IHttpRequest.java 接口定义

```
import java.io.IOException;
import java.io.UnsupportedEncodingException;

public interface IHttpRequest {

    /**
     * 常用字符
     */
    String ISO_8859_1="ISO-8859-1",UTF_8="UTF-
8",GBK="GBK",GB2312="GB2312",
            GET="GET",POST="POST",AND="&",QM="?",EQ="=",
            BOUNDARY="---------------doyouloveme";
    /**
     * 设置请求参数
     * @param key
     * @param value
     */
    public void setParameter(String key,String value) throws
UnsupportedEncodingException;

    /**
     * 初始化请求属性头
     */
    public void initRequestProperty() throws IOException ;

    /**
     * 添加请求头
     * @param key
     * @param value
     */
    public void addRequestProperty(String key,String value);

    /**
     * 打开连接
     */
```

```
    public void openURLConnection() throws IOException;

    /**
     * response 响应结果
     * @return
     */
    public HttpResponse getResponse() throws IOException;

    /**
     * 发送HTTP请求
     */
    public void sendHTTP() throws IOException;

    /**
     * 关闭WEB连接
     */
    public void disconnect();

    /**
     * 设置页面编码
     * @param encode
     */
    public void setPageEncode(String encode);

    /**
     * 设置user-agent
     */
    public void setUserAgent(String agent);

}
```

HttpResponse.java 响应结果封装

```
public class HttpResponse {

    /**
     * 响应结果
     */
    private String value;
```

```java
/**
 * 返回的响应代码
 */
private int code;

/**
 * 响应消息
 */
private String message;

/**
 * 附件名字，如果存在
 */
private String fileName;

/**
 * 附件
 */
private byte[] fileBytes;

public String getValue() {
    return value;
}

public void setValue(String value) {
    this.value = value;
}

public int getCode() {
    return code;
}

public void setCode(int code) {
    this.code = code;
}

public String getMessage() {
```

```
        return message;
    }

    public void setMessage(String message) {
        this.message = message;
    }

    public String getFileName() {
        return fileName;
    }

    public void setFileName(String fileName) {
        this.fileName = fileName;
    }

    public byte[] getFileBytes() {
        return fileBytes;
    }

    public void setFileBytes(byte[] fileBytes) {
        this.fileBytes = fileBytes;
    }

}
```

AHttpRequest.java 抽象类基本实现

```
import java.io.ByteArrayOutputStream;
import java.io.IOException;
import java.io.InputStream;
import java.io.OutputStream;
import java.io.PrintWriter;
import java.io.UnsupportedEncodingException;
import java.net.HttpURLConnection;
import java.net.URL;
import java.net.URLConnection;
import java.net.URLEncoder;
```

```
import java.util.HashMap;
import java.util.List;
import java.util.Map;
import java.util.Set;

public abstract class AHttpRequest implements IHttpRequest {

    /**
     * 请求参数key=value&key=value
     */
    protected StringBuilder paramBuilder=new StringBuilder();

    /**
     * 条件MAP 使用MAP方便条件替换
     */
    protected Map<String, String> paramMap=new
HashMap<String, String>();

    /**
     * 请求结果
     */
    protected HttpResponse httpResponse;

    /**
     * http请求连接
     */
    protected HttpURLConnection httpURLConnection;

    /**
     * 请求编码
     */
    protected String encode=UTF_8;

    /**
     *资源链接
     */
    protected URL url;
```

```
    /**
     * 请求方法默认为GET
     */
    protected String METHOD=GET;

    /**
     * SESSION ID
     */
    private static String JSESSIONID;

    /**
     * 参数是否被改动过
     */
    private boolean paramChange=false;
    /**
     * 上传附件
     */
    protected boolean upload=false;

    public AHttpRequest(URL url,String method) throws
IOException {
        this.METHOD = method;
        this.url = url;
    }

    @Override
    public void setParameter(String key, String value) throws
UnsupportedEncodingException {
        paramMap.put(key, value);
        paramChange=true;
    }

