Android Programing

Gavrilut Dragos

Camera

- Hardware component that allows one to take picture / record videos / audio stream / preview and so on.
- It can be access indirectly via intent or directly through te Camera object. If access directly the following permissions must be added:
  - android.permission.CAMERA
  - android.hardware.camera
  - android.permission.RECORD_AUDIO (optional if you want to perform audio recording).

Check and Access Camera Object

```
@SuppressLint("NewApi")
public int GetCamerasCount()
{
    if (this.getPackageManager().hasSystemFeature(PackageManager.FEATURE_CAMERA))
    {
        if (android.os.Build.VERSION.SDK_INT >= 9)
            return Camera.getNumberOfCameras();
        else
            return 1;
    } else {
        return 0;
    }
}
public Camera GetCamera(int cameraID)
{
    int cameraCount = GetCamerasCount();
    if (cameraCount <= 0)
        return null;
    try {
        if (cameraCount == 1)
            return Camera.open();
        else
            return Camera.open(cameraID);
    }
    catch (Exception e) {
        return null;
    }
}
```

Camera usage flow

- Obtain an instance of Camera
- Preview
- Take a picture / video
- Be carefull with system events (Pause / Terminate). Camera object should be released (with Camera.release() API)

Camera Preview

```
public class CameraPreviewSurface extends SurfaceView implements SurfaceHolder.Callback {
    private SurfaceHolder surfaceHolder;
    private Camera cameraObject;
    public CameraPreviewSurface(Context context, Camera camera) {
        super(context);
        cameraObject = camera;
        surfaceHolder = getHolder();
        surfaceHolder.addCallback(this);
        surfaceHolder.setType(SurfaceHolder.SURFACE_TYPE_PUSH_BUFFERS);
    }
    public void surfaceCreated(SurfaceHolder holder) {
        try {
            cameraObject.setPreviewDisplay(holder);
            cameraObject.startPreview();
        } catch (Exception e) { … }
    }
    public void surfaceDestroyed(SurfaceHolder holder) {
        try {
            cameraObject.stopPreview();
        } catch (Exception e){ … }
    }
    public void surfaceChanged(SurfaceHolder holder, int format, int w, int h) {
        if (surfaceHolder.getSurface() == null){
          return;
        }
        try {
            cameraObject.stopPreview();
        } catch (Exception e) { … }
        try {
            cameraObject.setPreviewDisplay(surfaceHolder);
            cameraObject.startPreview();
        } catch (Exception e) { … }
    }
}
```

Camera TakePicture

```
public class CameraPreviewSurface extends SurfaceView implements SurfaceHolder.Callback {
    private SurfaceHolder surfaceHolder;
    private Camera cameraObject;

    private String pictureFileName;
    private PictureCallback pictureCallbackObject = new PictureCallback() {
        @Override
        public void onPictureTaken(byte[] data, Camera camera) {

            File pictureFile = new File(pictureFileName);
            if (pictureFile == null)
                return;

            try {
                FileOutputStream fos = new FileOutputStream(pictureFile);
                fos.write(data);
                fos.close();
            } catch (Exception e) { }
        }
    };


    public void TakePicture(String fileName)
    {
        if (cameraObject!=null)
        {
            pictureFileName = fileName;
            cameraObject.takePicture(null, null, pictureCallbackObject);
        }
    }
}
```

Camera Features

- Can be set/read using Camera.Parameters object
- A normal flow for setting the Camera parameters can be done in the following way:
  - Obtain a Camera.Parameters from the current Camera object
  - Set the parameters using the newly obtain object
  - Apply that object to the camera

```
public void SetCameraParameters(Camera cameraObject)
{
    Camera.Parameters params = cameraObject.getParameters();
    // ...
    List<Size> previewSizes = params.getSupportedPreviewSizes();
    // ....
    params.setPreviewSize(previewSizes.get(0).width, previewSizes.get(0).width);
    // ...
    cameraObject.setParameters(params);
}
```

Camera Features

Feature	API	Description
Face Detection	14	Identify human faces within a picture and use them for focus, metering and white balance
Metering Areas	14	Specify one or more areas within an image for calculating white balance
Focus Areas	14	Set one or more areas within an image to use for focus
White Balance Lock	14	Stop or start automatic white balance adjustments
Exposure Lock	14	Stop or start automatic exposure adjustments
Video Snapshot	14	Take a picture while shooting video (frame grab)
Time Lapse Video	11	Record frames with set delays to record a time lapse video
Multiple Cameras	9	More than one camera on a device, including front-facing and back-facing cameras
Focus Distance	9	Reports distances between the camera and objects that appear to be in focus
Zoom	8	Set image magnification
Exposure Compensation	8	Increase or decrease the light exposure level
GPS Data	5	Include or omit geographic location data with the image
White Balance	5	Set the white balance mode, which affects color values in the captured image
Focus Mode	5	Set how the camera focuses on a subject such as automatic, fixed, macro or infinity
Scene Mode	5	Types of photography situations such as night, beach, snow or candlelight scenes
JPEG Quality	5	Compression level for a JPEG image/
Flash Mode	5	Turn flash on, off, or use automatic setting
Color Effects	5	Apply a color effect to the captured image such as black and white, sepia tone or negative.
Anti-Banding	5	Reduces the effect of banding in color gradients due to JPEG compression
Picture Format	1	Specify the file format for the picture
Picture Size	1	Specify the pixel dimensions of the saved picture

Camera – Record Video

- It is done using the MediaRecorder Object
- The following steps must be performed to start recording:
  - Unlock the Camera
  - Prepare a MediaRecorder object
  - Start Recording
- The following steps must be performed to stop the record:
  - Stop the Media Recorder Object
  - Release the Media Recorder Object
  - Lock the Camera

Camera – Record Video

- It is done using the MediaRecorder Object
- The following steps must be performed to start recording:
  - Unlock the Camera
  - Prepare a MediaRecorder object
  - Start Recording
- The following steps must be performed to stop the record:
  - Stop the Media Recorder Object
  - Release the Media Recorder Object
  - Lock the Camera

Camera – Record Video

```
public boolean StartRecord(Camera cameraObject, MediaRecorder mr)
{
    cameraObject.unlock();
    mr.setCamera(cameraObject);
    mr.setAudioSource(MediaRecorder.AudioSource.CAMCORDER);
    mr.setVideoSource(MediaRecorder.VideoSource.CAMERA);
    mr.setProfile(CamcorderProfile.get(CamcorderProfile.QUALITY_HIGH));
    mr.setOutputFile("...");
    mr.setPreviewDisplay(surfaceHolder.getSurface());
    try {
        mr.prepare();
        mr.start();
    } catch (Exception e) {
        mr.reset();
        mr.release();
        cameraObject.lock();
        return false;
    }
    return true;
}
public void StopRecord(Camera cameraObject, MediaRecorder mr)
{
    mr.stop();
    mr.reset();
    mr.release();
    cameraObject.lock();
}
```

Camera – Record Audio

```
public boolean StartAudioRecording(MediaRecorder mr, String fileName)
{
    mr.setAudioSource(MediaRecorder.AudioSource.MIC);
    mr.setOutputFormat(MediaRecorder.OutputFormat.THREE_GPP);
    mr.setOutputFile(fileName);
    mr.setAudioEncoder(MediaRecorder.AudioEncoder.AMR_NB);

    try {
        mr.prepare();
        mr.start();
        return true;
    }
    catch (Exception e) {
        return false;
    }
}

public void StopAudioRecording(MediaRecorder mr)
{
    mr.stop();
    mr.release();
}
```