Intelligent Devices and Cloud Computing

Camera

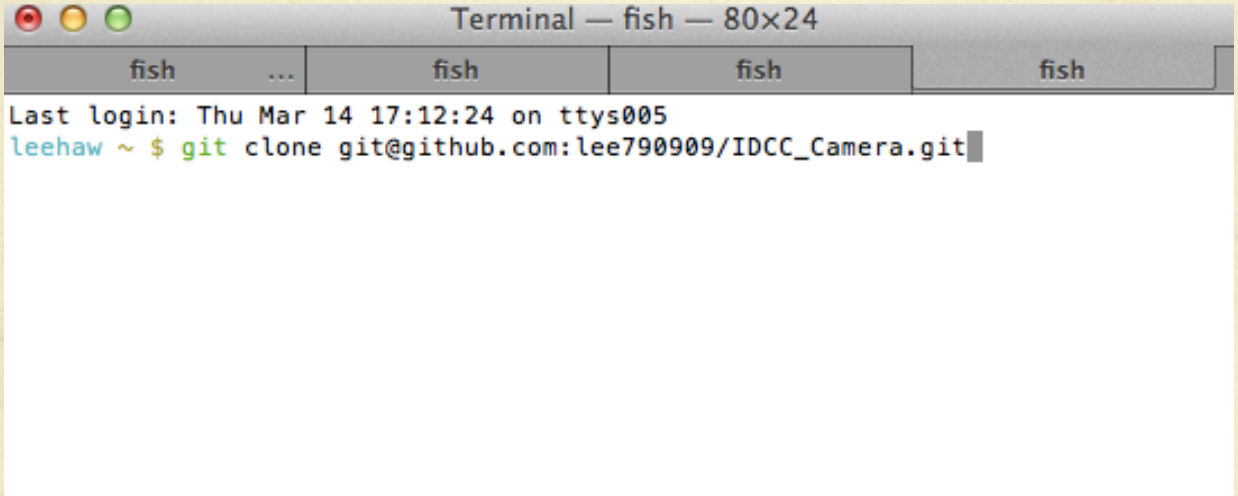
R01921043 李鎬

Outline

- Introduction
- Create a camera app
  - Camera Intent
  - Camera API
- Register this app as a “camera app”
  - Intent filter
- Reference

