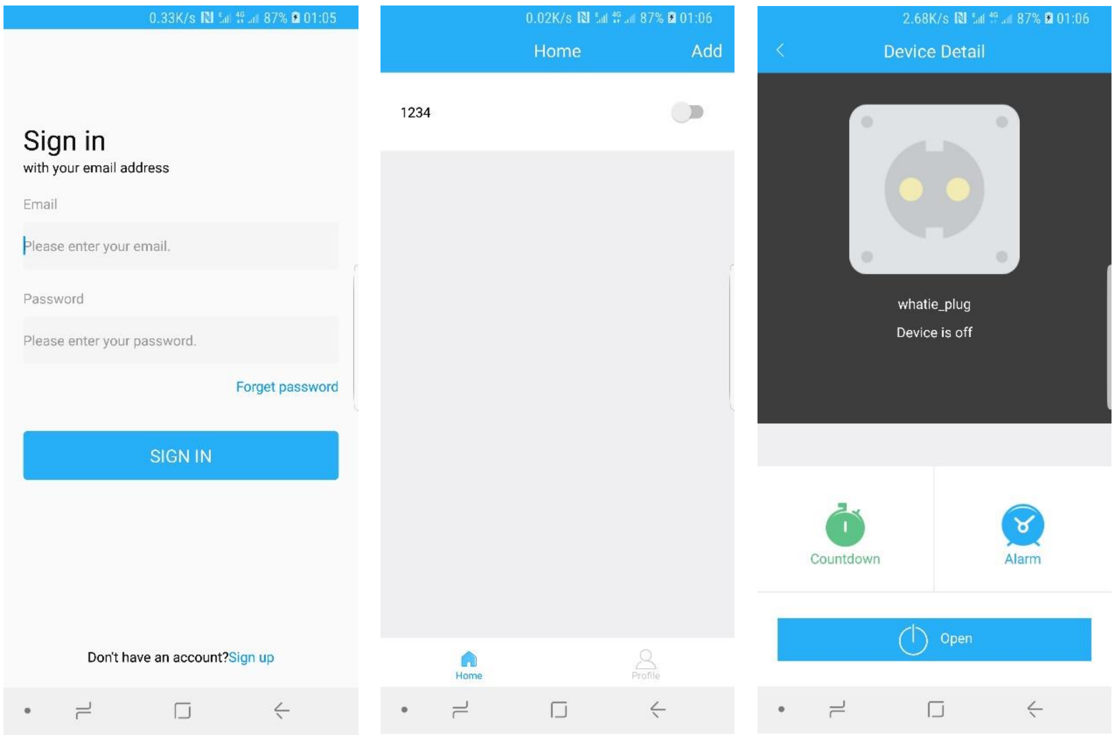
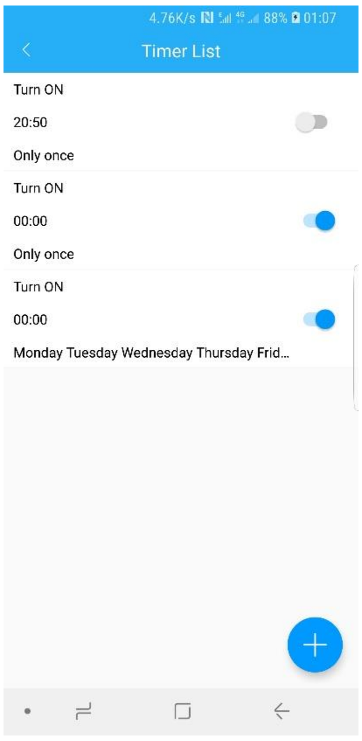
WahtieSDK for Android updated at 2018-06-09

```
What's new:
All APIs for electrical outlets. SDKs for bulbs will be provided about June 12, 2018.
```

WhatieSDK is a SDK provided by ATI TECHNOLOGY (WUHAN) CO.,LTD. for the 3rd party accessing to our IOT cloud platform easily and quickly. Using this SDK, developers can do almost all function points on electrical outlets and RGBW bulbs (to be uploaded on June 12), such as user registration/login/logout, smart configuration, add/share/remove devices, device control, timing countdown, timer, etc.