    @Override
    public void initRequestProperty() throws IOException {
        System.setProperty("http.keepAlive", "false");
       HttpURLConnection.setFollowRedirects(true);
```

```
        httpURLConnection.setInstanceFollowRedirects(true);
        httpURLConnection.setDoOutput(METHOD.equals(POST));
        httpURLConnection.setDoInput(true);
        httpURLConnection.setUseCaches(false);
        httpURLConnection.setAllowUserInteraction(false);
        httpURLConnection.setConnectTimeout(10000);
        httpURLConnection.setRequestProperty("User-
Agent","Mozilla/4.0(compatible;MSIE6.0;Windows NT 5.0)");
        httpURLConnection.setRequestProperty("Host",getURLHost());

httpURLConnection.setRequestProperty("Accept","image/gif,image/x-
xbitmap,image/jpeg,application/x-shockwave-
flash,application/vnd.ms-excel,application/vnd.ms-
powerpoint,application/msword,*/*");
        httpURLConnection.setRequestProperty("Accept-
Language","zh-cn");
        if(!upload){
            httpURLConnection.setRequestProperty("Content-
Type","application/x-www-form-urlencoded");
            httpURLConnection.setRequestProperty("Content-
Length",String.valueOf(getContentLength()));
        }
        httpURLConnection.setRequestProperty("Cache-Control","no-
cache");
        httpURLConnection.setRequestProperty("Connection", "Keep-
Alive");
        httpURLConnection.setRequestProperty("Cookie",
JSESSIONID==null?"":JSESSIONID);
        httpURLConnection.setRequestMethod(METHOD);
    }

    @Override
    public void openURLConnection() throws IOException {
        if(METHOD.equals(GET)&&!
StringUtil.isNull(paramBuilder.toString())){
            String urlString=this.url.toString();
            if(urlString.indexOf("?")>0){
```

```
                this.url=new URL(urlString+paramBuilder.replace(0, 1,
"&"));
            }else{
                this.url=new URL(urlString+paramBuilder);
            }
        }
        URLConnection urlConnection=this.url.openConnection();
        if(urlConnection instanceof HttpURLConnection){
            httpURLConnection=(HttpURLConnection)urlConnection;
        }else{
            throw new IllegalArgumentException("不是一个HTTP协议
请求");
        }
    }

    /**
     * 发送请求，一般不需要重写
     */
    @Override
    public void sendHTTP() throws IOException {
        paramMap2String();
        openURLConnection();
        initRequestProperty();
        if (METHOD.equals(POST) ) {
            OutputStream outputStream =
httpURLConnection.getOutputStream();
            PrintWriter printWriter = new PrintWriter(outputStream);
            if(StringUtil.isNull(paramBuilder.toString())){
                //must be out
                printWriter.print("");
            }else{
                printWriter.print(paramBuilder.substring(1));
            }
            printWriter.flush();
            printWriter.close();
        }
        afterSendHTTP();
    }
```

```
    /**
     * HTTP请求执行后的执行方法
     * 进行一些数据的读取等处理
     * @throws IOException
     */
    protected void afterSendHTTP() throws IOException {
        readResponse();
        paramChange=false;
    }

    @Override
    public void addRequestProperty(String key, String value) {
        if(httpURLConnection==null)
            throw new NullPointerException("HTTP连接不存在");
        httpURLConnection.addRequestProperty(key, value);
    }

    @Override
    public HttpResponse getResponse() throws IOException {
        return this.httpResponse;
    }

    protected void readResponse() throws IOException{
        httpResponse=new HttpResponse();
        InputStream inputStream=null;
        ByteArrayOutputStream outByte=new
ByteArrayOutputStream();
        byte[] bs=new byte[1024];
        try{
            inputStream=httpURLConnection.getInputStream();
            if(inputStream==null)return;

    httpResponse.setCode(httpURLConnection.getResponseCode());


