```
//
//  Tools + Wifi + HttpClient.java
//  SoDemon
//

#1.获取Android系统属性参数<语言 + 提示框操作 + 日期处理>

1.获取Android系统属性参数

1）Android系统语言检测
说明：使用以下函数，可以检测出Android当前的语言。
函数：public static String check_language()
返回：返回一个String类型，
      返回"";代表当前是中文（包含简体中文和繁体中文）；
      返回"en"，代表当前是非中文
示例：
if (Tools.check_language().equals(""))
    Log.e("SoDemo","--Current--language--is--Chinese");
else
    Log.e("SoDemo","--Current--language--is-not-Chinese");;

2）定制消息提示框
说明：以下三个函数，是用来提示信息用的，即Android的Toast控件。
函数1：
/*
 *显示提示框，Toast.LENGTH_LONG后自动消失
 *参数tip:要显示的内容
 */
public static void showLongToast(Context context, String tip)

函数2：
/*
 *显示提示框，Toast.LENGTH_SHORT后自动消失
 *参数tip:要显示的内容
 */
public static void showShortToast(Context context, String tip)

函数3：
/*
 *强制取消提示框
 */
 public static void cancelToast()

 3）将当前路径下的文件更新到系统相册
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
函数：
/*
 *path：照片或录像文件等媒体文件的存储路径
 */
public static void add_media(Context context, String path)
示例：
Tools.add_media(LiveActivity.this, filepath);

4）根据提供的参数，转换为日期格式返回
函数：
public static Date StrToDate(String str);
返回：返回Date格式字符，返回"YYYYMMddhhmmss"格式


#2.apk信息检测<版本>
2.app信息检测
1）app版本号检测
说明：使用此函数可以获取到apk的版本号
函数：
public static String GetCurrentVersion(Context context);
返回：返回一个String，就是apk的当前版本号
示例：
Tools.GetCurrentVersion(m_parent)

2）apk当前SDK版本查询
说明：使用此函数可以获取到当前apk的SDK版本
函数：
public static String get_sosocamlib_version()
```

#3.字符串操作<密码强度>

1）.判断当前设置的密码强弱
说明：计算密码的
函数：
```
public static int passwordStrength(String str)
```
返回：返回一个int型，一共有四个等级，详细如下：
0:表示密码长度小于8:
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。
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。
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。
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。

示例：
```
int score= Tools.passwordStrength("tonly666666");
```

#4.GET URL操作打开网址获取图片或图片数据

4.GET URL操作[HttpClient类中]
1)JSON返回
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可设置超时时间。
函数1:
```
public static JSONObject get_json(String url);
```
url:目标url
返回：返回一个JSONObject对象
示例：
```
String url = (m_https?"https://":"http://") + m_ip + ":" + m_port + "/get_params.cgi?"
                + "user=" + m_user + "&pwd=" + m_pwd + "&json=1&id=";
JSONObject json = HttpClient.get_json(url);
```

函数2:
```
public static JSONObject get_json(String url, int timeout);
/*
 *url:目标url
 *timeout:超时时间，单位ms
*/
```
返回：返回一个JSONObject对象
示例：
```
String url = (m_https?"https://":"http://") + m_ip + ":" + m_port + "/get_params.cgi?"
                + "user=" + m_user + "&pwd=" + m_pwd + "&json=1&id=";
JSONObject json = HttpClient.get_json(url, 10*1000);
```

2）Bitmap返回
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可设置超时时间。
函数1:
```
public static Bitmap get_image(String url);
```
示例：
```
Bitmap image = HttpClient.get_image(url);
```

函数2:
```
/*
 *url:目标url
 *timeout:超时时间，单位ms
*/
public static Bitmap get_image(String url, int timeout);
```
示例：
```
Bitmap image = HttpClient.get_image(url, 10*1000);
```

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，默认是10S，函数2可设置超时时间。
说明：打开一个url,返回一个byte[]类型的对象。
函数1:
```
public static byte[] get_binary(String url)
```
示例：
```
byte[] data = HttpClient.get_binary(url);
```

