Evan
Only let oneself become strong enough, good enough, can afford the life that you want to.

Android 网络编程--上传文件及相应的参数到服务器

标签： android 网络编程 上传文件

2016-05-11 21:06 4260人阅读 评论(0) 收藏 举报

分类： 移动开发（38） 网络编程

之前一直在做SiteCheck的项目，所用到的知识大部分都涉及到网络编程方面，所以现在有时间先把它的使用方法及一些注意事项记录下来。在这里我用两种例子让大家了解它的使用方法：

(1)上传图片及相应参数到服务器 （2）上传语音及相应参数到服务器。代码比较多。。。。

先贴上代码，再解析：

UploadFileTask .Java：（实现异步上传的执行类）

[java]

```
<span style="font-size:14px;">public class UploadFileTask extends AsyncTask<String, Void, String>{
    /**
     *  可变长的输入参数，与AsyncTask.exucute()对应
     */
    private  ProgressDialog pdialog;
    private  Activity context=null;
    public UploadFileTask(Activity ctx){
        this.context=ctx;
        pdialog=ProgressDialog.show(context, "正在加载...", "系统正在处理您的请求");
    }

    @Override
    protected void onPostExecute(String result) {
        // 返回HTML页面的内容
        pdialog.dismiss();
        if(UploadUtils.taskSUCCESS.equals(result)){
            Toast.makeText(context, "上传成功!",Toast.LENGTH_LONG ).show();
            int size = Bimp.tempSelectBitmap.size();
            while(size>0){
                ImageItem imageItem = Bimp.tempSelectBitmap.get(size-1);
                String imgPath = imageItem.getImagePath();
                //从SD卡中删除图片
                FileUtils.delFile(imgPath);
                Bimp.tempSelectBitmap.remove(imageItem);
                Bimp.max--;
                //获取对应照片的音频数据
                List<Voice> mDatas = StaticVariable.mapVoice.get(imgPath);
                if(mDatas!=null){
                    int voiceNum = mDatas.size();
                    for(int j=0;j<voiceNum;j++){
                        String voicePath = mDatas.get(j).getFilePath();
                        //从SD卡中删除音频
                        FileUtils.delFile(voicePath);
                        StaticVariable.mapVoice.remove(imgPath);
                    }
                }
                size--;
            }
            FieldFragment.adapter.notifyDataSetChanged();
```

```
            FieldFragment.imgRemark.setText("");

        }else if(UploadUtils.terSUCCESS.equals(result)){
            Toast.makeText(context, "上传成功!",Toast.LENGTH_LONG ).show();
            int size = Bimp.terminalSelectBitmap.size();
            while(size>0){
                ImageItem imageItem = Bimp.terminalSelectBitmap.get(size-1);
                if(imageItem.isCamera()){
                    //从SD卡中删除
                    FileUtils.delFile(imageItem.getImagePath());
                }
                Bimp.terminalSelectBitmap.remove(imageItem);
                Bimp.terminalmax--;
                size--;
            }
            UpdateTerminalFragment.terminalAdapter.notifyDataSetChanged();

        }else{
            Toast.makeText(context, "上传失败!",Toast.LENGTH_LONG ).show();
         @Override
    protected String doInBackground(String... params) {
        String result = "";
        //上传终端图片
        if(params[0].equals(StaticVariable.TERMINAL_UPLOAD_IMG)){
            //获取对应照片的文件
            ArrayList<ImageItem> terBitmap = Bimp.terminalSelectBitmap;
            File[] files = null;
            if(terBitmap!=null){
                int imgNum = terBitmap.size();
                files = new File[imgNum];
                for(int i=0;i<imgNum;i++){
                    String imgPath = terBitmap.get(i).getImagePath();
                    files[i] = new File(imgPath);
                }
            }
            result = UploadUtils.uploadTerFile(files, StaticUrl.urlUploadTImgs);
        }
        //上传任务信息
        else if(params[0].equals(StaticVariable.TASK_UPLOAD_IMG)){
            //获取对应照片的文件
            ArrayList<ImageItem> tempBitmap = Bimp.tempSelectBitmap;
            for(int i=0;i<tempBitmap.size();i++){
                String imgPath = tempBitmap.get(i).getImagePath();
                //获取对应照片的音频数据
                List<Voice> mDatas = StaticVariable.mapVoice.get(imgPath);
                File[] voiceFiles = null;
                if(mDatas!=null){
                    int voiceNum = mDatas.size();
                    voiceFiles = new File[voiceNum];
                    for(int j=0;j<voiceNum;j++){
                        String voicePath = mDatas.get(j).getFilePath();
                        voiceFiles[j] = new File(voicePath);
                    }
                }
                File file=new File(imgPath);
                result = UploadUtils.uploadFile(file, voiceFiles, StaticUrl.urlAddImg);
            }
        }
        return result;    }


    @Override
    protected void onProgressUpdate(Void... values) {
    }
}
</span>
```