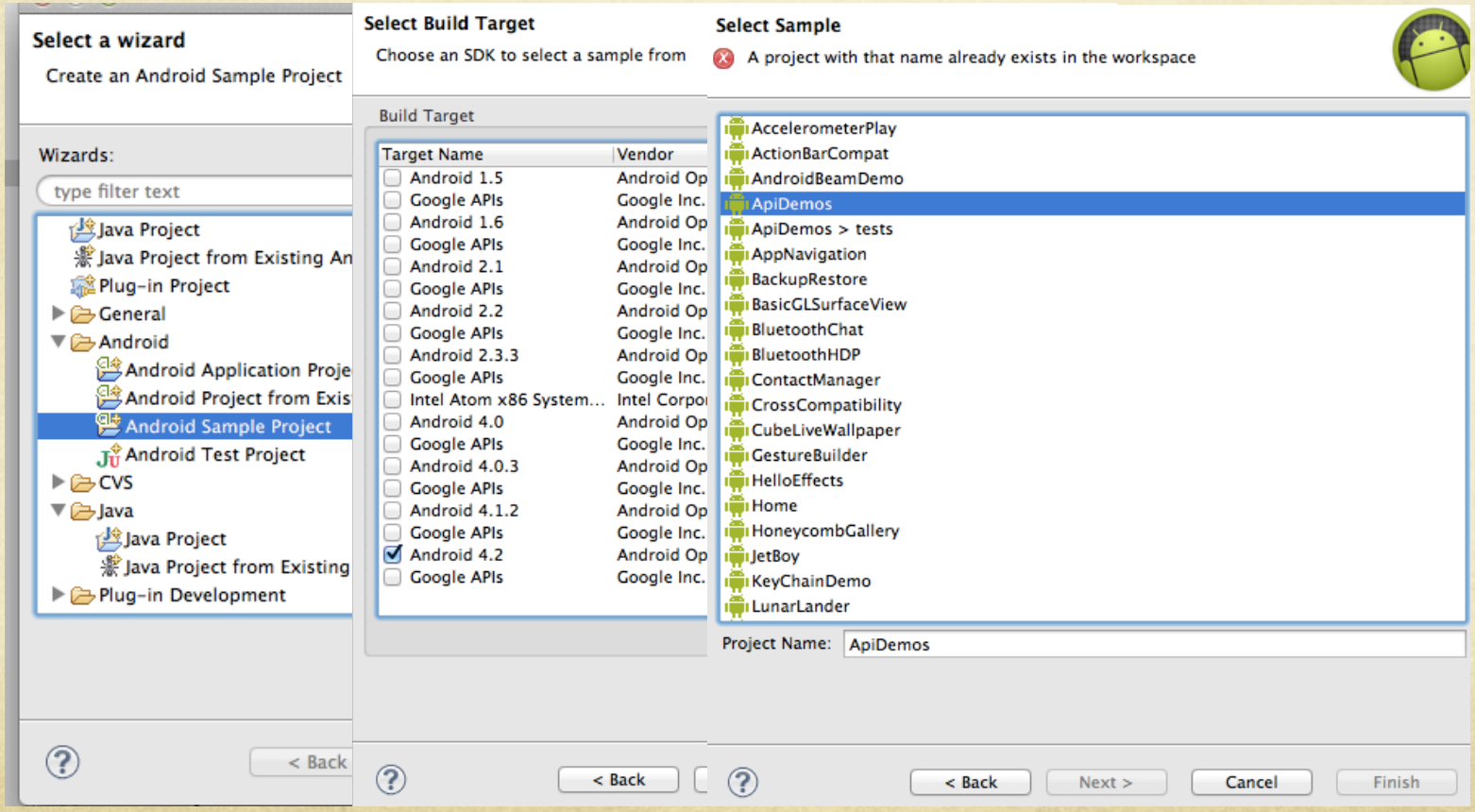
Source

- https://github.com/lee790909/IDCC_Camera



Introduction

- Camera preview in ApiDemos:



Introduction

- Graphics -> CameraPreview

Create a Camera App

- Reference: Android developer
  - **Guides**
    - Media and Camera
  - **Training**
    - Building Apps with Multimedia
      - Controlling the Camer
- Camera Intent
- Camera API

Camera Intent

- Compose a camera intent

```
case R.id.main_button_intent:
    intent = new Intent(MediaStore.ACTION_IMAGE_CAPTURE);
    cameraOutputUri = PictureFiles.getOutputMediaFileUri(
            PictureFiles.MEDIA_TYPE_IMAGE);
    intent.putExtra(MediaStore.EXTRA_OUTPUT, cameraOutputUri);
    requestCode = AR_IMAGE_FROM_CAMERA;
    break;
```

  - `intent.putExtra(name, data)`
  - Remember to use `requestCode` in order to receive the intent result.

Camera Intent

- Start the camera intent:

```
if (intent != null) {
    startActivityForResult(intent, requestCode);
}
```

- Receive the intent result

```
@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
    switch (requestCode) {
    case AR_IMAGE_FROM_CAMERA:
        if (resultCode == RESULT_OK) {
            if (cameraOutputUri != null)
                Toast.makeText(this, "saved photo in: " + cameraOutputUri.getPath(),
                        Toast.LENGTH_LONG).show();
        } else {
            Toast.makeText(this, "Oops! Some errors occur!! (camera via intent)",
                    Toast.LENGTH_LONG).show();
        }
        break;
```

Camera Intent

Camera Intent

- In Android Developer Guide:

```
private static final int CAPTURE_IMAGE_ACTIVITY_REQUEST_CODE = 100;
@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
    if (requestCode == CAPTURE_IMAGE_ACTIVITY_REQUEST_CODE) {
        if (resultCode == RESULT_OK) {
            // Image captured and saved to fileUri specified in the Intent
            Toast.makeText(this, "Image saved to:\n" +
                     data.getData(), Toast.LENGTH_LONG).show();
        } else if (resultCode == RESULT_CANCELED) {
            // User cancelled the image capture
        } else {
            // Image capture failed, advise user
        }
    }
    startActivityForResult(intent, CAPTURE_IMAGE_ACTIVITY_REQUEST_CODE);
}
```

