签到二维码

多媒体进阶-拍照与录像

字节跳动Android工程师　朱勤翔

ByteDance字节跳动

目录

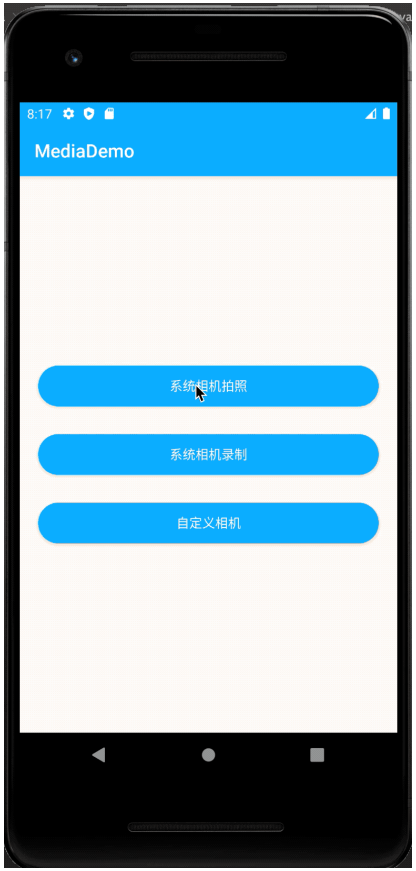
- 使用系统相机拍照
- 从图片到视频
- 使用系统相机录制
- 自定义相机拍照、录制
- 课后探索

第一章 使用系统相机拍照

效果展示

拍照流程：

- 调起系统相机
- 拍摄一张图片
- 使用ImageView控件显示照片



步骤一 调起系统相机

- AndroidManifest中声明权限

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

- 申请权限

```
1 ActivityCompat.requestPermissions(this, permissions,
  PERMISSION_REQUEST_CAMERA_CODE);
```

- 调起系统相机

```
1 Intent intent = new Intent(MediaStore.ACTION_IMAGE_CAPTURE);
2 startActivityForResult(intent, REQUEST_CODE_TAKE_PHOTO);
```

步骤二 接收数据

- 在onActivityResult回调方法中接收数据，拿到返回的bitmap，并显示在ImageView控件上

```
@Override
protected void onActivityResult(int requestCode, int resultCode, @Nullable
Intent data) {
    super.onActivityResult(requestCode, resultCode, data);
    if (requestCode == REQUEST_CODE_TAKE_PHOTO && resultCode == RESULT_OK) {
        Bundle extras = data.getExtras();
        Bitmap bitmap = (Bitmap) extras.get("data");
        imageView.setImageBitmap(bitmap);
    }
}
```

步骤二 接收数据

- 为什么图像这么小?

p imageBitmap = {Bitmap@12602} ""... View Bitmap
- f mCacheInfo = null
- f mColorSpace = null
- f mDensity = 480
- f mGalleryCached = false
- f mHeight = 228
- f mNativePtr = 502325909888
- f mNinePatchChunk = null
- f mNinePatchInsets = null
- f mRecycled = false
- f mReferenceCount = 0
- f mRequestPremultiplied = true
- f mWidth = 171
- f shadow$_klass_ = {Class@4169} "class android.graphics.Bitmap"... Navigate
- f shadow$_monitor_ = -2093064486

步骤三 解决问题-自定义存储路径（一）

- AndroidManifest中声明权限

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

- 申请权限

```
String[] permissions = new String[]{Manifest.permission.CAMERA};
ActivityCompat.requestPermissions(this, permissions,
PERMISSION_REQUEST_CAMERA_PATH_CODE);
```

- 创建文件

```
private String getOutputMediaPath() {
    File mediaStorageDir = getExternalFilesDir(Environment.DIRECTORY_PICTURES);
    String timeStamp = new SimpleDateFormat("yyyyMMdd_HHmmss").format(new
Date());
    File mediaFile = new File(mediaStorageDir, "IMG_" + timeStamp + ".jpg");
    if (!mediaFile.exists()) {
        mediaFile.getParentFile().mkdirs();
    }
    return mediaFile.getAbsolutePath();
}
```

步骤三 解决问题-自定义存储路径（二）

- 获取content:// URI，7.0以上手机不允许使用file:// URI跳出应用

```
<provider
    android:name="androidx.core.content.FileProvider"
    android:authorities="${applicationId}.fileprovider"
    android:exported="false"
    android:grantUriPermissions="true">
    <meta-data
        android:name="android.support.FILE_PROVIDER_PATHS"
        android:resource="@xml/filepaths" />
</provider>
```

