```
//
//  IPCamMgr.java
//
IPCamMgr
使用此SDK必须要使用IPCamMgr类，IPCamMgr类用于批量管理列表中所有的cam

#1.1SDK初始化和反初始化
    1.1SDK初始化和反初始化

        说明：使用此sdk必须先初始化，才能使用其他函数，不再使用此sdk时也必须要反初始化
        。
        函数：下面两个函数，必须成对出现，有初始化必须要有反初始化；
        public static void init(Context
        context)//创建IPCamMgr初始化，并开始批量管理cams，开始连接cams；
        public static void deinit();//释放IPCamMgr，反初始化，停止连接cams；

        示例：
        IPCamMgr.init(this); //开始使用此SDK时  先初始化SDK,开始连接cams;
        IPCamMgr.deinit(); //不在使用此SDK时 反初始化SDK,并停止连接cams;

#1.2局域网搜索
    1.2局域网搜索
        说明：只要创建了IPCamMgr,程序就一直在后台搜素局域网中的cams
        /*
         *返回的是DISCOVER_CAMERA_INFO类型的对象：
         *String id;   //摄像机id
         *int id_type; //摄像机id_type
         *String alias;    //设备的别名
         *String fw_version;  //系统固件版本
         *String ui_version;  //网页固件版本
         *int model;  //摄像机类型，两个值，
         *String current_ip; //设备当前的IP地址；
         *int port;   //设备当前的端口；
         *boolean https;  //设备当前是否安全加密传输，true:加密  false:不加密
         *boolean used;   //当前设备是否已被添加到IPCamMgr中，true:已被添加
         false:未被添加
        */
        函数：public static LinkedHashMap<String, DISCOVERED_CAMERA_INFO>
        get_discovered_cameras_list();//获取局域网里的cams
              public static void
              rediscover_cameras();//重新搜索局域网中的摄像机，结果从get_discovered_ca
              meras_list()方法中获取
        示例：
        LinkedHashMap<String, DISCOVERED_CAMERA_INFO> ipcam_list;
        ipcam_list = IPCamMgr.get_discovered_cameras_list();
        for(DISCOVERED_CAMERA_INFO ipcam : ipcam_list.values())
        {
            if(!ipcam.used)
                Log.e("SoDemo","Current is not used");
        }

#1.3获取IPCamMgr类的cams
    1）获取已添加到IPCamMgr类中的cams

    说明：获取被添加到IPCamMgr的所有cams，通过函数返回一个LinkedHashMap，里面每一个元
    素是一个IPCam对象
    函数：public static LinkedHashMap<String, IPCam> get_cameras_list();
    示例：
    LinkedHashMap<String, IPCam> ipcam_list = IPCamMgr.get_cameras_list();;
    for(IPCam ipcam : ipcam_list.values())
    {
        Log.e("SoDemo","---The id of ipcam in IPCamMgr is:" + ipcam.id());
    }

    2）通过id获取IPCamMgr中的cam
    说明：
    函数：public static IPCam get_camera(String id);//通过摄像机id获取IPCam对象
    示列：
    IPCam ipcam = IPCamMgr.get_camera("RTEST-001566-ZSIOH");

#1.4添加和删除IPCamMgr中的cam以及相关回调

    1.4.1添加和删除回调
```

说明：添加回调public interface IPCamMgr_Listener;当IPCamMgr里的cam有变动时，就会回调函数：

两个函数：

```
public static void add_listener(IPCamMgr_Listener listener);//添加IPCamMgr_Listener回调
public static void remove_listener(IPCamMgr_Listener listener);//删除IPCamMgr_Listener回调
```

三个回调：

```
public void on_camera_added(String id);//添加IPCamMgr的cam的回调，且返回camera_id
public void on_camera_removed(String id);//删除IPCamMgr的cam的回调，且返回camera_id
public void on_cameras_cleared();//清空IPCamMgr所有的cams的回调
```

示例：

```
IPCamMgr.add_listener(this);//先添加回调
//回调响应
public void on_camera_added(String id)
{
    Log.e("SoDemo","current added camera_id is::" + id);
}
public void on_camera_removed(String id)
{
    Log.e("SoDemo","current delete camera_id is::" + id);
}
public void on_cameras_cleared()
{
    Log.e("SoDemo","all cameras is removed");
}
IPCamMgr.remove_listener(this);;//不需要时 删除回调
```

### 1.4.2添加

说明：如果想要批量管理cam，添加摄像机到本地后，必须要添加摄像机到IPCamMgr，添加完以后返回一个IPCam对象

函数：public static IPCam add_camera(String alias, String id, String user, String pwd, boolean https);//添加摄像机到IPCamMgr*

示例：

```
LinkedHashMap<String, DISCOVERED_CAMERA_INFO> ipcam_list = IPCamMgr.get_discovered_cameras_list();
DISCOVERED_CAMERA_INFO m_ipcam;
for(DISCOVERED_CAMERA_INFO ipcam : ipcam_list.values())
{
    if(!ipcam.used){
        Log.e("SoDemo","Current is not used");
        m_ipcam = ipcam;
        break;
    }
}
CAMERA_INFO add_cam = new CAMERA_INFO();
add_cam.setObj_id(sosocam_camera_id);
add_cam.setAlias(m_ipcam.alias);
add_cam.setId(m_ipcam.id);
add_cam.setUser("admin");
add_cam.setPwd("ipcam_pwd");
add_cam.setHttps(m_ipcam.https);
add_cam.setModel(m_ipcam.model);
Storage.add_camera(add_cam);//添加摄像机到本地
Storage.save_cameras();
IPCam ipcam = IPCamMgr.add_camera(m_ipcam.alias, m_ipcam.id,
 "admin", "ipcam_pwd", m_ipcam.https);      //添加摄像机到IPCamMgr
```