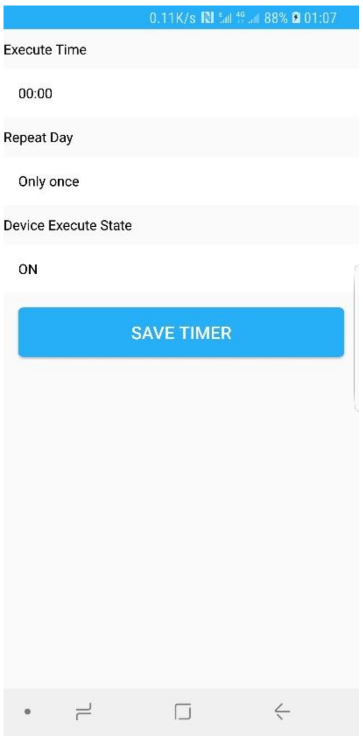
Note: For all function points, no any backend development on cloud platform is needed for integrating the SDK into your APP. You just do all your work in your APP side.



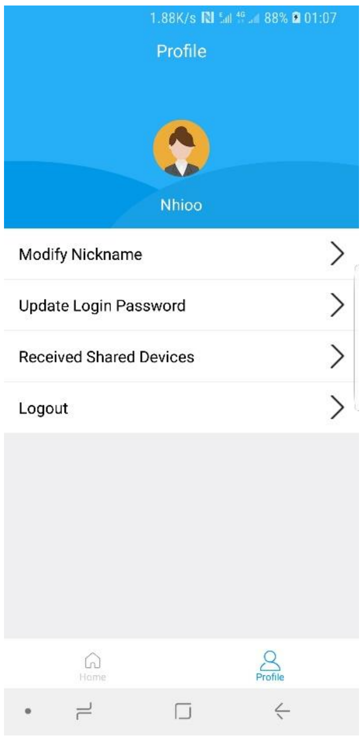
Turn ON
20:50
Only once

Turn ON
00:00
Only once

Turn ON
00:00
Monday Tuesday Wednesday Thursday Frid...



Execute Time
00:00
Repeat Day
Only once
Device Execute State
ON



1. Preparation

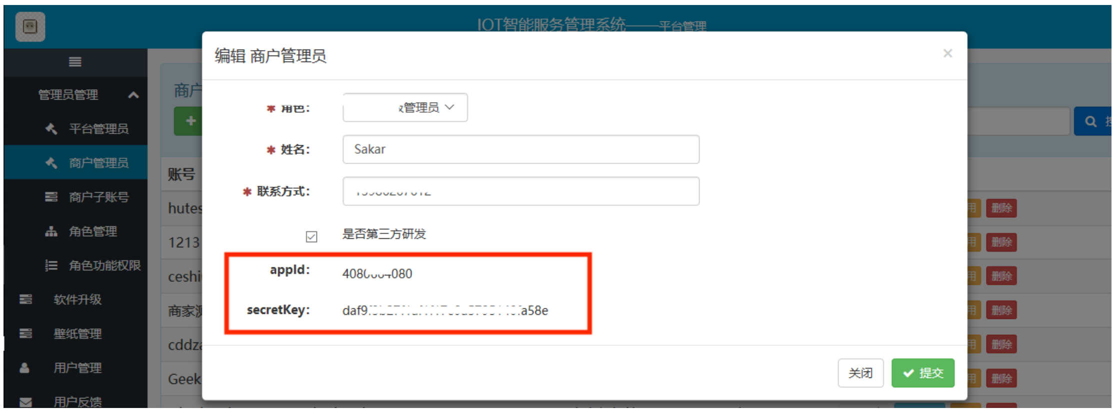
Sign up a developer account

Sign up a 3rd party developer account at ATI cloud platform to create self-developed products, create function points, and so on.