- res/xml文件夹新建filepaths.xml

```
<?xml version="1.0" encoding="utf-8"?>
<paths xmlns:tools="http://schemas.android.com/tools"
tools:ignore="ResourceName">
    <external-files-path
        name="Pictures"
        path="." />
</paths>
```

步骤三 解决问题-自定义存储路径（三）

- 使用自定义存储路径启动系统相机

```
1 Intent intent = new Intent(MediaStore.ACTION_IMAGE_CAPTURE);
2 takeImagePath = getOutputMediaPath();
3 intent.putExtra(MediaStore.EXTRA_OUTPUT, getUriForFile(takeImagePath));
4 if (intent.resolveActivity(getPackageManager()) != null) {
5     startActivityForResult(intent, REQUEST_CODE_TAKE_PHOTO_PATH);
6 }
```

步骤四 显示图片

- 获取view的宽高
- 获取图片的宽高
- 计算缩放比例
- 获取bitmap
- 显示在ImageView上

```
//获取ImageView控件宽高
int targetWidth = imageView.getWidth();
int targetHeight = imageView.getHeight();
//创建Options,设置inJustDecodeBounds为true，只解码图片宽高信息
BitmapFactory.Options options = new BitmapFactory.Options();
options.inJustDecodeBounds = true;
BitmapFactory.decodeFile(takeImagePath, options);
int photoWidth = options.outWidth;
int photoHeight = options.outHeight;
//计算图片和控件的缩放比例，并设置给Options,然后inJustDecodeBounds置为false，解码真正的图片信息
int scaleFactor = Math.min(photoWidth / targetWidth, photoHeight / targetHeight);
options.inJustDecodeBounds = false;
options.inSampleSize = scaleFactor;
Bitmap bitmap = BitmapFactory.decodeFile(takeImagePath, options);
imageView.setImageBitmap(bitmap);
```

