

```

1 //
2 // Tools + Wifi + HttpClient.java
3 // SoDemon
4 //
5
6 #1.获取Android系统属性参数<语言 + 提示框操作 + 日期处理>
7
8 1.获取Android系统属性参数
9
10 1) Android系统语言检测
11 说明: 使用以下函数, 可以检测出Android当前的语言。
12 函数: public static String check_language()
13 返回: 返回一个String类型,
14       返回"";代表当前是中文(包含简体中文和繁体中文);
15       返回"en", 代表当前是非中文
16 示例:
17 if (Tools.check_language().equals(""))
18     Log.e("SoDemo", "--Current--language--is--Chinese");
19 else
20     Log.e("SoDemo", "--Current--language--is-not-Chinese");;
21
22 2) 定制消息提示框
23 说明: 以下三个函数, 是用来提示信息用的, 即Android的Toast控件。
24 函数1:
25 /*
26  *显示提示框, Toast.LENGTH_LONG后自动消失
27  *参数tip:要显示的内容
28  */
29 public static void showLongToast(Context context, String tip)
30
31 函数2:
32 /*
33  *显示提示框, Toast.LENGTH_SHORT后自动消失
34  *参数tip:要显示的内容
35  */
36 public static void showShortToast(Context context, String tip)
37
38 函数3:
39 /*
40  *强制取消提示框
41  */
42 public static void cancelToast()
43
44 3) 将当前路径下的文件更新到系统相册
45 说明: 使用此函数可以将当前路径下的图像和录像文件更新到系统相册
46 函数:
47 /*
48  *path: 照片或录像文件等媒体文件的存储路径
49  */
50 public static void add_media(Context context, String path)
51 示例:
52 Tools.add_media(LiveActivity.this, filepath);
53
54 4) 根据提供的参数, 转换为日期格式返回
55 函数:
56 public static Date StrToDate(String str);
57 返回: 返回Date格式字符, 返回"YYYYMMddhhmmss"格式
58
59
60 #2.apk信息检测<版本>
61 2.app信息检测
62 1) app版本号检测
63 说明: 使用此函数可以获取到apk的版本号
64 函数:
65 public static String GetCurrentVersion(Context context);
66 返回: 返回一个String, 就是apk的当前版本号
67 示例:
68 Tools.GetCurrentVersion(m_parent)
69
70 2) apk当前SDK版本查询
71 说明: 使用此函数可以获取到当前apk的SDK版本
72 函数:
73 public static String get_sosocamlib_version()

```

```

74
75 #3.字符串操作<密码强度>
76
77 1) .判断当前设置的密码强弱
78 说明: 计算密码的
79 函数:
80 public static int passwordStrength(String str)
81 返回: 返回一个int型, 一共有四个等级, 具体如下:
82 0:表示密码长度小于8:
83 1:表示密码由数字, 大写字母, 小写字母, 特殊字符其中一种类型组成。
84 2:表示密码由数字, 大写字母, 小写字母, 特殊字符其中两种类型组成。
85 3:表示密码由数字, 大写字母, 小写字母, 特殊字符其中三种类型组成。
86 4:表示密码由数字, 大写字母, 小写字母, 特殊字符其中四种类型组成。
87
88 示例:
89 int score= Tools.passwordStrength("tonly666666");
90
91 #4.GET URL操作打开网址获取图片或图片数据
92
93 4.GET URL操作[HttpClient类中]
94 1) JSON返回
95 说明: 以下两个函数, 都是用于打开url的。函数1没有提供设置超时时间接口, 默认是10s, 函数2
96 可设置超时时间。
97 函数1:
98 public static JSONObject get_json(String url);
99 url:目标url
100 返回: 返回一个JSONObject对象
101 示例:
102 String url = (m_https?"https://":"http://") + m_ip + ":" + m_port + "/get_params.cgi?"
103             + "user=" + m_user + "&pwd=" + m_pwd + "&json=1&id=";
104 JSONObject json = HttpClient.get_json(url);
105
106 函数2:
107 public static JSONObject get_json(String url, int timeout);
108 /*
109  *url:目标url
110  *timeout:超时时间, 单位ms
111 */
112 返回: 返回一个JSONObject对象
113 示例:
114 String url = (m_https?"https://":"http://") + m_ip + ":" + m_port + "/get_params.cgi?"
115             + "user=" + m_user + "&pwd=" + m_pwd + "&json=1&id=";
116 JSONObject json = HttpClient.get_json(url, 10*1000);
117
118 2) Bitmap返回
119 说明: 打开一个url, 返回一个Bitmap对象。函数1没有提供设置超时时间接口, 默认是10s, 函数2
120 可设置超时时间。
121 函数1:
122 public static Bitmap get_image(String url);
123 示例:
124 Bitmap image = HttpClient.get_image(url);
125
126 函数2:
127 /*
128  *url:目标url
129  *timeout:超时时间, 单位ms
130 */
131 public static Bitmap get_image(String url, int timeout);
132 示例:
133 Bitmap image = HttpClient.get_image(url, 10*1000);
134
135 3) byte[] 返回, 打开一个url, 返回一个byte[]类型的对象。函数1没有提供设置超时时间接口,
136 默认是10s, 函数2可设置超时时间。
137 说明: 打开一个url, 返回一个byte[]类型的对象。
138 函数1:
139 public static byte[] get_binary(String url)
140 示例:
141 byte[] data = HttpClient.get_binary(url);
142
143 函数2:
144 /*
145  *url:目标url
146  *timeout:超时时间, 单位ms

