Android programming

TX00CK66 Sensor Based Mobile

Applications

Lecture Camera and File Provider

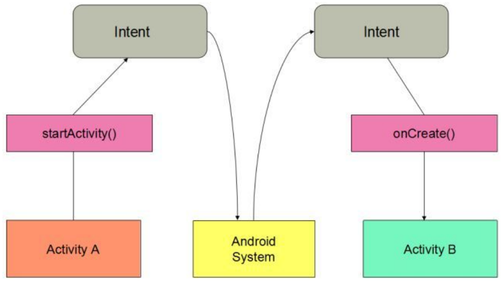
Jarkko.Vuori@metropolia.fi

Use camera app

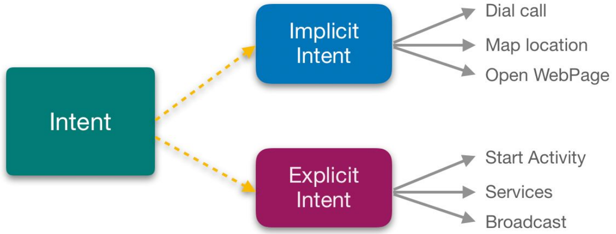
- We will use existing Camera Application (installed on your phone), not implementing our own camera app
- Use Intent to find a camera application to take a photo
- Either receive a thumbnail or create an image file, where to save the full-size photo

- Note: If you want to create your own camera app then there is a new camera API library – CameraX (based on Camera2 API)
  - Use this if you want to do real-time image analysis, or if you want to control the picture taking process (shutter speed, aperture, etc.)

Intents



- Intent is a message to someone
- Activities, services, and broadcast receivers are activated through intents
- An intent is an Intent object that holds the content of the message
  - For activities, it names the action being requested and specifies the URI of the data to act on, among other things
- One activity often starts the next one by calling `startActivity()` or `startActivityForResult()` if it expects a result back from the activity it's starting
- Two main types
  - Explicit - intent targeted to specific Activity
  - Implicit – intent targeted to any Activity that can handle it



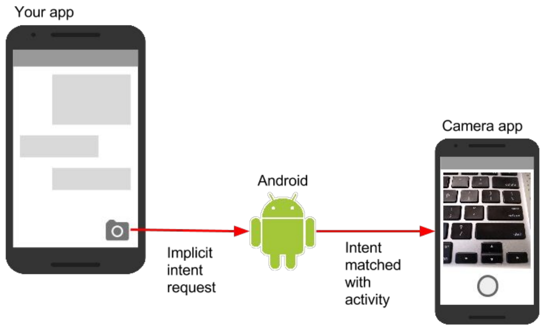
Get a thumbnail

```
val REQUEST_IMAGE_CAPTURE = 99
…
val myIntent = Intent(MediaStore.ACTION_IMAGE_CAPTURE)
if (myIntent.resolveActivity(getPackageManager()) != null) {
            startActivityForResult(myIntent, REQUEST_IMAGE_CAPTURE)
}
…

override fun onActivityResult(requestCode: Int, resultCode: Int, recIntent: Intent?) {
        if (requestCode == REQUEST_IMAGE_CAPTURE && resultCode == Activity.RESULT_OK) {
            val extras = recIntent!!.extras
            val imageBitmap = extras!!.get("data") as Bitmap
         …
        }
    }
```

Check if suitable app is available

The Android Camera application encodes the photo in the return Intent delivered to onActivityResult() as a small Bitmap in the extras, under the key "data"



<uses-feature android:name="android.hardware.camera" android:required="true" />

Get a thumbnail in Jetpack Compose

- It is not possible to use Intents to start an Activity in Jetpack Compose
    - rememberLauncherForActivityResult() API allows you to get a result from an activity in your composable
- Check: https://developer.android.com/jetpack/compose/libraries
- Example Get a thumbnail (using default contract)

We have TakePicturePreview here. launch() starts the activity (Camera). It returns the bitmap as a parameter to the given lambda (it on Kotlin).

```
val result = remember { mutableStateOf<Bitmap?>(null) }
val launcher = rememberLauncherForActivityResult(ActivityResultContracts.TakePicturePreview()) {
   result.value = it
}
       .
       .
       .
   Button(onClick = { launcher.launch() }) {
       Text(text = "Take a picture")
   }
   Spacer(Modifier.height(16.dp))
   result.value?.let { image ->
       Image(image.asImageBitmap(), null, modifier = Modifier.fillMaxWidth())
   }
}
```

Returns an ActivityResultLauncher which you'll use to launch the other activity.

