要开启Camera需要三个权限：

<uses-permission android:name="android.permission.CAMERA"/>  
<uses-feature android:name="android.hardware.Camera"/>  
<uses-feature android:name="android.hardware.camera.autofocus"/>

照相机的图像需要用SurfaceView来显示：

<SurfaceView  
 android:id="@+id/mCamera\_view"  
 android:layout\_width="300dp"  
 android:layout\_height="300dp" />

java代码：

要实现SurfaceHolder.Callback,Camera.PictureCallback接口。

cameraView = (SurfaceView) findViewById(R.id.mCamera\_view);  
cameraView.setFocusable(true);  
cameraView.setFocusableInTouchMode(true);  
cameraView.setClickable(true);

SurfaceHolder holder = cameraView.getHolder();  
holder.setType(SurfaceHolder.SURFACE\_TYPE\_PUSH\_BUFFERS);  
holder.addCallback(this);

SurfaceHolder.Callback,Camera.PictureCallback接口要实现的方法：

@Override  
 public void surfaceCreated(SurfaceHolder holder) {  
 camera = Camera.open(1);//1为打开前置摄像头  
 camera.setDisplayOrientation(90);//加了这一句，图像翻转的问题解决了  
 try {  
 Camera.Parameters param = camera.getParameters();  
// if (this.getResources().getConfiguration().orientation != //Configuration.ORIENTATION\_LANDSCAPE) {  
// param.set("orientation", "landscape");//landscape  
// } else {  
// param.set("orientation", "portrait");  
// }  
  
 List<String> colorEffects = param.getSupportedColorEffects();  
 Iterator<String> colorItor = colorEffects.iterator();  
 while (colorItor.hasNext()) {  
 String color = colorItor.next();  
 if (color.equals(Camera.Parameters.EFFECT\_SOLARIZE)) {  
 param.setColorEffect(Camera.Parameters.EFFECT\_SOLARIZE);  
 break;  
 }  
 }  
  
 camera.setParameters(param);  
  
 camera.setPreviewDisplay(holder);  
  
 int bestWidth = 0;  
 int bestHeight = 0;  
  
 List<Camera.Size> sizesList = param.getSupportedPreviewSizes();  
 if(sizesList.size() > 1){  
 Iterator<Camera.Size> itor = sizesList.iterator();  
 while(itor.hasNext()){  
 Camera.Size cur = itor.next();  
 if(cur.width > bestWidth && cur.height>bestHeight && cur.width <=MAX\_WIDTH && cur.height <= MAX\_HEIGHT){  
 bestWidth = cur.width;  
 bestHeight = cur.height;  
 }  
 }  
 if(bestWidth != 0 && bestHeight != 0){  
// param.setPreviewSize(bestWidth,bestHeight);  
// cameraView.setLayoutParams(new LinearLayout.LayoutParams(bestWidth,bestHeight));  
 }  
  
 }  
  
// camera.setParameters(param);  
  
 }catch (Exception e){  
 camera.release();  
 }  
  
 camera.startPreview();  
 }  
  
 @Override  
 public void surfaceChanged(SurfaceHolder holder, int format, int width, int height) {  
  
 }  
  
 @Override  
 public void surfaceDestroyed(SurfaceHolder holder) {  
 camera.stopPreview();  
 camera.release();  
 }  
  
 @Override  
 public void onPictureTaken(byte[] data, Camera camera) {  
 Uri imageUri = this.getContentResolver().  
 insert(MediaStore.Images.Media.EXTERNAL\_CONTENT\_URI,new ContentValues());  
 try {  
 OutputStream ops = this.getContentResolver().openOutputStream(imageUri);  
 ops.write(data);  
 ops.flush();  
 ops.close();  
 } catch (FileNotFoundException e) {  
 e.printStackTrace();  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 camera.startPreview();  
 }