Note: We have signed up an account for SAKAR, which has been emailed to SAKAR. SAKAR can just skip this step.

Obtain appId and secretKey

Go to Development Platform - Application Management - Create a new application to obtain an appId and secretKey to initialize SDKs (for both Android and iOS).



编辑 商户管理员

* 角色:	管理员
* 姓名:	Sakar
* 联系方式:	[redacted]
☑	是否第三方研发
appId:	408[redacted]080
secretKey:	daf9[redacted]a58e

Note: We have applied appId and secretKey for SAKAR, which has been emailed to SAKAR. SAKAR can just skip this step.

SDK Demo

SDK Demo is a complete APP incorporating the main flows and operations such as registration, login, sharing, feedback, network configuration and device control, etc. The Demo code can be used as a good reference for the 3rd party development. Download link

---

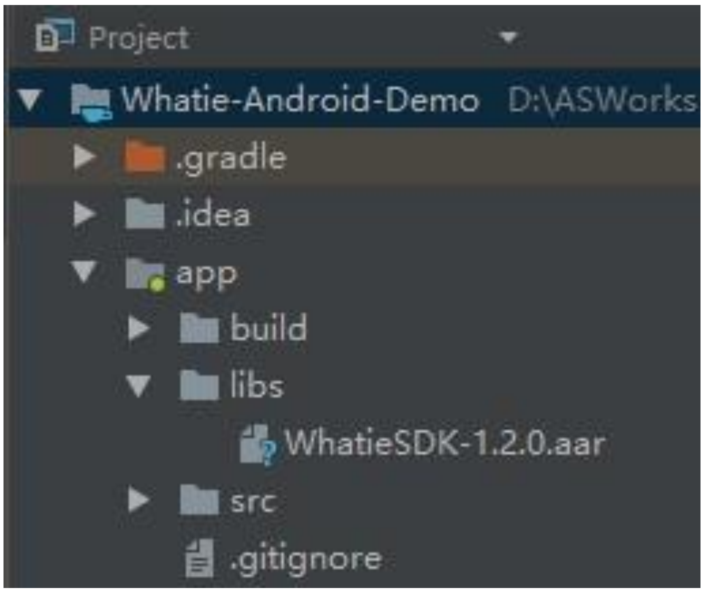
2. SDK Import and Configuration

Requirements

- IDE：Android Studio

Import the .aar lib package

Create a libs directory in your Android Studio project. Copy the downloaded WhatieSDK-xxxx.aar into that directory. For example, as shown in the following figure, a demo project is created, and the .aar package is copied to the libs directory.



Configure the build.gradle

Add the following configuration code in file build.gradle to include the .aar lib package and some 3rd-party plugins.

```
compile(name:'WhatieSDK-x.x.x', ext:'aar')

repositories {
            flatDir {
                 dirs 'libs'
            }
```

```
    }

compile 'com.mylhyl:zxingscanner:2.0.0'      //encode and decode QRcode
compile 'com.lzy.net:okgo:3.0.4'                   // HTTP connection
compile 'com.alibaba:fastjson:1.2.20'          //json2object
compile 'org.eclipse.paho:org.eclipse.paho.client.mqttv3:1.1.0'         //about mqtt
compile 'org.eclipse.paho:org.eclipse.paho.android.service:1.1.1'      //about mqtt
compile 'org.greenrobot:eventbus:3.0.0'             // communications among threads
```