Camera Intent

- `NullPointerException` in `onActivityResult#data.getData()`

```
private static final int CAPTURE_IMAGE_ACTIVITY_REQUEST_CODE = 100;
@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
    if (requestCode == CAPTURE_IMAGE_ACTIVITY_REQUEST_CODE) {
        if (resultCode == RESULT_OK) {
            // Image captured and saved to fileUri specified in the Intent
            Toast.makeText(this, "Image saved to:\n" +
                     data.getData(), Toast.LENGTH_LONG).show();
        } else if (resultCode == RESULT_CANCELED) {
            // User cancelled the image capture
        } else {
            // Image capture failed, advise user
        }
    }
    startActivityForResult(intent, CAPTURE_IMAGE_ACTIVITY_REQUEST_CODE);
}
```

Camera Intent

```
Intent intent = null;
int requestCode = -1;
switch (v.getId()) {
case R.id.main_button_intent:
    intent = new Intent(MediaStore.ACTION_IMAGE_CAPTURE);
    cameraOutputUri = PictureFiles.getOutputMediaFileUri(
            PictureFiles.MEDIA_TYPE_IMAGE);
    intent.putExtra(MediaStore.EXTRA_OUTPUT, cameraOutputUri);
    requestCode = AR_IMAGE_FROM_CAMERA;
    break;
```

```
@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
    switch (requestCode) {
    case AR_IMAGE_FROM_CAMERA:
        if (resultCode == RESULT_OK) {
            if (cameraOutputUri != null)
                Toast.makeText(this, "saved photo in: " + cameraOutputUri.getPath(),
                        Toast.LENGTH_LONG).show();
        } else {
            Toast.makeText(this, "Oops! Some errors occur!! (camera via intent)",
                    Toast.LENGTH_LONG).show();
        }
        break;
```

Camera Intent

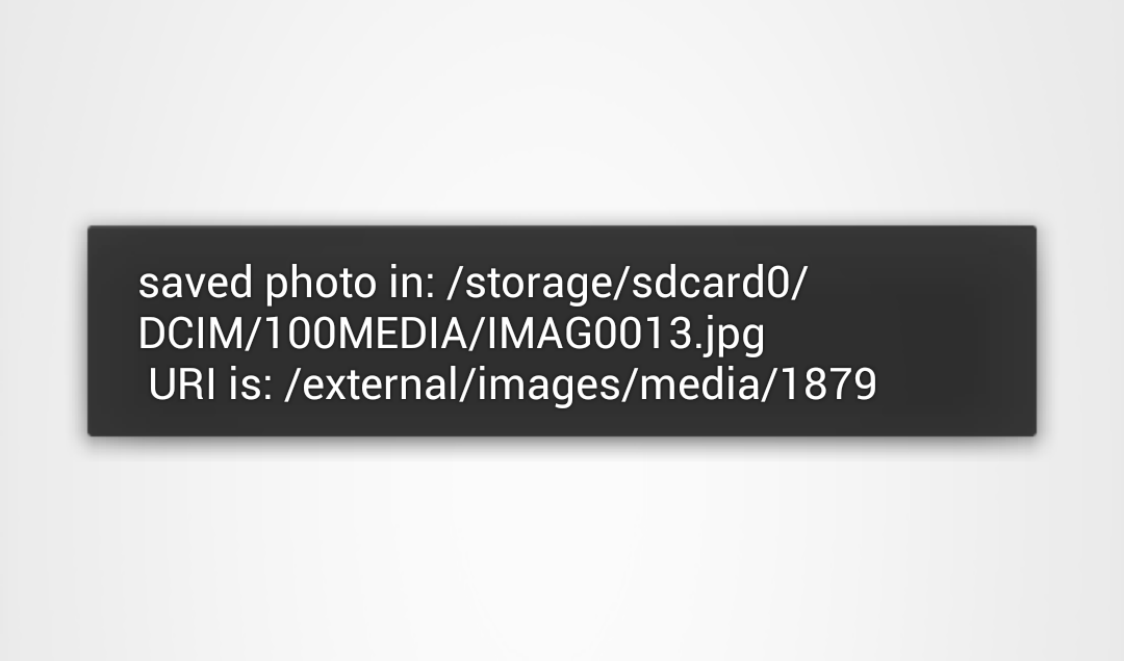
- If we don't use `Intent.putExtra(name, data)`
  - Save this photo in the "default album".
  - `data.getdata()` works.

```
public String getRealPathFromURI(Uri contentUri) {
    String[] proj = { MediaStore.Images.Media.DATA };
    Cursor cursor = managedQuery(contentUri, proj, null, null, null);
    int column_index = cursor.getColumnIndexOrThrow(MediaStore.Images.Media.DATA);
    cursor.moveToFirst();
    return cursor.getString(column_index);
}
```