这个异步上传处理类是比较简单的：

在要上传的地方执行下面代码就会进入到异步处理类中进行相应的操作：

```
UploadFileTask uploadFileTask=new UploadFileTask(activity);

uploadFileTask.execute(StaticVariable.TASK_UPLOAD_IMG);
```

下面解析一下对上传类的操作：

result = UploadUtils.uploadFile(file, voiceFiles, StaticUrl.urlAddImg);

file: 这里是图片文件参数

voiceFiles：这个是语音文件，是一个文件数组

StaticUrl.urlAddImg: 文件上传的地址

resutl：返回结果

UploadUtils .java:

```java
<span style="font-size:14px;">public class UploadUtils {
    private static final int TIME_OUT = 10*10000000;   //超时时间
    private static final String CHARSET = "utf-8";     //设置编码
    public static final String FAILURE="0";            //上传失败的返回结果
    public static final String taskSUCCESS="1";<span style="white-space:pre"> </span>   //上传成功的返回结果
    public static final String terSUCCESS = "2";

    //上传任务信息
    public static String uploadFile(File file, File[] files, String RequestURL) {
        String  BOUNDARY =  UUID.randomUUID().toString();  //边界标识   随机生成（也可以自定义）
        String PREFIX = "--" , LINE_END = "\r\n";
        String CONTENT_TYPE = "multipart/form-data";   //内容类型（这是标准通用型，不过最好指明明确的文件类型）
        try {
            URL url = new URL(RequestURL);
            HttpURLConnection conn = (HttpURLConnection) url.openConnection();
            conn.setReadTimeout(TIME_OUT);
            conn.setConnectTimeout(TIME_OUT);
            conn.setDoInput(true);  //允许输入流
            conn.setDoOutput(true); //允许输出流
            conn.setUseCaches(false);  //不允许使用缓存
            conn.setRequestMethod("POST");  //请求方式
            conn.setRequestProperty("Charset", CHARSET);  //设置编码
            conn.setRequestProperty("connection", "keep-alive");
            conn.setRequestProperty("Content-Type", CONTENT_TYPE + ";boundary=" + BOUNDARY);
            if(file!=null) {
                /**
                 * 当文件不为空，把文件包装并且上传
                 */
                OutputStream outputSteam=conn.getOutputStream();           //获得网络传输的输出流

                DataOutputStream dos = new DataOutputStream(outputSteam);
                StringBuffer sb = new StringBuffer();

                sb.append(PREFIX); sb.append(BOUNDARY); sb.append(LINE_END);
                sb.append("Content-Disposition: form-data; name=\"uid\"" + LINE_END);     //uid 作为上传参数的key（服务器端接收这个参数要对应这个key）
                sb.append(LINE_END); sb.append(LoginActivity.uid + LINE_END);             //LoginActivity.uid 是要上传参数的值

                if(IScreateRecordHttp.hasSameRecord==0){
                    sb.append(PREFIX); sb.append(BOUNDARY); sb.append(LINE_END);
                    sb.append("Content-Disposition: form-data; name=\"hasSameRecord\""+LINE_END);
                    sb.append(LINE_END); sb.append(0+LINE_END);

                    sb.append(PREFIX); sb.append(BOUNDARY); sb.append(LINE_END);
                    sb.append("Content-Disposition: form-data; name=\"isid\""+LINE_END);
                    sb.append(LINE_END); sb.append(SeeTaskFragment.isid+LINE_END);
                }else if(IScreateRecordHttp.hasSameRecord==1){
                    sb.append(PREFIX); sb.append(BOUNDARY); sb.append(LINE_END);
                    sb.append("Content-Disposition: form-data; name=\"hasSameRecord\""+LINE_END);
                    sb.append(LINE_END); sb.append(1+LINE_END);

                    sb.append(PREFIX); sb.append(BOUNDARY); sb.append(LINE_END);
                    sb.append("Content-Disposition: form-data; name=\"rid\""+LINE_END);
                    sb.append(LINE_END); sb.append(IScreateRecordHttp.rid+LINE_END);
                }

                sb.append(PREFIX); sb.append(BOUNDARY); sb.append(LINE_END);
                sb.append("Content-Disposition: form-data; name=\"remark\""+LINE_END);
                sb.append(LINE_END); sb.append(FieldFragment.getRemark() + LINE_END);

                /**
```

```
                 * 这里重点注意:
                 * name里面的值为服务器端需要key   只有这个key 才可以得到对应的文件