Configure the AndroidManifest.xml

Configure appId and secretKey in file AndroidManifest.xml, and configure the appropriate permissions, etc.

```
<application>
       <!— "\ " before appId! -->
       <meta-data
           android:name="appId"
           android:value="\ appId" />
       <meta-data
           android:name="secretKey"
           android:value="appSecretKey" />
</application>


<!—necessary permissions -->
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
<uses-permission android:name="android.permission.CHANGE_WIFI_MULTICAST_STATE" />
<uses-permission android:name="android.permission.CHANGE_WIFI_STATE" />
<uses-permission android:name="android.permission.WAKE_LOCK"/>
<uses-permission android:name="android.permission.CAMERA"/>

<!—necessary services -->
<service android:name="org.eclipse.paho.android.service.MqttService"/>
<service android:name="com.d9lab.ati.whatiesdk.mqtt.MyMqttService"/>
<service android:name="com.d9lab.ati.whatiesdk.tcp.TcpService"/>
<service android:name="com.d9lab.ati.whatiesdk.udp.UdpService"/>
```

Initialize the Whatie SDK in the application

[Description]

This is mainly used to initialize EventBus, communication services and other components.

[Sample Code]

```
public class DemoApplication extends Application {

    @Override
    public void onCreate() {
        super.onCreate();
        EHomeInterface.getINSTANCE().init(this);
    }
}
```

[Cautions]

While appId and secretKey should be configured in file AndroidManifest.xml, or in the build environment configuration, they can also be written in the code.

---

3. User Management

The SDK provides user management functions, such as user registration, user login, user logout, login password update, and change nickname.

Note:

1. all other information on user management procedure is not needed for SDK.
2. The user email and password ciphertext will be also stored in our cloud platform.
3. No any backend development (on cloud side) is needed for integrating the SDK into your APP.

3.1 User registration

Note: The following example code is a successful call of the registration method. After registration, user logins automatically, and it is unnecessary to call the login method anymore.

Email registration

No verification code is required during email registration. Users may register their accounts directly using their emails.

[Sample Code]

```
/**
     *
     * @param tag           context
     * @param email         email account name
     * @param passwd        account password
     * @param callback
     */
EHomeInterface.getINSTANCE().registerAccountWithEmail(mContext,
    etEmail.getText().toString().trim(),
    etPwd.getText().toString().trim(),
    new UserCallback() {
        @Override
        public void onSuccess(Response<BaseModelResponse<User>> response) {
    }
          @Override
          public void onError(Response<BaseModelResponse<User>> response) {

  });
```

3.2 User login

Upon a successful call, the user’s session will be stored locally by the SDK. When the app is launched next time, the used is logged in by default and no more login process is required.

The session will timeout if the app remains unused for a long time. In this case, the notification of the expired session should be processed to ask the user to log in again.

Email login

[Sample Code]

```
/**
         * ensure single sign-on, or the background service will report a mistake
         * @param mContext    The activity that uses this method
         * @param email           user’s email
         * @param password    user’s password
         * @param callback       callback of network cummunicaiton
         */
EHomeInterface.getINSTANCE().loginWithEmail(mContext, email, password,
                            new UserCallback() {
           @Override
           public void onSuccess(Response<BaseModelResponse<User>> response) {
           if (response.body().isSuccess()) {
                EHome.getInstance().setLogin(true);
                EHome.getInstance().setmUser(response.body().getValue());
                EHome.getInstance().setToken(response.body().getToken());
                SharedPreferenceUtils.put(mContext,
                                Code.SP_MD5_PASSWORD, MD5Utils.encode(password));
            } else {
            if(response.body().getMessage()!=null||!response.body().getMessage()
                .isEmpty()) {
              Toast.makeText(mContext,response.body().getMessage(),
Toast.LENGTH_SHORT).show();
                } else {
              Toast.makeText(mContext, "login fail", Toast.LENGTH_SHORT).show();
                        }
                }
        }
                @Override
        public void onError(Response<BaseModelResponse<User>> response) {
            super.onError(response);
            if(response.body().getMessage()!=null||!response.body().getMessage().isEmpty()) {
            Toast.makeText(mContext,response.body().getMessage(),
Toast.LENGTH_SHORT).show();
        } else {
            Toast.makeText(mContext, "login fail", Toast.LENGTH_SHORT).show();
          }
      }
    });
```