FileUriExposedException

- Since Android 7.0 (API 24) some file system permission were changed in order to improve security:
  - “In order to improve the security of private files, the private directory of apps targeting Android 7.0 or higher has restricted access (0700, read-write-execute rights only to the current application). This setting prevents leakage of metadata of private files, such as their size or existence.”
- Camera application saves the picture in a file if requested so
  - and it is different to your application, therefore access 0700 for the directory and file is not useful
- In practise: if you pass a file:/// URI outside your package domain through an Intent, then you will get a FileUriExposedException
- Solution: use File Provider

FileProvider

URI scheme

- subclass of ContentProvider
- allows secure file sharing through a content:// URI instead of file:/// URI
    - A content:// URI allows you to grant read and write access using temporary access permissions
        - permissions are available to the client app for as long as your Activity is active
    - A file:/// URI requires you to modify the file system permissions of the underlying file
        - permissions are available to any app, and remain in effect until you change them

Define FileProvider

- Add a <provider> element in manifest

```
<provider
      android:name="androidx.core.content.FileProvider"
      android:authorities="com.example.kari.fileprovider"
      android:exported="false"
      android:grantUriPermissions="true">
  <meta-data …
  </meta-data>
</provider>
```

specify an authority consisting of the app's android:package value with the string "fileprovider" appended to it

- Create a resource file defining allowed directories under res/xml/ subfolder, for example file_paths.xml
  - A FileProvider can only generate a content URI for files in directories that you specify beforehand

```
<paths xmlns:android="http://schemas.android.com/apk/res/android">
    <external-files-path name="my_images" path="Pictures" />
</paths>
```

- Link xml file to the FileProvider by adding a <meta-data> element as a child of the <provider> element

```
<meta-data
  android:name="android.support.FILE_PROVIDER_PATHS"
  android:resource="@xml/file_paths">
</meta-data>
```

This is the directory you want to give access

files in the root of app's external storage area
/storage/emulated/0/Android/data/fi.metropolia.kari.camera2/files/Pictures

Generate Content URI

- Create a file into app's external storage area:

```
val fileName = "temp_photo"
val imgPath = getExternalFilesDir(Environment.DIRECTORY_PICTURES)
val imageFile = File.createTempFile(fileName, ".jpg", imgPath )
```

Same directory as defined in xml-file

- Create Content URI

```
val photoURI: Uri = FileProvider.getUriForFile(this,
                "com.example.kari.fileprovider",
                imageFile)
val currentPhotoPath = imageFile!!.absolutePath
```

Absolute path for BitmapFactory.decodeFile(currentPhotoPath)

Same as android:authorities' content in provider element in manifest

```
<uses-feature android:name="android.hardware.camera" android:required="true" />
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
```

Using Activity Result APIs

```
val result = remember { mutableStateOf<Bitmap?>(null) }
val launcher = rememberLauncherForActivityResult (ActivityResultContracts .TakePicture()) {
   if (it)
       result.value = BitmapFactory.decodeFile(currentP hotoPath)
   else
       Log.i("DBG", "Picture not taken")
}
       .
       .
       .
launcher.launch(PhotoURI)
       .
       .
       .
```