```

```

144 */
145 public static byte[] get_binary(String url, int timeout);
146 示例:
147 byte[] data = HttpClient.get_binary(url, 10*1000);
148
149 函数3:
150 public static byte[] get_gif(String url);
151 byte[] data = HttpClient.get_gif(url);
152
153
154 #5.摄像机id处理<id编码用于jpush tag/soundwave>
155
156 5.摄像机id处理
157 1) 推送摄像机id转义[reecam自己定义]
158 说明: 遇到非字母数字的都转义成与下划线开始后面带他的16进制。消息推送, 以id为tag注册时
159 用, 瑞彩自定义
160 函数: public static String replaceExceptonChar(String s)
161 返回: 返回一个转码后的String
162 示例:
163 Tools.replaceExceptonChar("RTEST-001015-DUGMR"); //返回: RTEST_2d001015_2dDUGMR
164
165 2) 声音添加摄像机id转义
166 说明: 把摄像机id拆分, 转换成两个int. 使用声音添加时, 需要用到。
167 函数:
168 public static int get_name_of_camera_id(String id)
169 //和声音添加中的特征码1配合使用, 用来查找当前声音添加所设置的唯一机器ID
170 public static int get_serial_of_camera_id(String id)
171 //和声音添加中的特征码2配合使用, 用来查找当前声音添加所设置的唯一机器ID
172
173 返回: 都是返回一个int类型的数
174 示例:
175 private SoundWaveWifiSetting m_sound_wave_wifi_setting = new SoundWaveWifiSetting();
176 m_sound_wave_wifi_setting.init(m_parent);
177 public void checkID() {
178     m_camera_id = "RTEST-001015-DUGMR";
179     name_of_camera_id = Tools.get_name_of_camera_id(m_camera_id);
180
181     serial_of_camera_id = Tools.get_serial_of_camera_id(m_camera_id);
182     m_name_of_camera_id = m_sound_wave_wifi_setting.name_of_camera_id();
183
184     m_serial_of_camera_id = m_sound_wave_wifi_setting.serial_of_camera_id();
185     if (name_of_camera_id == m_name_of_camera_id &&
186         serial_of_camera_id == m_serial_of_camera_id) {
187         camera_id = id;
188         Log.e("SoDemo", "Camera id is" + id);
189     }
190 }
191
192 #6.手机录音 <手机录音和播放>
193
194 6.手机录音
195 说明: 以下函数是手机录音和回放的实现
196 1) 手机录音
197 函数:
198 /*
199 *开始手机录音+实现接口record_audio_listener
200 *参数max_seconds: 设置录音时长, 时间到了会自动停止录音。
201 */
202 public static boolean start_record_audio(int max_seconds, record_audio_listener
203 listener);
204 /*
205 *停止手机录音
206 */
207 public static void stop_record_audio();
208 /*
209 *录音结束后的回调, 返回录音数据data
210 */
211 public void on_record_audio_finished(byte[] data);
212
213 2) 手机回放录音文件
214 /*
215 *开始手机录音回放+实现接口play_audio_listener