3.3 Password reset by users

Password reset with an email address

If you forget your password, you can reset your password with the e-mail address consisting of 3 steps:

- Sending a verification code to the mailbox

[Sample Code]

```
/**
  *
  * @param tag
```

```
 * @param email
 * @param callback
 */
EHomeInterface.getINSTANCE().sendVerifyCodeByEmail(mContext,          email,          new
BaseCallback() {
            @Override
            public void onSuccess(Response<BaseResponse> response) {
            }
        @Override
            public void onError(Response<BaseResponse> response) {
                super.onError(response);
            }
        });
```

- Get and check the verification code

```

/**
 *
 * @param tag
 * @param email
 * @param verifyCode
 * @param callback
 */
EHomeInterface.getINSTANCE().checkVerifyCode(mContext, email, verifyCode,
                new BaseCallback() {
                    @Override
                    public void onSuccess(Response<BaseResponse> response) {
                        if (response.body().isSuccess()) {

                        } else {

                        }
                    }
                    @Override
                    public void onError(Response<BaseResponse> response) {
                        super.onError(response);
                    }
                });/**
```

- Reset the password

```
  /**
     *
     * @param tag
```

```
         * @param email
         * @param password
         * @param callback
         */
EHomeInterface.getINSTANCE().resetPasswordByEmail(mContext,    email,    md5password,
new BaseCallback() {
                    @Override
                    public void onSuccess(Response<BaseResponse> response) {
                         Toast.makeText(mContext,      "Set      new      password      success.",
Toast.LENGTH_SHORT).show();
                    }

                    @Override
                    public void onError(Response<BaseResponse> response) {
                         super.onError(response);
                         Toast.makeText(mContext,      "Set      new      password      failed.",
Toast.LENGTH_SHORT).show();
                    }
               });
```

3.4 Password reset by old password

If you just want to update your password, you can reset your password with the e-mail address and old password

[Sample Code]

```
 /**
 *
 * @param mContext
 * @param email
 * @param oldPwd
 * @param newPwd
 * @param baseCallback
 */
EHomeInterface.getINSTANCE().changePassword (mContext, email, oldPwd, newPwd ,
               new BaseCallback() {
                    @Override
                    public void onSuccess(Response<BaseResponse> response) {
                    }
                    @Override
```

```
                public void onError(Response<BaseResponse> response) {
                        super.onError(response);
                }
        });
```

3.4 Updating a user's device list

Update the user's current device list.

[Sample Code]

```
/**
  * After get device list, "saveDevices" method must be called.
  * @param mContext
  * @param callback
  */
EHomeInterface.getINSTANCE().getMyDevices(mContext , new DevicesCallback() {
        @Override
        public void onSuccess(Response<BaseListResponse<DeviceVo>> response) {
                if (response.body().isSuccess()){
                        EHomeInterface.getINSTANCE().saveDevices(response.body().getList());
                }
        }

        @Override
        public void onError(Response<BaseListResponse<DeviceVo>> response) {
                super.onError(response);
        }
});
```

3.5 Update the nickname

The user can change user name by update the nickname.

[Sample Code]

```
/**
  *
  * @param tag           context
  * @param nickName    new nickname
  * @param callback
  */
```

```
EHomeInterface.getINSTANCE().modifyNickname(mContext, nickName,
  new UserCallback() {
                    @Override
                    public void onSuccess(Response<BaseModelResponse<User>> response) {
                                    Toast.makeText(mContext,     "Change     nickname     success!.",
Toast.LENGTH_SHORT).show();
                                            }
                        @Override
                                            public  void  onError(Response<BaseModelResponse<User>>
response) {
                                                    super.onError(response);
                                            }
                                    });
```

3.6 Logout

The user can logout the APP by the method as below.