                 * filename是文件的名字，包含后缀名的   比如:abc.png
                 */
                sb.append(PREFIX); sb.append(BOUNDARY); sb.append(LINE_END);
                sb.append("Content-Disposition: form-data; name=\"file\"; filename=\""+file.getName()+"\""+LINE_END)
                sb.append("Content-Type: image/jpeg");            //指明文件类型为图片jpeg类型
                sb.append(LINE_END);sb.append(LINE_END);

                dos.write(sb.toString().getBytes());

                InputStream is = new FileInputStream(file);     //把图片文件转换成二进制流的方式 写入到dos中
                byte[] bytes = new byte[1024];
                int len = 0;
                while((len=is.read(bytes))!=-1) {
                    dos.write(bytes, 0, len);
                }
                is.close();
                dos.write(LINE_END.getBytes());

                byte[] end_data = (PREFIX+BOUNDARY+PREFIX+LINE_END).getBytes();
                dos.write(end_data);
                dos.flush();          //释放流
                /**
                 * 获取响应码  200=成功
                 * 当响应成功，获取响应的流
                 */
                int res = conn.getResponseCode();
                Log.e("Mine", "response code:"+res);
                if(res==200) {                          //这里是上传成功就继续上传音频文件，假如不需要就直接 return
                    InputStream isre = conn.getInputStream();
                    BufferedReader buffer = new BufferedReader(
                            new InputStreamReader(isre,"gbk"));
                    String line = "";
                    StringBuffer sbre = new StringBuffer();
                    while ((line = buffer.readLine()) != null) {
                        sbre.append(line);
                    }
                    String result = sbre.toString();   //获取返回结果
                    Log.e("Mine",result);

                    try{
                        JSONObject jsonObject = new JSONObject(result);
                        int iid = jsonObject.getInt("iid");
                        String type = jsonObject.getString("type");
                        String msg = jsonObject.getString("msg");
                        if(type.equals("SUCCESS")){
                            /*
                             在AsyncTask异步处理方法里面不可以再使用异步处理，
                              会导致UI（主）线程执行错误
                        <span style="white-space:pre">    </span>*/
                            uploadVoice(iid,files);
                        }
                    }catch(JSONException e){
                        e.printStackTrace();
                    }catch(ParseException e1){
                        e1.printStackTrace();
                    }

                    return taskSUCCESS;
                }
            }
        } catch (MalformedURLException e) {
            e.printStackTrace();
        } catch (IOException e) {
            e.printStackTrace();
        }
        return FAILURE;
    }

    //上传图片对应的语言
    private static void uploadVoice(int iid,File[] files) {
        String  BOUNDARY =  UUID.randomUUID().toString();  //边界标识   随机生成
        String PREFIX = "--" , LINE_END = "\r\n";
        String CONTENT_TYPE = "multipart/form-data";   //内容类型

        try{
            String RequestURL = StaticUrl.urlAddVoices;
            URL url = new URL(RequestURL);
```

```
            HttpURLConnection conn = (HttpURLConnection) url.openConnection();
            conn.setReadTimeout(TIME_OUT);
            conn.setConnectTimeout(TIME_OUT);
            conn.setDoInput(true);  //允许输入流
            conn.setDoOutput(true); //允许输出流
            conn.setUseCaches(false);  //不允许使用缓存
            conn.setRequestMethod("POST");  //请求方式
            conn.setRequestProperty("Charset", CHARSET);  //设置编码
            conn.setRequestProperty("connection", "keep-alive");
            conn.setRequestProperty("Content-Type", CONTENT_TYPE + ";boundary=" + BOUNDARY);
            if(files!=null) {
                /**