```

```

213     *参数data: 需要播放的录音数据
214     */
215     public static boolean start_play_audio(byte[] data, play_audio_listener listener);
216
217     /*
218     *停止手机录音回放
219     */
220     public static void stop_play_audio();
221
222     /*
223     *play_audio_listener回调
224     *当前录音文件播放完毕会触发on_play_audio_finished()回调函数,说明当前录音已播放完毕
225     */
226     public void on_play_audio_finished();
227
228     #7二维码图片生成
229     说明: 将WiFi的ssid和psk配置转化为二维码的字符串
230     函数:
231     public static String wifi_setting_2_qrcode_string(String ssid, String psk)
232     /*-
233     参数: ssid: WiFi的ssid名字
234           psk: WiFi的密码
235     返回值: String
236     --*/
237
238     /*-----以下为Wifi.java函数说明-----*/
239     1.Wifi的初始化和反初始化
240     说明: 整个客户端如果需要使用无线,那程序开始时需先进行初始化,程序退出时不再使用后必须
241     反初始化,初始化和反初始化需成对使用
242     函数:
243     public static void init(Context context); //Wifi的初始化,初始化WifiManager等
244     public static void deinit(); //Wifi反初始化,释放相应资源
245
246     2.环境中Wifi热点搜索和连接
247     1) 搜索环境中的Wifi
248     说明: 使用以下函数,可搜索环境中的热点
249     函数:
250     /*
251     *搜索环境中的热点+实现接口scan_listener
252
253     *参数: scan_4_camera: 两个值, false: 搜索环境中所有的AP, true: 搜索处于AP模式下的摄像
254     机
255     * include_list: 保留参数, 设为null
256     * exclude_list: 保留参数, 设为null
257     */
258     public static boolean scan(boolean scan_4_camera, List<String> include_list,
259     List<String> exclude_list, scan_listener listener)
260
261     /*
262     *scan_listener回调
263     *当调用上述scan函数,会触发on_wifi_scan_result(boolean succeed)回调函数
264     */
265     public void on_wifi_scan_result(boolean succeed);
266
267     2) 获取wifi列表
268     说明: 调用scan函数成功,就会返回一个AP对象的列表
269     /*
270     *返回一个列表,列表的每一个元素为AP对象
271     *String bssid: wifi的MAC地址
272     *String ssid: wifi的名字
273     *int quality: wifi的信号强度
274     *WIFI_AUTH auth: wifi的加密方式
275     *WIFI_ENCRYPT encrypt: wifi的加密方式
276     *int wep_key_index: 当wifi加密方式为wep时,16进制还是ASCII索引
277     *String key: wifi的密码
278     */
279     函数:
280     public static ArrayList<AP> get_ap_list()
281
282     3) 获取手机当前连接的wifi名称
283     函数: public static String get_current_connect_ssid()
284     返回: 当前wifi的ssid

```

282 3) 连接wifi

283 说明：连接当前搜索到的某一个AP

284 函数：

```
285 /*
286  *连接当前的AP + 实现connect_listener接口
287  *AP ap: 当前需要连接的目标AP
288  */
289 public static int connect(AP ap, connect_listener listener)
290
291 /*
292  *connect_listener回调
293  *上述connect函数调用成功，会触发 on_wifi_connect_result(boolean succeed) 回调函数
294  */
295 public void on_wifi_connect_result(boolean succeed);
296
297 /*
298  *取消当前wifi的动作，用在当前activity退出时，和deinit函数不同
299  */
300 public static void cancel ()
```

```
301
302
303 示例：
304 Wifi.init(ChooseAPActivity.this); //初始化wifi的相关参数
305 Wifi.scan(false, null, null, ChooseAPActivity.this); //调用搜索函数
```

```
306
307 public ArrayList<AP> m_ap_list = new ArrayList<AP>();
308 private AP m_ap;
309
310 public void on_wifi_scan_result(boolean succeed) {
311     this.m_ap_list = Wifi.get_ap_list(); //获取搜索到的wifi列表
312     m_ap = m_ap_list.get(0);
313 }
314
315 if (0 > Wifi.connect(m_ap, ChooseAPActivity.this)) { //调用连接wifi的函数
316     Log.e("Sodemo", "---Wifi--is--connectting--now-");
317 }
318 public void on_wifi_connect_result(boolean succeed) { //查看连接结果
319     if (succeed) {
320         Log.e("Sodemo", "---Wifi--connect--succeed-");
321     } else {
322         Log.e("Sodemo", "---Wifi--connect--failed-");
323     }
324 }
325
326 protected void onDestroy() { //当前activity退出时，需取消wifi的动作
327     Wifi.cancel ();
328     super.onDestroy ();
329 }
330
331 protected void onDestroy() { //当前应用程序退出时，需饭初始化wifi的动作
332     Wifi.deinit ();
333     super.onDestroy ();
334 }
```