### 1.4.3删除IPCamMgr类的cam

#### 1）删除IPCamMgr类里指定的cam

说明：这个函数一般删除本地摄像机后，需要从IPCamMgr类里删除

函数：public static IPCam remove_camera(String id);//删除IPCamMgr类里id摄像机

示例：

```
Storage.remove_camera("RTEST-001566-ZSIOH");
Storage.save_cameras();;//从本地删除:"RTEST-001566-ZSIOH"摄像机
```

```
        IPCamMgr.remove_camera("RTEST-001566-ZSIOH");//从IPCamMgr删除:"RTEST-001566-ZSIOH"摄像机
129
130     2）删除IPCamMgr里的所有cams
131
            说明：当不需要再对cams做管理的时（如账号切换的时候），需要删除IPCamMgr里的所有cams
132         函数：public static void clear_cameras_list();
133         示例：
134         IPCamMgr.clear_cameras_list();
135
136 # 1.5更新cam的密码到IPCamMgr
137     说明：一般摄像机密码做了修改，需要保存到本地，同时保存到IPCamMgr类
138     函数：public static void update_camera_pwd(String id, String pwd);
139     示例：
140     public ERROR reset_pwd(String pwd, reset_pwd_listener.this)//修改摄像机的密码
141     Storage.update_camera_pwd(m_ipcam.id(), m_ipcam.pwd());
142     Storage.save_cameras();;//更新密码到本地
143     IPCamMgr.update_camera_pwd(m_ipcam.id(),
        m_edittext_pwd.getText().toString());;//更新密码到IPCamMgr，此处更新的是用来连接访问的密码
144
145 # 1.6立即连接
146
        说明：立即连接是指手动断开cam然后重新连接，用这个函数摄像机必须是添加到了IPCamMgr类
147     函数：public static void reset_camera(String id);
148     示例：
149     IPCamMgr.reset_camera("RTEST-001566-ZSIOH");
150
151 # 1.7重新初始化
152
        说明：手动重新初始化camlib库，和need_restart()配合使用，用于当程序从后台回到前台时，重新初始化camlib库，一般在onResume()中调用
153     函数：public static void restart();
154           public static boolean need_restart();
155     示例：
156     if (IPCamMgr.need_restart())
157         IPCamMgr.restart();
158
159 # 1.8一次性摄像机管理
160     说明：一次性管理摄像机，主要用于收藏的摄像机管理，一次性摄像机数量小于等于1
161     函数：public static IPCam get_disposable_camera();//获取一次性摄像机
162     public static boolean set_disposable_camera(String id, String user, String pwd,
        boolean https);//设定一次性摄像机
163     public static void clear_disposable_camera();//清除一次性摄像机
164     public static boolean update_disposable_camera_pwd(String
        pwd);//更新一次性摄像机密码到IPCamMgr
165     public static boolean reset_disposable_camera();//重连一次性摄像机
166     示例：
167     IPCamMgr.set_disposable_camera(id, user, pwd, https)
168     IPCamMgr.get_disposable_camera();
169     IPCamMgr.update_disposable_camera_pwd(pwd);
170     IPCamMgr.reset_disposable_camera();
171     IPCamMgr.clear_disposable_camera();
172
173 # 1.9数据(moblie)联网下的连接设置
174
        说明：把forbidden_in_mobile参数设置到本地后，必须要设置到IPCamMgr才生效.针对所有摄像机，缺省值是数据流量下允许连接。
175     函数：public static void set_forbidden_in_mobile(boolean forbidden);
176     示例：
177     boolean forbidden_mobile =
        Storage.forbidden_with_mobile();//从本地获取get_forbidden_with_mobile值（默认值为NO）设置到本地
178     Storage.set_forbidden_with_mobile(forbidden_mobile);//从m_forbidden_mobile值到本地
179
        IPCamMgr.set_forbidden_in_mobile(forbidden_mobile);//把m_forbidden_mobile设置到IPCamMgr生效
180
181 # 1.10 获取到当前网络类型
182     说明：获取到当前网络类型，返回网络类型
183     public static enum NETWORK_TYPE {
```

```
        DOWN, //无网络
        WIFI, //WIFI网络
        MOBILE;//数据流量
    }
    函数：public static NETWORK_TYPE get_network_type();
    示例：
    m_network_type = IPCamMgr.get_network_type();

# 1.11 摄像机后台连接设置
    说明：设置当前联网状态下后台是否保持对摄像机的连接
    函数：public static void set_keepalive_in_wifi(boolean
    keepalive);//设置wifi联网下在后台是否保持摄像机的连接，true：保持，false:不保持
          public static void set_keepalive_in_mobile(boolean
          keepalive);//设置mobile联网下在后台是否保持摄像机的连接，true：保持，false:
          不保持
    示例：
    IPCamMgr.set_forbidden_in_wifi(true);
    IPCamMgr.set_forbidden_in_mobile(true);
```