httpResponse.setMessage(httpURLConnection.getResponseMessage()
);
```

```
            httpResponse.setFileName(_getFileName());
            int count=-1;
            while((count=inputStream.read(bs))!=-1){
                outByte.write(bs,0,count);
            }
            if(outByte.size()==0)return;
            if(StringUtil.isNull(httpResponse.getFileName())){
                bs=outByte.toByteArray();
                String value=new String(bs,encode);
                httpResponse.setValue(value);
            }else{
                httpResponse.setFileBytes(outByte.toByteArray());
            }
        }finally{
            if(inputStream==null)return;
            inputStream.close();
            outByte.close();
        }
    }

    @Override
    public void setPageEncode(String encode) {
        this.encode=encode;
    }

    @Override
    public void disconnect() {
        if(httpURLConnection==null)
            throw new NullPointerException("HTTP连接不存在");
        httpURLConnection.disconnect();
    }

    @Override
    public void setUserAgent(String agent) {
        if(httpURLConnection==null)
            throw new NullPointerException("HTTP连接不存在");
        httpURLConnection.setRequestProperty("User-Agent", agent);
    }
```

```
    private String getURLHost(){
        return url.getHost();
    }

    protected String getJSESSIONID(){
        return JSESSIONID;
    }

    public void setJSESSIONID() {
        if(StringUtil.isNull(JSESSIONID)){
            String
cookieValue=httpURLConnection.getHeaderField("Set-Cookie");
            if(cookieValue!=null)
           JSESSIONID=cookieValue.substring(0,
cookieValue.indexOf(";"));
        }
    }

    /**
     * 退出程序时把SESSION置空
     */
    public static void invalidateSession(){
        JSESSIONID=null;
    }

    private String _getFileName() throws
UnsupportedEncodingException{
        String
_fileName=httpURLConnection.getHeaderField("Content-
Disposition");
        if(_fileName!=null){
            String fileName=
_fileName.substring(_fileName.lastIndexOf("=")+1);
            return new String(fileName.getBytes(ISO_8859_1),GBK);
        }
        return null;
    }
```

```
    /**
     * 更新URL连接
     * @param url
     * @throws IOException
     */
    public void setURL(URL url) throws IOException{
        this.url=url;
        this.httpResponse=null;
    }

    /**
     * 请求正文长度
     * @return
     */
    protected int getContentLength() {
        int length=paramBuilder.length();
        if(METHOD.equals(POST)&&length>0){
            return length-1;
        }
        return 0;
    }

    /**
     * 参数MAP集转换成字符串形式
     * @throws UnsupportedEncodingException 不支持的字符集
     */
    private void paramMap2String() throws
UnsupportedEncodingException{
        if(paramChange){
            if(paramBuilder.length()>0)
            paramBuilder.delete(0, paramBuilder.length());
            paramBuilder.append("?");
            for(Map.Entry<String, String> me:paramMap.entrySet()){
                paramBuilder.append(me.getKey()
+EQ+URLEncoder.encode(me.getValue(), UTF_8)+AND);
            }
            paramBuilder.deleteCharAt(paramBuilder.length()-1);
```

```
        }
    }

    /**
     * 清空参数 不在实行自动清空
     * 以便保存查询条件
     */
    public void clearParameter(){
        paramMap.clear();
        paramChange=true;
    }

    /**
     * 连接相关信息
     */
    public String toString(){
        if(httpURLConnection!=null){
            Map<String,List<String>>
httpMap=httpURLConnection.getHeaderFields();
            Set<String> keys=httpMap.keySet();
            StringBuilder headerInfo=new
StringBuilder("HeaderFields\n");
           for(String key:keys){
              List<String> vs=(List<String>)httpMap.get(key);
              headerInfo.append(key+":");
              for(String v:vs){
