Android将本地图片转化成base64的方法

1. 获取本地图片地址：

|  |
| --- |
| **public static final int *REQUEST\_PICK\_IMAGE*** = 11101;  Intent intent; **if** (Build.VERSION.***SDK\_INT*** < 19) {  intent = **new** Intent(Intent.***ACTION\_GET\_CONTENT***);  intent.setType(**"image/\*"**);  } **else** {  intent = **new** Intent(Intent.***ACTION\_PICK***, android.provider.MediaStore.Images.Media.***EXTERNAL\_CONTENT\_URI***); } startActivityForResult(intent, ***REQUEST\_PICK\_IMAGE***);  **protected void** onActivityResult(**int** requestCode, **int** resultCode, @Nullable Intent data) {  **super**.onActivityResult(requestCode, resultCode, data);  **if** (resultCode == Activity.***RESULT\_OK***) {  **switch** (requestCode) {  **case *REQUEST\_PICK\_IMAGE***:  **if** (data != **null**) {  **realPathFromUri** = RealPathFromUriUtils.*getRealPathFromUri*(**this**, data.getData());  //获取到图片地址/storage/emulated/0/zjcmcc/Image/logo.png  Log.*i*(**"zjc"**, **realPathFromUri**);  String baseEncode = **""**;baseEncode = **"data:image/jpeg;base64,"**+*getImageStr*(**realPathFromUri**);  *// Log.i("zjcf", strBase64);* Log.*i*(**"zjcx"**, baseEncode);  *// 从相册返回的数据* **if** (data != **null**) {  *// 得到图片的全路径* Uri uri = data.getData();  **adduser\_tupian**.setImageURI(uri);  }  } **else** {  Toast.*makeText*(**this**, **"图片损坏，请重新选择"**, Toast.***LENGTH\_SHORT***).show();  }  **break**;  }  }  **public static** String getImageStr(String imgFile) {  File param = **new** File(imgFile);  Bitmap bitmap= BitmapFactory.*decodeFile*(param.getPath());  ByteArrayOutputStream bos=**new** ByteArrayOutputStream();  bitmap.compress(Bitmap.CompressFormat.***JPEG***, 40, bos);*//参数100表示不压缩* **byte**[] bytes=bos.toByteArray();   **return** Base64.*encodeToString*(bytes, Base64.***DEFAULT***);  }  **public class** RealPathFromUriUtils {  */\*\*  \* 根据Uri获取图片的绝对路径  \*  \** ***@param context*** *上下文对象  \** ***@param uri*** *图片的Uri  \** ***@return*** *如果Uri对应的图片存在, 那么返回该图片的绝对路径, 否则返回null  \*/* **public static** String getRealPathFromUri(Context context, Uri uri) {  **int** sdkVersion = Build.VERSION.***SDK\_INT***;  **if** (sdkVersion >= 19) { *// api >= 19* **return** *getRealPathFromUriAboveApi19*(context, uri);  } **else** { *// api < 19* **return** *getRealPathFromUriBelowAPI19*(context, uri);  }  }   */\*\*  \* 适配api19以下(不包括api19),根据uri获取图片的绝对路径  \*  \** ***@param context*** *上下文对象  \** ***@param uri*** *图片的Uri  \** ***@return*** *如果Uri对应的图片存在, 那么返回该图片的绝对路径, 否则返回null  \*/* **private static** String getRealPathFromUriBelowAPI19(Context context, Uri uri) {  **return** *getDataColumn*(context, uri, **null**, **null**);  }   */\*\*  \* 适配api19及以上,根据uri获取图片的绝对路径  \*  \** ***@param context*** *上下文对象  \** ***@param uri*** *图片的Uri  \** ***@return*** *如果Uri对应的图片存在, 那么返回该图片的绝对路径, 否则返回null  \*/* @SuppressLint(**"NewApi"**)  **private static** String getRealPathFromUriAboveApi19(Context context, Uri uri) {  String filePath = **null**;  **if** (DocumentsContract.*isDocumentUri*(context, uri)) {  *// 如果是document类型的 uri, 则通过document id来进行处理* String documentId = DocumentsContract.*getDocumentId*(uri);  **if** (*isMediaDocument*(uri)) { *// MediaProvider  // 使用':'分割* String id = documentId.split(**":"**)[1];   String selection = MediaStore.Images.Media.***\_ID*** + **"=?"**;  String[] selectionArgs = {id};  filePath = *getDataColumn*(context, MediaStore.Images.Media.***EXTERNAL\_CONTENT\_URI***, selection, selectionArgs);  } **else if** (*isDownloadsDocument*(uri)) { *// DownloadsProvider* Uri contentUri = ContentUris.*withAppendedId*(Uri.*parse*(**"content://downloads/public\_downloads"**), Long.*valueOf*(documentId));  filePath = *getDataColumn*(context, contentUri, **null**, **null**);  }  } **else if** (**"content"**.equalsIgnoreCase(uri.getScheme())) {  *// 如果是 content 类型的 Uri* filePath = *getDataColumn*(context, uri, **null**, **null**);  } **else if** (**"file"**.equals(uri.getScheme())) {  *// 如果是 file 类型的 Uri,直接获取图片对应的路径* filePath = uri.getPath();  }  **return** filePath;  }   */\*\*  \* 获取数据库表中的 \_data 列，即返回Uri对应的文件路径  \*  \** ***@return*** *\*/* **private static** String getDataColumn(Context context, Uri uri, String selection, String[] selectionArgs) {  String path = **null**;   String[] projection = **new** String[]{MediaStore.Images.Media.***DATA***};  Cursor cursor = **null**;  **try** {  cursor = context.getContentResolver().query(uri, projection, selection, selectionArgs, **null**);  **if** (cursor != **null** && cursor.moveToFirst()) {  **int** columnIndex = cursor.getColumnIndexOrThrow(projection[0]);  path = cursor.getString(columnIndex);  }  } **catch** (Exception e) {  **if** (cursor != **null**) {  cursor.close();  }  }  **return** path;  }   */\*\*  \** ***@param uri*** *the Uri to check  \** ***@return*** *Whether the Uri authority is MediaProvider  \*/* **private static boolean** isMediaDocument(Uri uri) {  **return "com.android.providers.media.documents"**.equals(uri.getAuthority());  }   */\*\*  \** ***@param uri*** *the Uri to check  \** ***@return*** *Whether the Uri authority is DownloadsProvider  \*/* **private static boolean** isDownloadsDocument(Uri uri) {  **return "com.android.providers.downloads.documents"**.equals(uri.getAuthority());  }  } |