步骤四 显示效果

- 图片为什么旋转了
- 如何进行装正呢

步骤五 图片转正

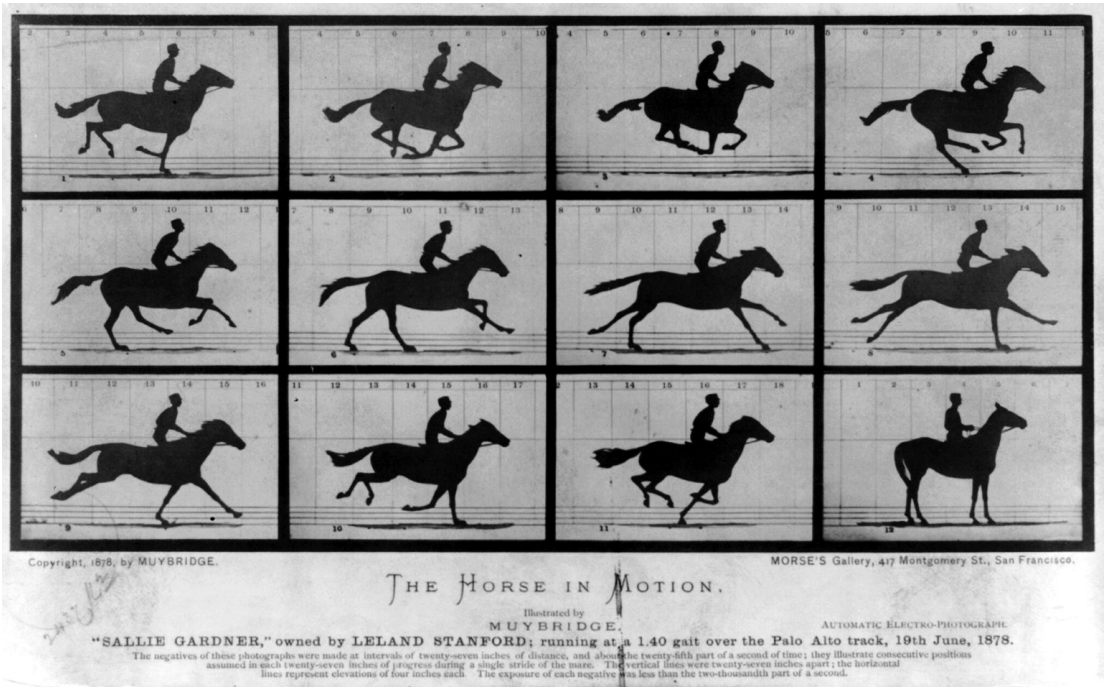
- 读取图片的旋转角度
- 在matrix中设置要旋转的角度
- 旋转图片

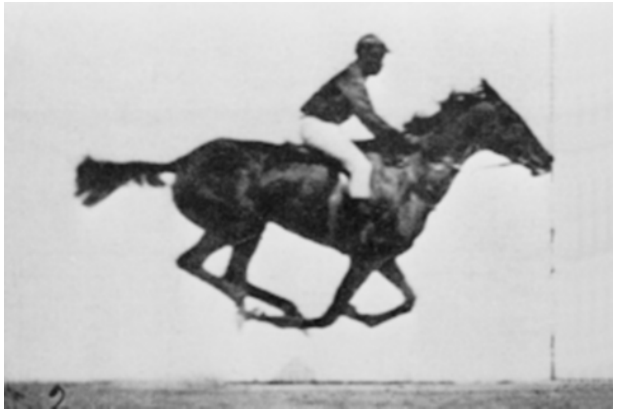
```
private Bitmap rotateImage(Bitmap bitmap,String path){
    try {
        ExifInterface srcExif = new ExifInterface(path);
        Matrix matrix = new Matrix();
        int angle = 0;
        int orientation =
srcExif.getAttributeInt(ExifInterface.TAG_ORIENTATION,ExifInterface.ORIENTATION_
NORMAL);
        switch (orientation){
            case ExifInterface.ORIENTATION_ROTATE_90:{
                angle = 90;
                break;
            }
            case ExifInterface.ORIENTATION_ROTATE_180:{
                angle = 180;
                break;
            }
            case ExifInterface.ORIENTATION_ROTATE_270:{
                angle = 270;
                break;
            }
            default:
                break;
        }
        matrix.postRotate(angle);
        return
Bitmap.createBitmap(bitmap,0,0,bitmap.getWidth(),bitmap.getHeight(),matrix,true)
;
    } catch (IOException e) {
        e.printStackTrace();
    }
    return null;
}
```