[Sample Code]

```
/**
  *   EHome.getInstance().logOut() must be called when logout success.
  * @param mContext
  * @param baseCallback
  */
EHomeInterface.getINSTANCE().logOut(mContext,
      new BaseCallback() {
        @Override
        public void onSuccess(Response<BaseResponse> response) {
                EHome.getInstance().logOut();  //  This  method  must  be  called  when  logout
success.
        }
      @Override
      public void onError(Response<BaseResponse> response) {
                super.onError(response);
        }
});
```

4. SmartConfig and Device Init

4.1 SmartConfig

You can complete your SmartConfig by the following 4 steps.

(1) Define inner class “WhatieAsyncTask” in activity to config network.

[Sample Code]

```
private class WhatieAsyncTask extends AsyncTask<String, Void, List<IEsptouchResult>> {

        @Override
        protected void onPreExecute() {
        }

        @Override
        protected List<IEsptouchResult> doInBackground(String... params) {
                int taskResultCount = -1;
                synchronized (mLock) {
                        String apSsid = mWifiAdmin.getWifiConnectedSsidAscii(params[0]);
                        String apBssid = params[1];
                        String apPassword = params[2];
                        String taskResultCountStr = params[3];
                        taskResultCount = Integer.parseInt(taskResultCountStr);
                        mEsptouchTask = new EsptouchTask(apSsid, apBssid, apPassword,
mContext);
                }
                List<IEsptouchResult> resultList =
mEsptouchTask.executeForResults(taskResultCount);
                return resultList;
        }
        @Override
        protected void onPostExecute(List<IEsptouchResult> result) {
                IEsptouchResult firstResult = result.get(0);
                if (!firstResult.isCancelled()) {
                        if (firstResult.isSuc()) {

                        } else {

                        }
                }
        }
}

```

(2) Get network token by getNetToken:

[Sample Code]

```
/**
        *
        * @param tag
        * @param baseStringCallback
        */
EHomeInterface.getINSTANCE().getNetToken(mContext, new BaseStringCallback() {
                    @Override
                    public void onSuccess(Response<BaseModelResponse<String>> response) {
                        if (response.body().isSuccess()){

                        }
                    }

                    @Override
                    public void onError(Response<BaseModelResponse<String>> response) {
                        super.onError(response);

                    }
                });
```

(3) After getting SmartConfig token:

[Sample Code]

```
private EspWifiAdminSimple mWifiAdmin = new EspWifiAdminSimple(this);
String apSsid = mWifiAdmin.getWifiConnectedSsid();
String apBssid = mWifiAdmin.getWifiConnectedBssid();

/**
  * Must be connected to 2.4 G Wi-Fi,
* and router, mobile phone and device are close enough
  * @param apSsid
  * @param apBssid
  * @param tokenAndPwd      token + router's password
  */

new WhatieAsyncTask().execute(apSsid, apBssid, tokenAndPwd, "1");
```

(4) Return message of the config device procedure:

Bind Success:
Receiving the MqttBindSuccessEvent event.
Bind failed:
Receiving the MqttAlreadyBindEvent event means the device has been bound by others.

4.2 Device Init

You can initialize your device once you have a successful device binding (i.e., you receive a MqttBindSuccessEvent message).

[Sample Code]

```
/**
        *
        * @param tag
        * @param devId
        * @param name
        * @param baseCallback
        */
EHomeInterface.getINSTANCE().getStarted(mContext, event.getDevId(), event.getName(),
                                    new BaseCallback() {
                @Override
                public void onSuccess(Response<BaseResponse> response) {
                            if(response.body().isSuccess()){

                               }
                @Override
                public void onError(Response<BaseResponse> response) {
                                            super.onError(response);
                }
});
```

5.Device

5.1 OnOff Device

You can turn on/off the device by the following method.

[Sample Code]

```
/**
  * Turn on or turn off outlets.
  * @param devId       devId of device
  * @param status       true is On, false is Off
  */
EHomeInterface.getINSTANCE().updateOutletsStatus(devId, status);
}
```

5.2 Rename device

The device name can be renamed.

