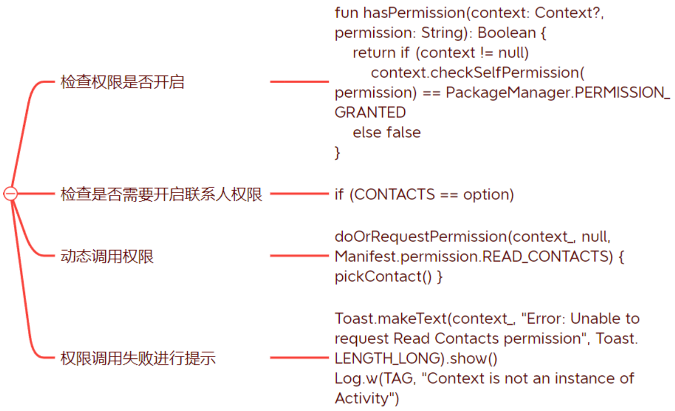
Reference open source software：

https://github.com/cyclestreets/android

（1）Permissions and corresponding APIs

Permissions： <uses-permission android:name="android.permission.READ_CONTACTS" /> API： import android.provider.ContactsContract;

（2） Call diagram



```
fun hasPermission(context: Context?,
permission: String): Boolean {
    return if (context != null)
        context.checkSelfPermission(
permission) == PackageManager.PERMISSION_
GRANTED
    else false
}
```

```
if (CONTACTS == option)
```

```
doOrRequestPermission(context_, null,
Manifest.permission.READ_CONTACTS) {
pickContact() }
```

```
Toast.makeText(context_, "Error: Unable to
request Read Contacts permission", Toast.
LENGTH_LONG).show()
Log.w(TAG, "Context is not an instance of
Activity")
```

5. Recording Permissions

Reference open source software：

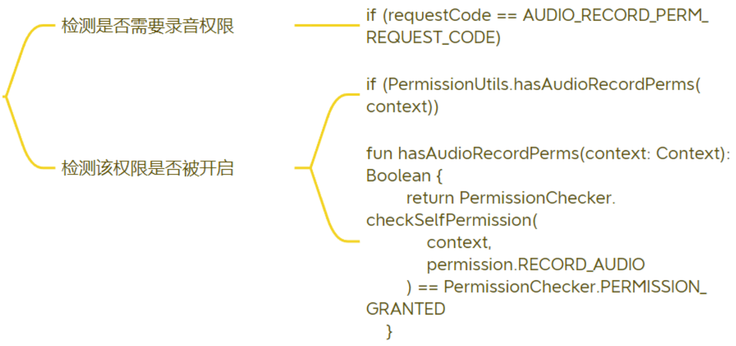
https://github.com/masuvern/barinsta

（1）Permissions and corresponding APIs

Permissions： <uses-permission android:name="android.permission.RECORD_AUDIO" /> API： import android.media.MediaRecorder

```
import android.provider.MediaStore
import android.media.MediaExtractor
import android.media.MediaFormat
import android.media.MediaMetadataRetriever
import android.media.MediaPlayer
```

（2） Call diagram



```
if (requestCode == AUDIO_RECORD_PERM_
REQUEST_CODE)
```

```
if (PermissionUtils.hasAudioRecordPerms(
context))

fun hasAudioRecordPerms(context: Context):
Boolean {
      return PermissionChecker.
checkSelfPermission(
         context,
         permission.RECORD_AUDIO
      ) == PermissionChecker.PERMISSION_
GRANTED
   }
```

6. Bluetooth Permissions

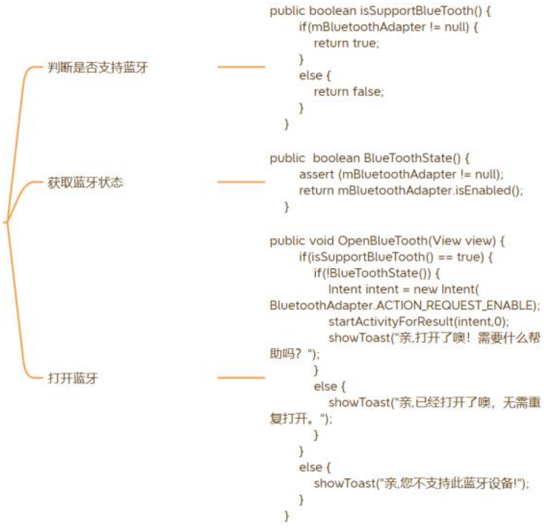
Reference open source software：

https://pan.baidu.com/s/19K2QCr1HIbAg2OBaJDCbNQ

（1）Permissions and corresponding APIs

```
Permissions：
<!-- Managing Bluetooth needs -->
<uses-permission android:name="android.permission.BLUETOOTH" />
<uses-permission android:name="android.permission.BLUETOOTH_ADMIN" />
<!-- Searching for Bluetooth is required, as Bluetooth can be used for location, so location
permission is needed -->
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION"/>
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION"/>
API：
import android.bluetooth.BluetoothAdapter;
import android.bluetooth.BluetoothDevice;
import android.bluetooth.BluetoothSocket;
```