函数2:
```
/*
 *url:目标url
 *timeout:超时时间，单位ms
```

```
*/
public static byte[] get_binary(String url, int timeout);
示例：
byte[] data = HttpClient.get_binary(url, 10*1000);

函数3：
public static byte[] get_gif(String url);
byte[] data = HttpClient.get_gif(url);


#5.摄像机id处理<id编码用于jpush tag/soundwave>

5.摄像机id处理
1）推送摄像机id转义[reecam自己定义]
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注册时用,瑞彩自定义
函数：public static String  replaceExceptonChar(String s)
返回：返回一个转码后的String
示例：
Tools.replaceExceptonChar("RTEST-001015-DUGMR");  //返回：RTEST_2d001015_2dDUGMR

2）声音添加摄像机id转义
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。
函数：
public static int get_name_of_camera_id(String id)
//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
public static int get_serial_of_camera_id(String id)
//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

返回：都是返回一个int类型的数
示例：
private SoundWaveWifiSetting m_sound_wave_wifi_setting = new SoundWaveWifiSetting();
m_sound_wave_wifi_setting.init(m_parent);
public void checkID() {
        m_camera_id = "RTEST-001015-DUGMR";
        name_of_camera_id = Tools.get_name_of_camera_id(m_camera_id);

        serial_of_camera_id = Tools.get_serial_of_camera_id(m_camera_id);
        m_name_of_camera_id = m_sound_wave_wifi_setting.name_of_camera_id();

        m_serial_of_camera_id = m_sound_wave_wifi_setting.serial_of_camera_id();
        if (name_of_camera_id == m_name_of_camera_id &&
            serial_of_camera_id == m_serial_of_camera_id) {
            camera_id = id;
            Log.e("SoDemo","Camera id is" + id);
        }
    }

#6.手机录音 <手机录音和播放>

6.手机录音
说明：以下函数是手机录音和回放的实现
1）手机录音
函数：
/*
 *开始手机录音＋实现接口record_audio_listener
 *参数max_seconds：设置录音时长，时间到了会自动停止录音。
 */
public static boolean start_record_audio(int max_seconds, record_audio_listener
listener);
/*
 *停止手机录音
 */
public static void stop_record_audio();
/*
 *录音结束后的回调，返回录音数据data
 */
public void on_record_audio_finished(byte[] data);


2）手机回放录音文件
/*
 *开始手机录音回放＋实现接口play_audio_listener
```

```
 *参数data：需要播放的录音数据
 */
public static boolean start_play_audio(byte[] data, play_audio_listener listener);

/*
 *停止手机录音回放
 */
public static void stop_play_audio();

/*
 *play_audio_listener回调
 *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
 */
public void on_play_audio_finished();

#7二维码图片生成
说明：将WiFi的ssid和psk配置转化为二维码的字符串
函数：
public static String wifi_setting_2_qrcode_string(String ssid, String psk)
/*-
参数：ssid：WiFi的ssid名字
      psk：WiFi的密码
返回值：String
--*/

/*------------以下为Wifi.java函数说明---------------------*/
1.Wifi的初始化和反初始化
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反初始化，初始化和反初始化需成对使用
函数：
public static void init(Context context); //Wifi的初始化，初始化WifiManager等
public static void deinit();  //Wifi反初始化，释放相应资源

2.环境中Wifi热点搜索和连接
1）搜索环境中的Wifi
说明：使用以下函数，可搜索环境中的热点
函数：
/*
 *搜索环境中的热点+实现接口scan_listener

 *参数：scan_4_camera：两个值，false：搜索环境中所有的AP，true：搜索处于AP模式下的摄像机
 *      include_list：保留参数，设为null
 *      exclude_list：保留参数，设为null
 */
public static boolean scan(boolean scan_4_camera, List<String> include_list, List<String> exclude_list, scan_listener listener)

/*
 *scan_listener回调
 *当调用上述scan函数，会触发on_wifi_scan_result(boolean succeed)回调函数
*/
public void on_wifi_scan_result(boolean succeed);

2）获取wifi列表
说明：调用scan函数成功，就会返回一个AP对象的列表
/*
 *返回一个列表，列表的每一个元素为AP对象
 *String bssid：wifi的MAC地址
 *String ssid：wifi的名字
 *int quality：wifi的信号强度
 *WIFI_AUTH auth：wifi的加密方式
 *WIFI_ENCRYPT encrypt：wifi的加密方式
 *int wep_key_index：当wifi加密方式为wep时，16进制还是ASCII索引
 *String key：wifi的密码
 */
函数：
public static ArrayList<AP> get_ap_list()

3）获取手机当前连接的wifi名称
函数：public static String get_current_connect_ssid()
返回：当前wifi的ssid
```

3）连接wifi
说明：连接当前搜索到的某一个AP
函数：

```
/*
 *连接当前的AP + 实现connect_listener接口
 *AP ap: 当前需要连接的目标AP
 */
public static int connect(AP ap, connect_listener listener)

/*
 *connect_listener回调
 *上述connect函数调用成功，会触发 on_wifi_connect_result(boolean succeed)回调函数
 */
public void on_wifi_connect_result(boolean succeed);

/*
 *取消当前wifi的动作，用在当前activity退出时，和deinit函数不同
 */
public static void cancel()
```

示例：

```
Wifi.init(ChooseAPActivity.this); //初始化Wifi的相关参数
Wifi.scan(false, null, null, ChooseAPActivity.this); //调用搜索函数

public ArrayList<AP> m_ap_list = new ArrayList<AP>();
private AP m_ap;

public void on_wifi_scan_result(boolean succeed) {
    this.m_ap_list = Wifi.get_ap_list();//获取搜索到的wifi列表
    m_ap = m_ap_list.get(0);
    }

if (0 > Wifi.connect(m_ap, ChooseAPActivity.this)){//调用连接wifi的函数
    Log.e("Sodemo","---Wifi--is--connectting--now-");
}
public void on_wifi_connect_result(boolean succeed) {//查看连接结果
    if (succeed) {
        Log.e("Sodemo","---Wifi--connect--succeed-");
    }else{
        Log.e("Sodemo","---Wifi--connect--failed-");
    }
}

protected void onDestroy() {//当前activity退出时，需取消wifi的动作
        Wifi.cancel();
        super.onDestroy();
    }

protected void onDestroy() {//当前应用程序退出时，需饭初始化wifi的动作
        Wifi.deinit();
        super.onDestroy();
    }
```