第二章 从图片到视频

电影的诞生

- 连续的图像
- 视觉暂留





电影的先驱：迈布里奇的马，1878

视频几个概念

❑ 帧率

每秒的帧数，画面帧率高于每秒10至12帧的时候，就会认为是连贯的，30帧以上就很流畅

❑ 分辨率

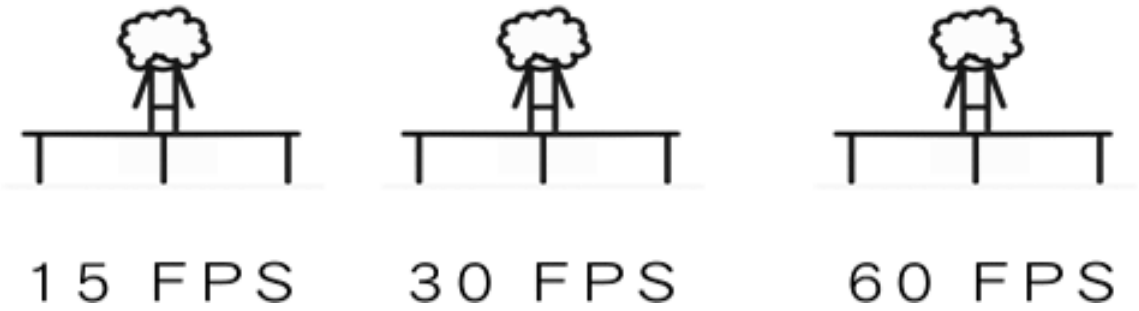
每帧图像的宽度和长度，例如 1080 * 1920

❑ 码率

每秒传送的比特(bit)数，用于视频压缩后得到一个期望的大小

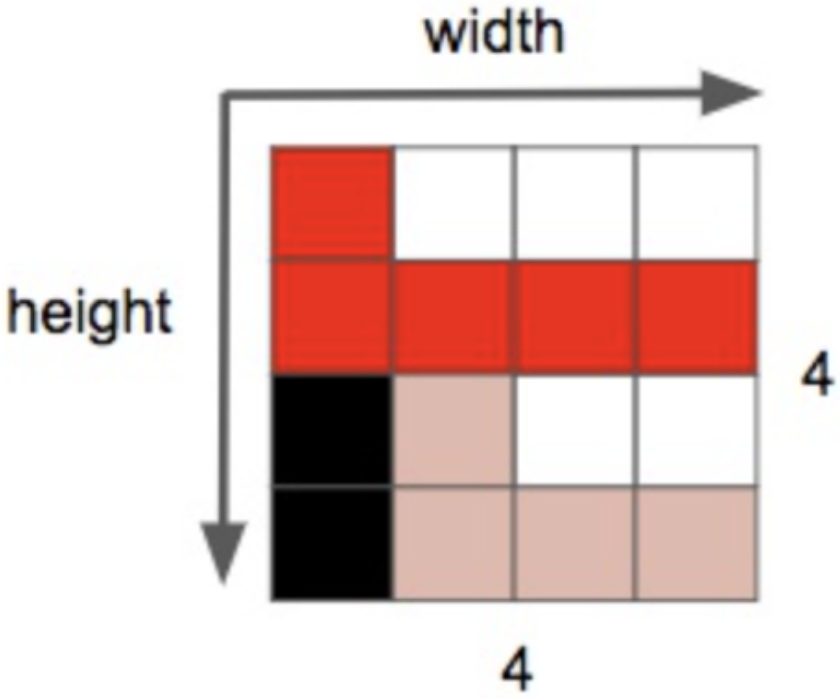
帧率

- 每秒钟的播放数量



分辨率

- 图像内数量的数量，通常使用宽*高表示



码率

- 每秒视频文件在单位时间内使用的数据流量，单位是kbps即千位每秒

- 视频大小 = duration 时长(s) x kbps 千位每秒/ 8 = xx MB

为什么要视频编码

- 分辨率1920*1080，帧率是30，未压缩情况下码率是多少?
- 每一帧是1920*1080*24=49766400bit
- 每一秒是49766400*30/8～=186.6MB
- 90分钟视频的大小大概是1000GB

视频几个概念

❑ 视频的编码

相邻图像之间有冗余，可以压缩成一个关键帧+变化差值

编码格式就是不同的压缩算法：MPEG/H264

❑ 视频的封装格式

把视频码流和音频码流按照一定的格式存储在一个文件中，与编码格式无关

第三章 最简单的录制

效果展示

- 申请相机、录音权限
- 调起手机相机录像
- 显示视频

步骤一 调起系统相机

- AndroidManifest中声明权限

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

- 申请权限

```
ActivityCompat.requestPermissions(this, permission.toArray(new
String[permission.size()]), PERMISSION_REQUEST_CODE);
```

- 调起相机的录像页面

```
Intent intent = new Intent(MediaStore.ACTION_VIDEO_CAPTURE);
mp4Path = getOutputMediaPath();
intent.putExtra(MediaStore.EXTRA_OUTPUT,PathUtils.getUriForFile(this,mp4Path));
intent.putExtra(MediaStore.EXTRA_VIDEO_QUALITY,1);
if (intent.resolveActivity(getPackageManager()) != null) {
    startActivityForResult(intent,REQUEST_CODE_RECORD);
}
```

步骤二 获取返回数据

- 获取拍摄的视频，并显示在页面上，开始播放

```
@Override
protected void onActivityResult(int requestCode, int resultCode, @Nullable
Intent data) {
    super.onActivityResult(requestCode, resultCode, data);
    if(requestCode == REQUEST_CODE_RECORD && resultCode == RESULT_OK){
        play();
    }
}
private void play() {
    mVideoView.setVideoPath(mp4Path);
    mVideoView.start();
}
```

步骤三 查看拍摄视频数据

- 视频的封装格式

  .mp4

- 视频的分辨率是多大?