（2）Call diagram



```
public boolean isSupportBlueTooth() {
      if(mBluetoothAdapter != null) {
         return true;
      }
      else {
         return false;
      }
   }

public  boolean BlueToothState() {
      assert (mBluetoothAdapter != null);
      return mBluetoothAdapter.isEnabled();
   }

public void OpenBlueTooth(View view) {
      if(isSupportBlueTooth() == true) {
         if(!BlueToothState()) {
            Intent intent = new Intent(
BluetoothAdapter.ACTION_REQUEST_ENABLE);
            startActivityForResult(intent,0);
            showToast("亲,打开了噢！ 需要什么帮
助吗？ ");
         }
         else {
            showToast("亲,已经打开了噢，无需重
复打开。");
         }
      }
      else {
         showToast("亲,您不支持此蓝牙设备!");
      }
   }
```

7. Location permission

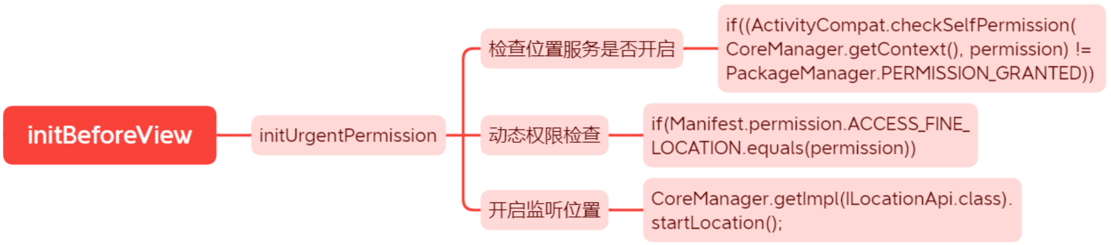
Reference open source software：

SilenceDut/KnowWeather: 一款美观、实用的天气 app。实践了模块化架构 和 Android Architecture Components (github.com)

（1）Permissions and corresponding APIs

Permissions：
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION"/>
API：
com.amap.api.location.AMapLocation;
com.amap.api.location.AMapLocationClient;
com.amap.api.location.AMapLocationClientOption;
com.amap.api.location.AMapLocationListener;

（2）Call diagram



```
if((ActivityCompat.checkSelfPermission(
CoreManager.getContext(), permission) !=
PackageManager.PERMISSION_GRANTED))

if(Manifest.permission.ACCESS_FINE_
LOCATION.equals(permission))

CoreManager.getImpl(ILocationApi.class).
startLocation();
```

