内容提供者
contentProvider 作用 暴露应用的私有数据库
① 创建一个类 继承ContentProvider 重写方法 onCreate getType insert delete update query
contentProvider操作数据库 还是要通过SQliteDataBase
UriMatcher addUri(authorities,子路径,返回码);
②清单文件注册内容提供者 <provider name> authorities exported

内容解析者 contentresolver 获取内容解析者 上下文的方法 getContentResolver();
Uri content://authorites/子路径
content://sms   body date address
query
insert

联系人 data raw_contact mime_type

内容观察者 ContentObserver

onchange()

1计算机表示图形的几种方式
BMP 质量最高 用于计算机
PNG 质量高 用于计算机 网络 (android应用中的图片资源 一般使用png格式的图片)
jpeg 较高质量的 用于计算机 网络 电子邮件
gif 质量较差 用于计算机 网络 电子邮件 (动图)

单色 一个像素用1位2进制数来表示   200*200 40000/8   5k 4.8 多出来的用来保存元数据
16色 1个像素 16总颜色 2^4   40000/2 /1024   20k
256色 1个像素 256总颜色 2^8 40k
24位位图 1个像素 用24位的2进制数来表示 一个像素3byte 40000*3   117k

argb8888 1个像素 4byte
alpha 8位二进制数
red   8位二进制数
green 8位二进制数
blue   8位二进制数

2 大图加载
09-24 01:42:20.623: E/dalvikvm-heap(2339): Out of memory on a 30720012-byte allocation.

思路① 计算屏幕的分辨率 和图片分辨率的比例 用这个比例来确定压缩大小

```
 public void loadpic2(){

        //比较图片和屏幕的分辨率 如果图片分辨率比屏幕分辨率高
        //可以使用 图片宽度/屏幕的宽度
        //图片的高度/屏幕的高度
        //通过这两个比值来确定 inSampleSize


        //②获取图片的宽高
        BitmapFactory.Options option = new Options();
        //inJustDecodeBounds 只解析图片的宽高以及类型
        //如果这个参数设置为true那么调用BitmapFactory.decodeXXXX方法 只会把图片的宽度 高度 图片的类型读出来
        //不会解码图片
        option.inJustDecodeBounds = true;
        Bitmap bitmap = BitmapFactory.decodeFile(path, option);
//      if(bitmap==null){
//          System.out.println("图片的宽度"+option.outWidth+"高度"+option.outHeight);
//      }
        int width = option.outWidth;
        int height = option.outHeight;
        //③计算inSampleSize
        //只有当图片的宽度和高度大于屏幕的宽度和高度才压缩图片
        if(width>screenWidth||height>screenHeight){
            int widthIndex = Math.round((float)width/(float)screenWidth);
            int heightIndex = Math.round((float)height/(float)screenHeight);
            //计算压缩比例 取宽高比例的最大值
            option.inSampleSize = Math.max(widthIndex, heightIndex);
        }
        //用计算好的inSampleSize 加载图片
        option.inJustDecodeBounds = false;//把inJustDecodeBounds设置为false 开始加载图片
        bitmap = BitmapFactory.decodeFile(path, option);
        //用imageview显示图片
        iv_image.setImageBitmap(bitmap);
    }
```

思路② 通过不断的尝试 提高inSampleSize的值 一直到inSampleSize提高到一个合适的值 成功加载图片'

```
 public void loadpic3(){
        BitmapFactory.Options option = new Options();
        option.inSampleSize = 1;
        Bitmap bitmap = null;
        int i = 1;
        for(;;){
            try {
                //不断试验inSampleSize 如果压缩比例不够高 抓住outofmemory提高压缩
```

```
比例 接着试验
                //直到可以成功加载为止
                option.inSampleSize = i;
                 bitmap = BitmapFactory.decodeFile(path, option);
                 break;
            } catch (Error e) {
                i*=2;
                System.out.println("i = "+i);
            }
        }
        //加载到imageView
        iv_image.setImageBitmap(bitmap);
    }
```