[Sample Code]

```
/**
  *
  * @param tag                          context
  * @param devId                        devId of device
  * @param newName                       new device name
  * @param devicesCallback
  */
EHomeInterface.getINSTANCE().updateDeviceName(mContext,
deviceVo.getDevice().getDevId(), newName,
                        new BaseCallback() {
                                        @Override
                                        public void onSuccess(Response<BaseResponse> response) {
                                                        if (response.body().isSuccess()) {
                                                                        Toast.makeText(mContext, "Change name
success.", Toast.LENGTH_SHORT);
                                                                } else {
                                                                        Toast.makeText(mContext,
response.body().getMessage(), Toast.LENGTH_SHORT).show();
                                                                }
                                                        }
                                                        @Override
                                        public     void     onError(Response<BaseResponse>
response) {
                                             super.onError(response);
                                             Toast.makeText(mContext, "Change name fail.",
Toast.LENGTH_SHORT).show();
                                        }
                                   });
```

5.3 Unbind device

You can unbind your device. If the device is unbound, the device is reset to network-pending state.

[Sample Code]

```
/**
      *
      * @param devId
      */
EHomeInterface.getINSTANCE().unbind(item.getDevice().getDevId());
```

5.4 Remove offline device

The device can be removed if it is offline. Usually, this function is called when the device is offline and out of control.

[Sample Code]

```
/**
  *
  * @param tag
  * @param id                    id of device
  * @param baseCallback
  */
EHomeInterface.getINSTANCE().removeDevice(mContext, item.getDevice().getId(),
                              new BaseCallback() {
                                   @Override
                                   public     void     onSuccess(Response<BaseResponse>
response) {
                                        if (response.body().isSuccess()) {
```

```
EHome.getInstance().removeDevice(item.getDevice().getDevId());
                                                    } else {
                                                            Toast.makeText(mContext,       "delete       fail.",
Toast.LENGTH_SHORT).show();
                                                    }
                                            }

                                            @Override
                                            public void onError(Response<BaseResponse> response)
{
                                                    super.onError(response);
                                                    Toast.makeText(mContext,          "delete          fail.",
Toast.LENGTH_SHORT).show();
                                            }
                                    });
```

Data model

DeviceVo

```
private Device device;
private List<FunctionPoint> functionList;
private HashMap<String, Boolean> functionValuesMap;    //Code.FUNCTION_MAP_KEY
private String homeName;
private int homeId;
private String roomName;
private boolean host;
private boolean hasCountDown;
```

Device

```
private int id;
private String name;
private int sellerId;
private int productId;
private Product product;
private long createTime;
private long updateTime;
private String uuid;
private String hid;
private String devId;    // about device control
private boolean actived;
private String authKey;
private String secKey;
private String localKey;
private String version;
private double lat;
private double lng;
private boolean deleted;
private String token;
private String status;//Code.DEVICE_STATUS_NORMAL, Code.DEVICE_STATUS_OFFLINE,
Code.DEVICE_STATUS_BUG
private long firstActiveTime;
private boolean isVirtual;
private boolean state;
private long rowId;
```

---

6. Sharing Devices

6.1 Share your device by email

The device can be shared to your friend by his/her email (note: such email has been registered as a user).

[Sample Code]

```
/**
        * share device with others by input email address
        * @param tag                  context
        * @param masterId          device owner's user id
        * @param userAccount     email of user who share the device with the owner
        * @param deviceId          id of the device to be shared
        * @param baseCallback
        */
EHomeInterface.getINSTANCE().addShare(mContext, masterId, userAccount, deviceId, new
BaseCallback() {
                @Override
                public void onSuccess(Response<BaseResponse> response) {
                            if(response.body().isSuccess()){
                                else {
                                }
          }
                @Override
                  public void onError(Response<BaseResponse> response) {
                                  super.onError(response);
                                  }
                  });
```