                 * 当文件不为空，把文件包装并且上传
                 */
                OutputStream outputSteam=conn.getOutputStream();

                DataOutputStream dos = new DataOutputStream(outputSteam);
                StringBuffer sb = new StringBuffer();

                sb.append(PREFIX); sb.append(BOUNDARY); sb.append(LINE_END);
                sb.append("Content-Disposition: form-data; name=\"uid\"" + LINE_END);
                sb.append(LINE_END); sb.append(LoginActivity.uid + LINE_END);

                sb.append(PREFIX); sb.append(BOUNDARY); sb.append(LINE_END);
                sb.append("Content-Disposition: form-data; name=\"iid\""+LINE_END);
                sb.append(LINE_END); sb.append(iid+LINE_END);

                dos.write(sb.toString().getBytes());

                for(File file : files){         //多文件上传，用循环

                    StringBuffer sbImgs = new StringBuffer();               //这里面要注意：重新实例化一个StringBuffer，不然导致
StringBuffer的重用，原来的参数也跟后来的参数一起传到服务器了。
                    // set 头部
                    sbImgs.append(PREFIX); sbImgs.append(BOUNDARY); sbImgs.append(LINE_END);
                    sbImgs.append("Content-Disposition: form-data; name=\"voices\"; filename=\""+file.getName()+"\""+LINE_END);
                    sbImgs.append("Content-Type: audio/mp3"+LINE_END);
                    sbImgs.append(LINE_END);

                    // 1. write sb
                    dos.writeBytes(sbImgs.toString());

                    // 取得文件的FileInputStream
                    FileInputStream fis = new FileInputStream(file);
                    // 设置每次写入1024bytes
                    int bufferSize = 1024;
                    byte[] buffer = new byte[bufferSize];

                    int length = -1;
                    // 从文件读取数据至缓冲区
                    while ((length = fis.read(buffer)) != -1) {
                        dos.write(buffer, 0, length);
                    }
                    dos.writeBytes(LINE_END);
                    fis.close();

                    dos.writeBytes(LINE_END);
                    dos.writeBytes(LINE_END);
                }

                byte[] end_data = (PREFIX+BOUNDARY+PREFIX+LINE_END).getBytes();
                dos.write(end_data);
                dos.flush();
                /**
                 * 获取响应码  200=成功
                 * 当响应成功，获取响应的流
                 */
                int res = conn.getResponseCode();
                Log.e("Mine", "response code:"+res);
                if(res==200) {
                //  return terSUCCESS;
                }
            }
        } catch (MalformedURLException e) {
            e.printStackTrace();
        } catch (IOException e) {
            e.printStackTrace();
        }
    //  return FAILURE;
```

```
    }

    //上传终端图片
    public static String uploadTerFile(File[] files, String RequestURL) {
        String  BOUNDARY =  UUID.randomUUID().toString();  //边界标识   随机生成
        String PREFIX = "--" , LINE_END = "\r\n";
        String CONTENT_TYPE = "multipart/form-data";   //内容类型

        try{
            URL url = new URL(RequestURL);
            HttpURLConnection conn = (HttpURLConnection) url.openConnection();
            conn.setReadTimeout(TIME_OUT);
            conn.setConnectTimeout(TIME_OUT);
            conn.setDoInput(true);  //允许输入流
            conn.setDoOutput(true); //允许输出流
            conn.setUseCaches(false);  //不允许使用缓存
            conn.setRequestMethod("POST");  //请求方式
            conn.setRequestProperty("Charset", CHARSET);  //设置编码
            conn.setRequestProperty("connection", "keep-alive");
            conn.setRequestProperty("Content-Type", CONTENT_TYPE + ";boundary=" + BOUNDARY);
            if(files!=null) {
                /**
                 * 当文件不为空，把文件包装并且上传
                 */
                OutputStream outputSteam=conn.getOutputStream();

                DataOutputStream dos = new DataOutputStream(outputSteam);