3.创建图片副本

4 图形处理的api
rotate 旋转

```
setRotate设置旋转的角度  后两个参数 设置旋转的中心点
matrix.setRotate(90,copybm.getWidth()/2,copybm.getHeight()/2);
```

translate平移

```
 setTranslate平移 第一个参数 x方向平移的距离  第二个参数 y方向平移的距离
 matrix.setTranslate(30, 0);
```

scale缩放

```
 matrix.setScale(1f, -1f);
//多次调用set 最近一次会把之前的效果清除掉
 //如果想在之前的效果基础上进一步处理图片 需要调用postXXXX方法
//matrix.postTranslate(copybm.getWidth(), 0);
matrix.postTranslate(0, copybm.getHeight());
```

5 画画板小案例
6 撕衣服小案例
6.1创建图片的副本
6.2给控件设置一个触摸监听
触摸监听需要注意 onTouch方法 返回值是true 否者只能接受到Down事件 move和up都收不到

7 mediaplayer播放音频
①混合方式开启服务

②在服务的onCreate方法中创建一个mediaplayer 并且准备好
③ 点击播放按钮开始播放音乐
服务暴露播放暂停方法/ 获得当前播放状态的方法
3.1 一个按钮实现 播放/暂停
3.2 根据播放的状态修改图标的状态
3.3 当activity退出再次进入的时候 根据当前播放状态更新图标状态(onserviceconnected)
④ 进度条展示播放进度
4.1 服务暴露 获取总时长的方法 获取当前播放进度的方法
4.2 通过Handler 执行延迟消息 每隔500ms 更新一次进度条进度
4.3 handler执行的延迟消息 当音乐暂停 以及界面退出的时候 要把消息移除掉
⑤拖动进度条跟新播放的进度
5.1 服务暴露 seekTo 移动到某个位置播放的方法
5.2 给seekbar设置拖动的监听 在onProgressChanged方法中 通过传入的progress更新播放器的播放进度

8 音乐播放器完成
9 mediaplayer生命周期
播放网络音乐
①要使用异步准备

```
1.    player.prepareAsync();
```

②异步准备之后要添加一个准备好的监听

```
1.  //添加准备好的监听
2.              player.setOnPreparedListener(new OnPreparedListener() {
3.                  //如果准备好了就会执行onPrepared方法  mp就是当前的MediaPlayer对象
4.                  @Override
5.                  public void onPrepared(MediaPlayer mp) {
6.                      // 如果当前媒体文件已经准备好了 就会走onPrepared
7.                      mp.start();
8.                  }
9.              });
```

③要添加联网的权限
10surfaceview介绍
通过surfaceView和mediaplayer配合 可以播放视频
mediaplayer负责音频的播放 和音视频的解码
surfaceview只是负责展示
surfaceView 是一个重量级控件 加载需要一些时间
surfaceHolder用来控制surfaceview的数据展示
surfaceView 中保存了一个surface 可以用来展示频繁变化的界面

MediaPlayer 设置显示的控件

```
player.setDisplay(holder);
```

获取surfaceHolder对象

```
surface = (SurfaceView) findViewById(R.id.surface);
holder = surface.getHolder();
```