  1080 * 1920

- 视频的文件大小和录制时长

  16.04MB /6秒

- 计算视频的码率（视频文件大小*8/时长）

  16.04 * 8 / 6 = 21.39 Mbps

第四章 自定义录制

效果展示

- 申请相机、麦克风权限
- 初始化相机
- 预览相机画面
- 拍照

步骤一 获取Camera实例

- AndroidManifest中声明权限

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

- 初始化Camera

```
private void initCamera() {
    mCamera = Camera.open();
    Camera.Parameters parameters = mCamera.getParameters();
    parameters.setPictureFormat(ImageFormat.JPEG);
    parameters.setFocusMode(Camera.Parameters.FOCUS_MODE_AUTO);
    parameters.set("orientation", "portrait");
    parameters.set("rotation", 90);
    mCamera.setParameters(parameters);
    mCamera.setDisplayOrientation(90);
}
```

步骤二 摄像头数据实时显示

- 使用SurfaceView显示

- 几个关键类

  Camera：操作和管理相机资源

  SurfaceView：用于绘制相机预览图像

  SurfaceHolder：可以控制Surface的尺寸、格式与像素

  SurfaceHolder.Callback：监听Surface状态变化的接口

```java
mSurfaceView = findViewById(R.id.surfaceview);
mHolder = mSurfaceView.getHolder();
mHolder.addCallback(this);
@Override
public void surfaceCreated(SurfaceHolder holder) {
    try {
        mCamera.setPreviewDisplay(holder);
        mCamera.startPreview();
    } catch (IOException e) {
        e.printStackTrace();
    }
}
@Override
public void surfaceChanged(SurfaceHolder holder, int format, int width, int height) {
    if (holder.getSurface() == null) {
        return;
    }
    //停止预览效果
    mCamera.stopPreview();
    //重新设置预览效果
    try {
        mCamera.setPreviewDisplay(holder);
        mCamera.startPreview();
    } catch (IOException e) {
        e.printStackTrace();
    }
}
@Override
public void surfaceDestroyed(SurfaceHolder holder) {
    mCamera.stopPreview();
    mCamera.release();
    mCamera = null;
}
```

步骤三 处理生命周期对预览的影响

```
@Override
protected void onResume() {
    super.onResume();
    if (mCamera == null) {
        initCamera();
    }
    mCamera.startPreview();
}

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

步骤四 拍摄一张实时照片

- 怎么用 camera api 拍照

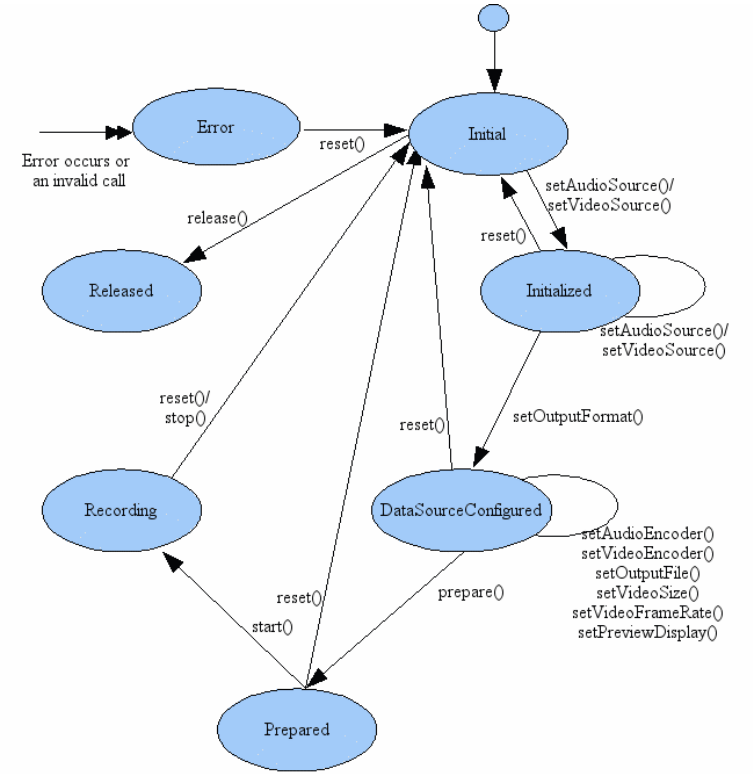
  mCamera.takePicture(null, null, mPicture)

- 拍照后继续预览