                StringBuffer sb = new StringBuffer();

                sb.append(PREFIX); sb.append(BOUNDARY); sb.append(LINE_END);
                sb.append("Content-Disposition: form-data; name=\"uid\"" + LINE_END);
                sb.append(LINE_END); sb.append(LoginActivity.uid + LINE_END);

                sb.append(PREFIX); sb.append(BOUNDARY); sb.append(LINE_END);
                sb.append("Content-Disposition: form-data; name=\"tid\""+LINE_END);
                sb.append(LINE_END); sb.append(TerminalInfo.tid+LINE_END);

                sb.append(PREFIX); sb.append(BOUNDARY); sb.append(LINE_END);
                sb.append("Content-Disposition: form-data; name=\"remark\""+LINE_END);
                sb.append(LINE_END); sb.append("备注" + LINE_END);

                dos.write(sb.toString().getBytes());

                for(File file : files){

                    StringBuffer sbImgs = new StringBuffer();
                    // set 头部
                    sbImgs.append(PREFIX); sbImgs.append(BOUNDARY); sbImgs.append(LINE_END);
                    sbImgs.append("Content-Disposition: form-data; name=\"imgs\"; filename=\""+file.getName()+"\""+LINE_END);
                    sbImgs.append("Content-Type: image/jpeg"+LINE_END);
                    sbImgs.append(LINE_END);

                    // 1. write sb
                    dos.writeBytes(sbImgs.toString());

                    // 取得文件的FileInputStream
                    FileInputStream fis = new FileInputStream(file);
                    // 设置每次写入1024bytes
                    int bufferSize = 1024;
                    byte[] buffer = new byte[bufferSize];

                    int length = -1;
                    // 从文件读取数据至缓冲区
                    while ((length = fis.read(buffer)) != -1) {
                        dos.write(buffer, 0, length);
                    }
                    dos.writeBytes(LINE_END);
                    fis.close();

                    dos.writeBytes(LINE_END);
                    dos.writeBytes(LINE_END);
                }

                byte[] end_data = (PREFIX+BOUNDARY+PREFIX+LINE_END).getBytes();
                dos.write(end_data);
                dos.flush();
            /**
<span style="white-space:pre">                </span> * 获取响应码  200=成功
<span style="white-space:pre">                </span> * 当响应成功，获取响应的流
```

```
297.  <span style="white-space:pre">              </span> */
298.  <span style="white-space:pre">              </span>int res = conn.getResponseCode();
299.  <span style="white-space:pre">              </span>Log.e("Mine", "response code:"+res);
300.  <span style="white-space:pre">              </span>if(res==200) {
301.  <span style="white-space:pre">                  </span>return terSUCCESS;
302.  <span style="white-space:pre">              </span>}
303.  <span style="white-space:pre">          </span>}
304.  <span style="white-space:pre">      </span>} catch (MalformedURLException e) {
305.  <span style="white-space:pre">          </span>e.printStackTrace();
306.  <span style="white-space:pre">      </span>} catch (IOException e) {
307.  <span style="white-space:pre">          </span>e.printStackTrace();
308.  <span style="white-space:pre">      </span>}
309.  <span style="white-space:pre">      </span>return FAILURE;
310.  <span style="white-space:pre">  </span>}
311.
312.
313.  }</span>
```

基本上掌握了这些方法，上传文件的操作就剩下套路了，以二进制流的方式上传文件要注意文件，参数它们的头，和尾，就是那些边界值。

好啦，大概就这些吧，下面推荐一下网络编程上传文件的文章：

http://blog.csdn.**NET**/carterjin/article/details/7571915

http://blog.csdn.**net**/lmj623565791/article/details/23781773

- 上一篇 Servlet3.0学习总结(二)——使用注解标注过滤器(Filter)
- 下一篇 Android 实现对图片 Exif 的修改（Android 自带的方法）

相关文章推荐

- Android网络编程—同时上传参数和文件到服务器
- （网络编程）通过Socket完成文件上传
- 黑马程序员---网络编程
- 关于网络编程知识的了解应用
- Android深入探究笔记之六 -- 与互联网互访数据
- Android 上百实例源码分析以及开源分析
- 黑马程序员——网络编程及GUI
- 浅谈android网络编程
- Android资源下载
- 我的IT历程学习备忘录