11videoview控件介绍

对surfaceview和MediaPlayer的封装

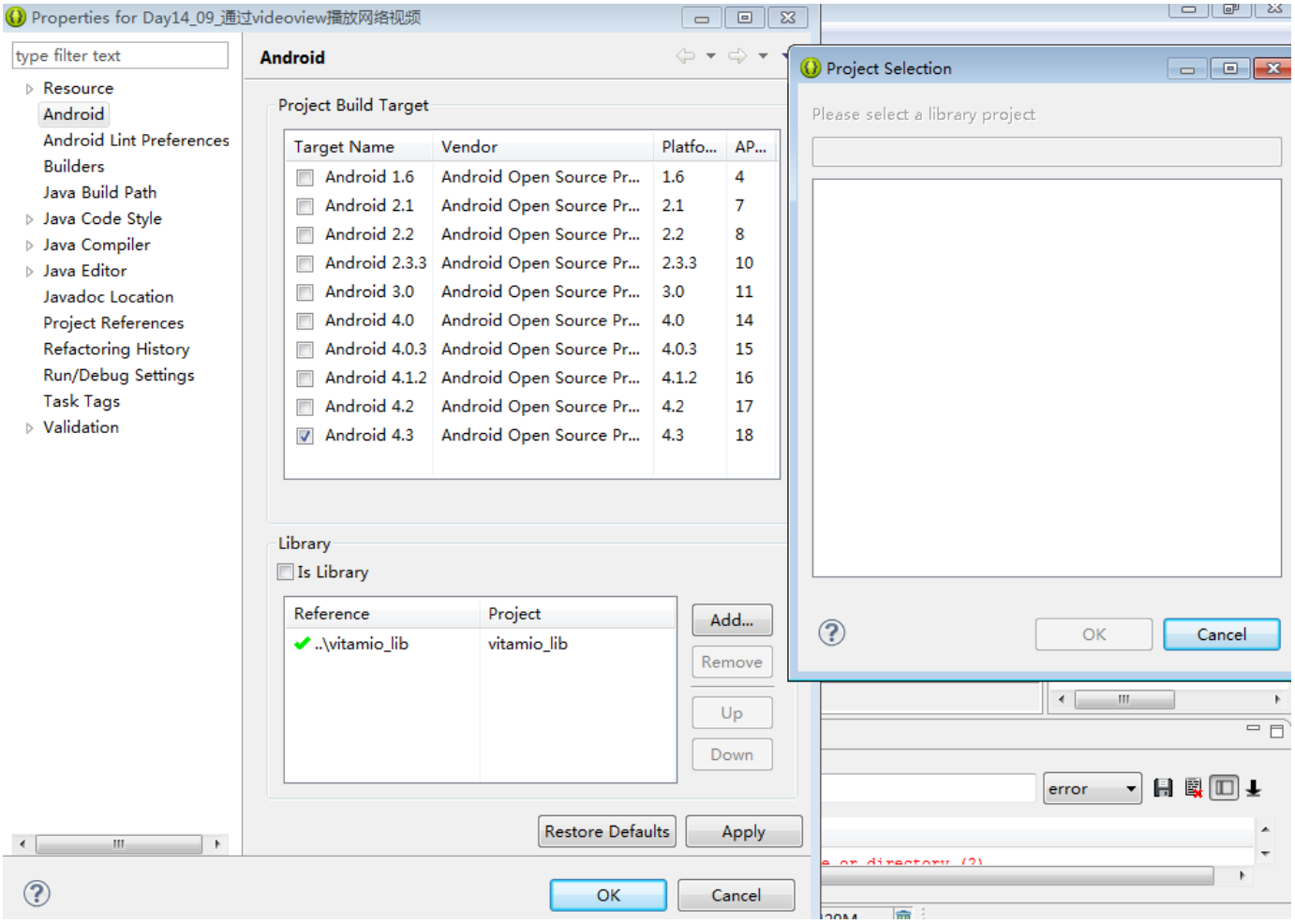
12 vitamio 三方播放视频的解码框架

ffmpeg c/c++开发的开源音视频解码项目

①library工程的使用

1 copy project to workspace 一定要选上 把项目拷贝到工作空间

②



③使用之前要clean一下 library工程

1 引入vitamio框架 以library、

2 在布局中定义VideoView

```
<io.vov.vitamio.widget.VideoView
 android:id="@+id/vv"
 android:layout_width="match_parent"
 android:layout_height="match_parent"
 />
```

3 mainactivity代码

插件vitamio框架检查是否可用

```
if (!LibsChecker.checkVitamioLibs(this)) {
        return;
    }

    final VideoView vv = (VideoView) findViewById(R.id.vv);
    vv.setVideoPath("http://192.168.1.2:8080/haha.avi");
    vv.setOnPreparedListener(new OnPreparedListener() {

        @Override
        public void onPrepared(MediaPlayer mp) {
            vv.start();

        }
    });
    //设置video的控制器
    vv.setMediaController(new MediaController(this));
```

4 一定要在清单文件初始化InitActivity

```
<activity android:name="io.vov.vitamio.activity.InitActivity"></activity>
```

①图片处理
加载大图
创建图片副本
setOntouchListener

画画板
撕衣服

② audio
音乐播放器
video
用vitamio 播放网络视频