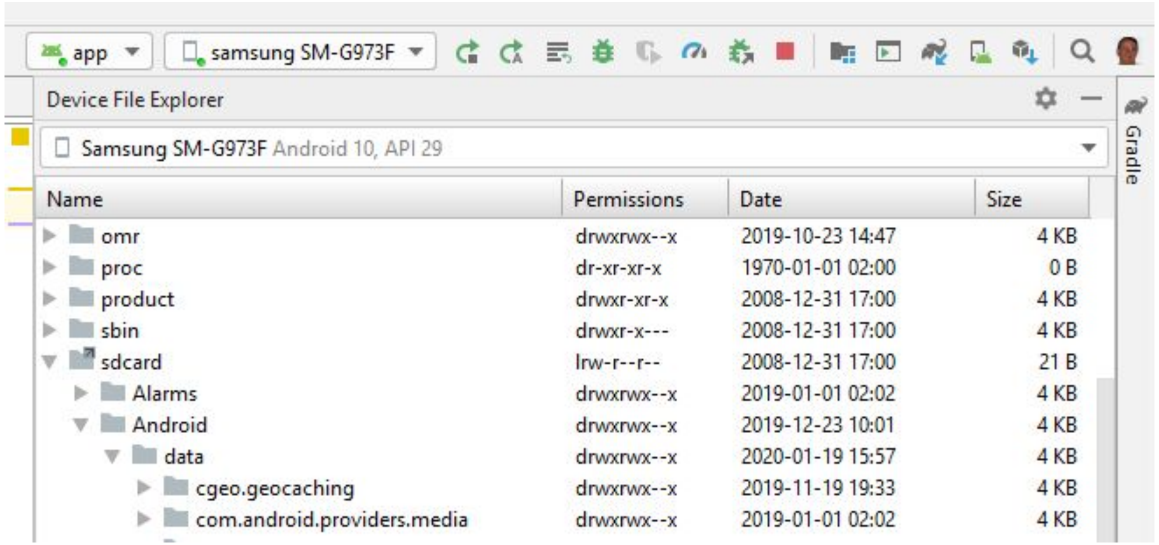
Now we have TakePicture here. launch(String path) starts the activity (Camera). It takes as an argument the absolute file path (photoURI) where the picture will be stored

File paths

- files-path (files in the files/ subdirectory of your **app's internal storage** area) - getFilesDir()
  - xml meta data: `<files-path name="my_images" path="Pictures/" />`
  - program code: `val ip = File(getFilesDir(),"Pictures")`
  - Device File Explorer / emulator: \data\app\data\app-package\files\Pictures\
- external-files-path (files in the root of your **app's external storage** area) - getExternalFilesDir(String)
  - xml meta data: `<external-files-path name="my_images" path="Pictures/" />`
  - program code: `val ip = getExternalFilesDir(Environment.DIRECTORY_PICTURES)`
  - Device File Explorer / emulator: \sdcard\Android\data\app_package\files\Pictures\

Find saved file

- Android Studio / Device File Explorer



Samsung SM-G973F Android 10, API 29

Name	Permissions	Date	Size
omr	drwxrwx--x	2019-10-23 14:47	4 KB
proc	dr-xr-xr-x	1970-01-01 02:00	0 B
product	drwxr-xr-x	2008-12-31 17:00	4 KB
sbin	drwxr-x---	2008-12-31 17:00	4 KB
sdcard	lrw-r--r--	2008-12-31 17:00	21 B
Alarms	drwxrwx--x	2019-01-01 02:02	4 KB
Android	drwxrwx--x	2019-12-23 10:01	4 KB
data	drwxrwx--x	2020-01-19 15:57	4 KB
cgeo.geocaching	drwxrwx--x	2019-11-19 19:33	4 KB
com.android.providers.media	drwxrwx--x	2019-01-01 02:02	4 KB

Since Android 11, it is not possible to access different application’s files at all

Reading list

- https://developer.android.com/training/camera/photobasics
- https://developer.android.com/training/basics/intents/result
- https://developer.android.com/training/camerax

- https://developer.android.com/training/secure-file-sharing/setup-sharing
- https://developer.android.com/reference/kotlin/androidx/core/content/FileProvider

- https://www.devbitsandbytes.com/configuring-camerax-in-jetpack-compose-to-take-picture/