```
Camera.PictureCallback mPictureCallback = new Camera.PictureCallback() {
    @Override
    public void onPictureTaken(byte[] data, Camera camera) {
        FileOutputStream fos = null;
        String filePath =
getExternalFilesDir(Environment.DIRECTORY_PICTURES).getAbsolutePath() +
File.separator + "1.jpg";
        File file = new File(filePath);
        try {
            fos = new FileOutputStream(file);
            fos.write(data);
            fos.flush();
            Bitmap bitmap = BitmapFactory.decodeFile(filePath);
            Bitmap rotateBitmap = PathUtils.rotateImage(bitmap, filePath);
            mImageView.setVisibility(View.VISIBLE);
            mVideoView.setVisibility(View.GONE);
            mImageView.setImageBitmap(rotateBitmap);
        } catch (Exception e) {
            e.printStackTrace();
        } finally {
            mCamera.startPreview();
            if (fos != null) {
                try {
                    fos.close();
                } catch (IOException e) {
                    e.printStackTrace();
                }
            }
        }
    }
};
```

视频录制效果展示

步骤一 认识MediaRecorder



MediaRecorder state diagram

步骤二 准备录制（按部就班）

- Unlock the Camera
- Configure MediaRecorder
  - setCamera()
  - setAudioSource()
  - setVideoSource()
  - setProfile
  - setOutputFile()
  - setPreviewDisplay()
- Prepare MediaRecorder

```
1  private boolean prepareVideoRecorder() {
2      mMediaRecorder = new MediaRecorder();
3      // Step 1: Unlock and set camera to MediaRecorder
4      mCamera.unlock();
5      mMediaRecorder.setCamera(mCamera);
6      // Step 2: Set sources
7      mMediaRecorder.setAudioSource(MediaRecorder.AudioSource.CAMCORDER);
8      mMediaRecorder.setVideoSource(MediaRecorder.VideoSource.CAMERA);
9      // Step 3: Set a CamcorderProfile (requires API Level 8 or higher)
10
   mMediaRecorder.setProfile(CamcorderProfile.get(CamcorderProfile.QUALITY_HIGH));
11     // Step 4: Set output file
12     mp4Path = getOutputMediaPath();
13     mMediaRecorder.setOutputFile(mp4Path);
14     // Step 5: Set the preview output
15     mMediaRecorder.setPreviewDisplay(mHolder.getSurface());
16     mMediaRecorder.setOrientationHint(90);
17     // Step 6: Prepare configured MediaRecorder
18     try {
19         mMediaRecorder.prepare();
20     }catch (Exception e) {
21         releaseMediaRecorder();
22         return false;
23     }
24     return true;
25 }
```

步骤三 开始结束录制（按部就班）

- Start MediaRecorder
- Stop MediaRecorder
- Reset MediaRecorder
- Release MediaRecorder
- Lock the Camera
- 播放视频

```
1  public void record(View view) {
2      if (isRecording) {
3          mRecordButton.setText("录制");
4          mMediaRecorder.stop();
5          mMediaRecorder.reset();
6          mMediaRecorder.release();
7          mMediaRecorder = null;
8          mCamera.lock();
9
10         mVideoView.setVisibility(View.VISIBLE);
11         mImageView.setVisibility(View.GONE);
12         mVideoView.setVideoPath(mp4Path);
13         mVideoView.start();
14     } else {
15         if(prepareVideoRecorder()) {
16             mRecordButton.setText("暂停");
17             mMediaRecorder.start();
18         }
19     }
20     isRecording = !isRecording;
21 }
```

Camera VS Camera2

- Camera API 是 Android 生态碎片化最严重的一块
- Camera 厂商支持的较好，Camera2 各厂商支持情况各有好有差
- 今天课程内容是 Camera

课后练习一

- 使用系统相机拍照
  - 解决权限申请
  - 使用应用私有存储
  - 图片显示方向正确

课后练习二

- 使用系统相机录制视频
  - 解决权限申请
  - 使用应用私有存储
  - 相机拍摄后在页面上播放
  - 点击暂停，再次点击恢复播放

课后练习三

- 自定义拍照和录制视频
  - 使用 Camera、SurfaceView 实时显示摄像头图像
  - 使用 Camera 拍摄照片并展示
  - 使用 MediaRecorder 录制视频并展示

第五章 课后探索

参考文献

- https://developer.android.com/training/camera/videobasics
- https://developer.android.com/guide/topics/media/camera
- https://blog.csdn.net/feiduclear_up/article/details/51968975
- https://juejin.cn/post/6844903534610087943

作业上交

- 使用 github 托管你的项目
- 发邮件
  - 发给：zhuqinxiang@bytedance.com
  - 标题：浙大Android课设-多媒体进阶
  - 内容：你的姓名、学号和项目地址

THANKS

.