  - `managedQuery` is deprecated in API level 11
  - Use `CursorLoader` instead when API level > 11

```
if (resultCode == RESULT_OK) {
    Toast.makeText(this, "saved photo in: " +
            getRealPathFromURI(data.getData()), Toast.LENGTH_LONG).show();
}
```

Camera Intent



Camera API

- Detect and access camera
- Create a preview class
- Build a preview layout
- Setup a listener for capture
- Capture and save files
- Release the camera

Camera API

- Require `<user-permission>`
  - In `AndroidManifest.xml`

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

  - Camera
    - If we are using camera via intent, we don't need this permission.
  - Write external storage
    - For saving files.

Detect and Access Camera

- Open a new `activity`.

```
/** Check if this device has a camera */
private boolean checkCameraHardware(Context context) {
    if (context.getPackageManager().hasSystemFeature(PackageManager.FEATURE_CAMERA)){
        // this device has a camera
        return true;
    } else {
        //
        re
    }
}
```

```
/** A safe way to get an instance of the Camera object. */
public static Camera getCameraInstance(){
    Camera c = null;
    try {
        c = Camera.open(); // attempt to get a Camera instance
    }
    catch (Exception e){
        // Camera is not available (in use or does not exist)
    }
    return c; // returns null if camera is unavailable
}
```

Detect and Access Camera

- Camera preview in API Demos

```
// Find the total number of cameras available
numberOfCameras = Camera.getNumberOfCameras();

// Find the ID of the default camera
CameraInfo cameraInfo = new CameraInfo();
    for (int i = 0; i < numberOfCameras; i++) {
        Camera.getCameraInfo(i, cameraInfo);
        if (cameraInfo.facing == CameraInfo.CAMERA_FACING_BACK) {
            defaultCameraId = i;
        }
    }
```

- Crash!!
  - Because there is no back-camera on Nexus 7.

Create a Preview Class

- Create a `SurfaceView` class

```
public class CameraPreviewSurfaceView extends SurfaceView implements SurfaceHolder.Callback {
```

Build the Preview Layout

- The Android Developer guide:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="horizontal"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
```

```
@Override
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.main);

    // Create an instance of Camera
    mCamera = getCameraInstance();

    // Create our Preview view and set it as the content of our activity.
    mPreview = new CameraPreview(this, mCamera);
    FrameLayout preview = (FrameLayout) findViewById(R.id.camera_preview);
    preview.addView(mPreview);
}
```

Build the Preview Layout

- Just define a custom view in the xml file.