                 headerInfo.append(v+"\t");
              }
              headerInfo.append("\n");
           }
           System.out.println(headerInfo);
            return headerInfo.toString();
        }else{
            return super.toString();
        }
    }

}
```

```
import java.io.IOException;
import java.net.URL;
/**
* Get 请求类
*/
public class HttpGet extends AHttpRequest {

    public HttpGet(URL url) throws IOException {
        super(url,GET);
    }

}
```

HttpPost.java

```
import java.io.IOException;
import java.io.OutputStream;
import java.net.URL;
import java.util.Map;

/**
 * 扩展了{@link AHttpRequest}
 * 可实现附件上传功能
 * @author Aaron.tian
 *
 */
public class HttpPost extends AHttpRequest {

    public final static String PREFIX="--";
    public final static String LINE_END="\r\n";
    /**
     * 附件结尾标识符
     */
    public final static byte[] end_data_mark =
(LINE_END+PREFIX + BOUNDARY + PREFIX+LINE_END).getBytes();
    /**
     * 附件名称
```

```
     */
    private String fileName;

    /**
     * 附件内容
     */
    private byte[] fileBytes;

    public HttpPost(URL url) throws IOException {
        super(url,POST);
    }

    public String getFileName() {
        return fileName;
    }

    public void setFileName(String fileName) {
        this.fileName = fileName;
    }

    public byte[] getFileBytes() {
        return fileBytes;
    }

    public void setFileBytes(byte[] fileBytes) {
        this.fileBytes = fileBytes;
    }

    /**
     * 附件上传
     * @param fileBytes 附件字节数组
     * @param fileName 附件名称
     * @return 响应信息
     * @throws IOException
     */
    public int uploadFile(byte[] fileBytes,String fileName) throws
IOException{
```

```
        this.fileBytes=fileBytes;
        this.fileName=fileName;
        return uploadFile();
    }

    /**
     * 上传附件
     * @return 响应信息
     * @throws IOException
     * @see {@link HttpResponse}
     */
    public int uploadFile() throws IOException{
        upload=true;
        //check file
        if(this.fileName==null){
            throw new NullPointerException("文件名不能为空");
        }
        if(this.fileBytes==null){
            throw new NullPointerException("文件内容不能为空");
        }
        openURLConnection();
        initRequestProperty();
        httpURLConnection.setRequestProperty("Content-Type",
"multipart/form-data; boundary="+BOUNDARY);

        StringBuilder paramBuilder = new StringBuilder();
        // 参数信息
        for (Map.Entry<String, String> entry : paramMap.entrySet())
{
            paramBuilder.append(PREFIX + BOUNDARY + LINE_END);
            paramBuilder.append(("Content-Disposition: form-data;
name=\""+ entry.getKey() + "\""+LINE_END+LINE_END));
            paramBuilder.append(entry.getValue());
            paramBuilder.append(LINE_END);
        }
        // 文件头信息
        StringBuilder fileInfo = new StringBuilder(PREFIX +
```

```
BOUNDARY + LINE_END);
        fileInfo.append("Content-Disposition: form-data;
name=\"file1\"; filename=\""+ this.fileName + "\""+LINE_END);
        fileInfo.append("Content-Type:application/octet-
stream"+LINE_END+LINE_END);
        byte[] paramBytes=paramBuilder.toString().getBytes();
        byte[] fileInfoBytes=fileInfo.toString().getBytes();
        int
contentLength=paramBytes.length+fileInfoBytes.length+this.fileByte
s.length+end_data_mark.length;
        //set content-length
        httpURLConnection.setRequestProperty("Content-Length",
String.valueOf(contentLength));
        //发送数据
        OutputStream
outputStream=httpURLConnection.getOutputStream();
        outputStream.write(paramBytes);
        outputStream.write(fileInfoBytes);
        outputStream.write(this.fileBytes);
        outputStream.write(end_data_mark);
        outputStream.flush();
        outputStream.close();
        return httpURLConnection.getResponseCode();
    }

}
```

具体用法自己研究，不在详述