8. Storage Permissions

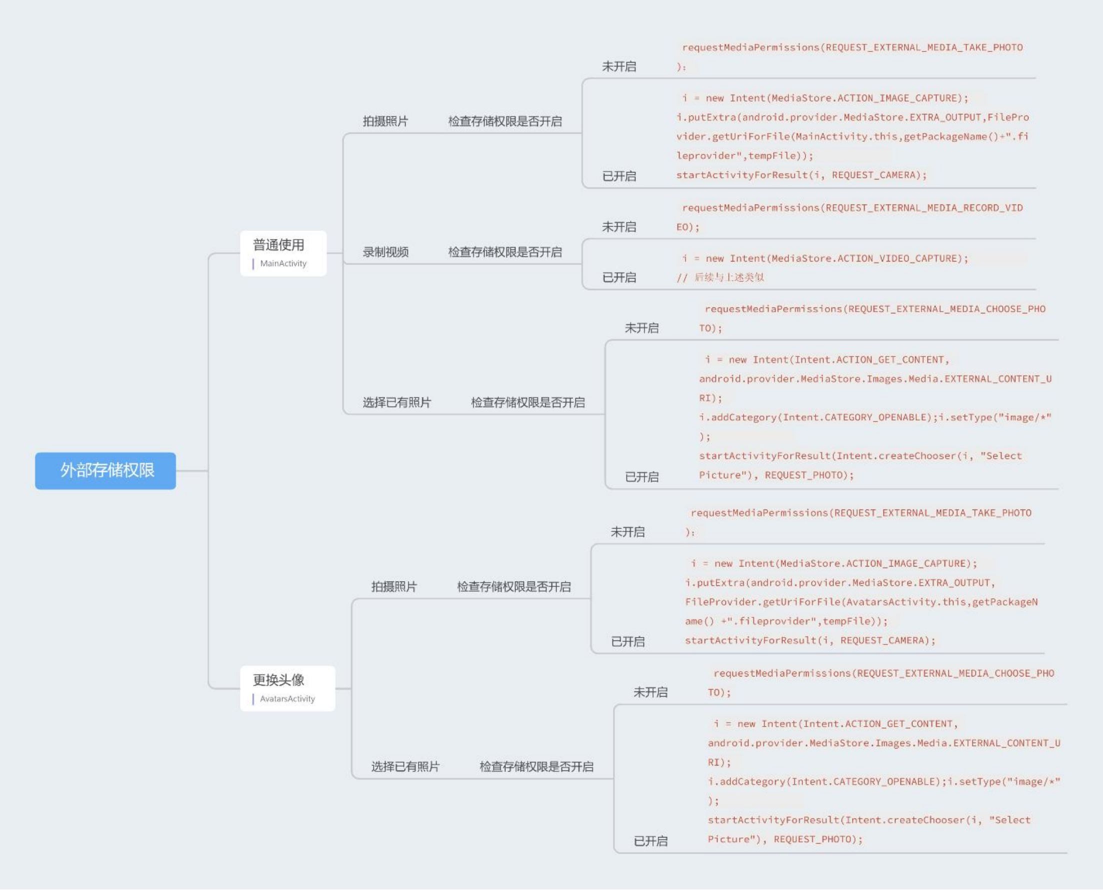
Reference open source software：

irccloud：https://github.com/irccloud/android

（1）Permissions and corresponding APIs

Permissions：
<uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE"></uses-permission>
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"></uses-permission>
API：
android.provider.MediaStore

（2）Call diagram



```
requestMediaPermissions(REQUEST_EXTERNAL_MEDIA_TAKE_PHOTO
);
```

```
i = new Intent(MediaStore.ACTION_IMAGE_CAPTURE);
i.putExtra(android.provider.MediaStore.EXTRA_OUTPUT,FilePro
vider.getUriForFile(MainActivity.this,getPackageName()+".fi
leprovider",tempFile));
startActivityForResult(i, REQUEST_CAMERA);
```

```
requestMediaPermissions(REQUEST_EXTERNAL_MEDIA_RECORD_VID
EO);
```

```
i = new Intent(MediaStore.ACTION_VIDEO_CAPTURE);
// 后续与上述类似
```

```
requestMediaPermissions(REQUEST_EXTERNAL_MEDIA_CHOOSE_PHO
TO);
```

```
i = new Intent(Intent.ACTION_GET_CONTENT,
android.provider.MediaStore.Images.Media.EXTERNAL_CONTENT_U
RI);
i.addCategory(Intent.CATEGORY_OPENABLE);i.setType("image/*"
);
startActivityForResult(Intent.createChooser(i, "Select
Picture"), REQUEST_PHOTO);
```

```
requestMediaPermissions(REQUEST_EXTERNAL_MEDIA_TAKE_PHOTO
);
```

```
i = new Intent(MediaStore.ACTION_IMAGE_CAPTURE);
i.putExtra(android.provider.MediaStore.EXTRA_OUTPUT,
FileProvider.getUriForFile(AvatarsActivity.this,getPackageN
ame() +".fileprovider",tempFile));
startActivityForResult(i, REQUEST_CAMERA);
```

```
requestMediaPermissions(REQUEST_EXTERNAL_MEDIA_CHOOSE_PHO
TO);
```

```
i = new Intent(Intent.ACTION_GET_CONTENT,
android.provider.MediaStore.Images.Media.EXTERNAL_CONTENT_U
RI);
i.addCategory(Intent.CATEGORY_OPENABLE);i.setType("image/*"
);
startActivityForResult(Intent.createChooser(i, "Select
Picture"), REQUEST_PHOTO);
```

2 Market app permission panel implementation method

1. The panel that pops up before applying

https://www.cnblogs.com/whycxb/p/9742385.html

https://blog.csdn.net/HHHceo/article/details/119767074

This can be achieved by dialog, adding the corresponding image resources in the res folder and setting the popup layout in detail in xml. After that, use the button click event settings to trigger this popup on the first click.

2. The panel that pops up at the top of the application

This can be achieved by DialogFragment, which sets Android's own popup window to be nested with the popup window above it. When the Android pop-up window pops up, it triggers the pop-up window above. The upper popup listens to the click event of the Android self popup button. When the click event is performed on the Android self popup button, the Android self popup disappears and also triggers the upper popup to disappear.