```
<hpc.idcc.cameraapp.CameraPreviewSurfaceView
    android:id="@+id/camera_preview_surfaceview"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    />
```

Listener for Capture

- The listener of camera is `PictureCallback`

```
private PictureCallback mPicture = new PictureCallback() {

    @Override
    public void onPictureTaken(byte[] data, Camera camera) {
        mPhotoData = data;
        mConfirmButton.setVisibility(View.VISIBLE);
        mCancelButton.setVisibility(View.VISIBLE);
    }
};
```

Save Files

- Use `FileOuputStream`

```
FileOutputStream fos = new FileOutputStream(pictureFile);
fos.write(data);
fos.close();
```

- Saving file may take a long time and block UI thread.

Save Files

- AsyncTask and ProgressDialog

```
public class SavePhotoAsyncTask extends AsyncTask<byte[], Void, Throwable>
```

```
public SavePhotoAsyncTask(CameraPreviewActivity activity, Uri uri) {
    mListener = activity;
    ctx = activity;
    mUri = uri;
}
```

```
@Override
protected void onPreExecute() {
    super.onPreExecute();
    mDialog = ProgressDialog.show(ctx, null, "saving...");
}
```

```
case R.id.camera_preview_confirm:
    SavePhotoAsyncTask mAsyncTask = new SavePhotoAsyncTask(this,
            Uri.parse(fileUri));
    mAsyncTask.execute(mPhotoData);
    break;
```

Save Files

```
@Override
protected Exception doInBackground(byte[]... params) {
    if (mUri == null) {
        mUri = Uri.parse(PictureFiles.getOutputMediaFile(PictureFiles.MEDIA_TYPE_IMAGE).getPath());
        return new IOException("file is null");
    }

    if (params == null || params[0] == null) {
        return new NullPointerException("data is null");
    }

    try {
        OutputStream fos = ctx.getContentResolver().openOutputStream(mUri);
        fos.write(params[0]);
        fos.close();
    } catch (FileNotFoundException e) {
        // (TODO) handle file not found error.
    } catch (IOException e) {
        // (TODO) handle accessing file error.
    }
    return null;
}
```

Save Files

```
public interface SavePhotoListener {
    public void onSaveFinish(Throwable e, Uri uri);
}
```

```
@Override
protected void onPostExecute(Throwable result) {
    super.onPostExecute(result);
    mDialog.cancel();
    if (result == null)
        mListener.onSaveFinish(result, mUri);
    else
        mListener.onSaveFinish(result, null);
}
```

Save Files

```
public class CameraPreviewActivity extends Activity implements OnClickListener, SavePhotoListener
```

```
@Override
public void onSaveFinish(Throwable e, Uri uri) {
    Intent intent = new Intent();
    if (e == null) {
        intent.setData(uri);
        intent.putExtra(MediaStore.EXTRA_OUTPUT, uri);
        setResult(RESULT_OK, intent);
    } else {
        setResult(RESULT_CANCELED, intent);
        Toast.makeText(this, e.toString(), Toast.LENGTH_SHORT).show();
    }
    finish();
}
```

Release Camera

```
@Override
protected void onPause() {
    super.onPause();

    // Because the Camera object is a shared resource, it's very
    // important to release it when the activity is paused.
    if (mCamera != null) {
        mPreview.setCamera(null);
        mCamera.release();
        mCamera = null;
    }
}
```

Register a “Camera App”

- Show our app in the list.

Intent Filter

```
<activity
    android:name="CameraPreviewActivity"
    android:screenOrientation="portrait">
    <intent-filter>
        <category android:name="android.intent.category.DEFAULT"/>
        <action android:name="android.media.action.IMAGE_CAPTURE"/>
    </intent-filter>
</activity>
```

Register a “Camera App”

- Though it doesn’t have a standard about intent filter.
- Follow the built-in app!

```
@Override
public void onSaveFinish(Throwable e, Uri uri) {
    Intent intent = new Intent();
    if (e == null) {
        intent.setData(uri);
        // Bundle bundle = new Bundle();
        // bundle.putByteArray("data", getBitmap());
        // intent.putExtras(bundle);
        setResult(RESULT_OK, intent);
    } else {
        setResult(RESULT_CANCELED, intent);
        Toast.makeText(this, e.toString(), Toast.LENGTH_SHORT).show();
    }
    finish();
}
```

- Some app use: `data.getExtras().get(“data”)` to get `Bitmap` directly.

Reference

- Android Developer
  - http://developer.android.com
- Stackoverflow
  - http://stackoverflow.com

Not Include...

- Camera feature
- Set camera parameters
- MediaScanner
- Video
- ...
- ...
- ...

Thanks