6.2 Query shared devices

[Sample Code]

```
/**
  * devices sharing from others
* After getting shared devices list, “saveSharedDevices” method must be called.
* @param mContext
  * @param devicesCallback
  */
EHomeInterface.getINSTANCE().querySharedDevices(mContext, new DevicesCallback() {
        @Override
        public void onSuccess(Response<BaseListResponse<DeviceVo>> response) {
                if (response.body().isSuccess()) {
                  EHomeInterface.getINSTANCE().saveSharedDevices(response.body().getList());
                }
        }
        @Override
        public void onError(Response<BaseListResponse<DeviceVo>> response) {
                super.onError(response);
        }
});
```

6.3 Save shared device

[Sample Code]

```
/**
  * call this method after querySharedDevices() success
  * @param list the list returned by querySharedDevices()
  */
EHomeInterface.getINSTANCE().saveSharedDevices(list);
```

6.4 Remove shared device

The device can be removed if you don't need this device. Usually, this function is called when the device is a device shared to you from your friend.

[Sample Code]

```
/**
 *
 * @param mContext
 * @param deviceId                    device’s Id
 * @param baseCallback
 */
EHomeInterface.getINSTANCE().removeSharedDevice(mContext, deviceId,
new BaseCallback() {
        @Override
        public void onSuccess(Response<BaseResponse> response) {
                if (response.body().isSuccess()) {
              EHome.getInstance().removeDevice(devId);    //this method must be called after
delete success.
                }
        }
        @Override
        public void onError(Response<BaseResponse> response) {
                super.onError(response);
        }
});
```

7. Timer

7.1 Add a timer

Set a timer to operate the device on some specific time. Your operation on the device will take effect once the time of the timer arrives.

Important Note: loops: @“0000000”, each bit, 0: off, 1: on, representing from left to right: Sunday Saturday Friday Thursday Wednesday Tuesday Monday.

[Sample Code]

```
/**
  *
  * @param mContext
  * @param deviceId
  * @param timerType //A seventh-bit binary string. From the lowest bit to highest bit,
indicating Monday, Tuesday, Wednesday, Thursday, Friday, Saturday, Sunday respectively.
“1” means this timer will execute, and “0” means not. For example, “1100000” means timer
will execute on Sunday and Saturday, “0000000” means timer will execute only once without
repeat.
    * @param hour    //from “00” to “23”
  * @param min     //from “00” to “59”
  * @param dps     //execution state of device
  * @param baseCallback
  */
EHomeInterface.getINSTANCE().addTimer(mContext, deviceId, timetype, finishHour,
finishMin, deviceState, new BaseCallback() {
            @Override
            public void onSuccess(Response<BaseResponse> response) {
            }

            @Override
            public void onError(Response<BaseResponse> response) {
                super.onError(response);
            }
        });
```

7.2 Update timer status

You can update/modify the assigned timer.

[Sample Code]

```
/**
  *
  * @param mContext
  * @param clockId
  * @param state    //state of timer
  * @param baseCallback
  */
EHomeInterface.getINSTANCE().updateTimerStatus(mContext, clockId, state,
        new BaseCallback() {
                @Override
                public void onSuccess(Response<BaseResponse> response) {
                }
                @Override
                public void onError(Response<BaseResponse> response) {
                    super.onError(response);
                }
            });
```

7.3 Remove a timer

Delete a specified timer under a specified device.

[Sample Code]

```
/**
 *
 * @param mContext
 * @param clockId
 * @param baseCallback
 */
EHomeInterface.getINSTANCE().removeTimer(mContext, clockId,
            new BaseCallback() {
                @Override
                public void onSuccess(Response<BaseResponse> response) {
                }
                @Override
                public void onError(Response<BaseResponse> response) {
                    super.onError(response);
                }
            });
```

7.4 Obtain all timers of a device

Obtain all timers under a specified device.

[Sample Code]

```
/**
 *
```

```
  * @param mContext
 * @param deviceId    //device's id
 * @param callback
 */
EHomeInterface.getINSTANCE().getTimerList(mContext, deviceId,
            new ClockCallback() {
                  @Override
                  public void onSuccess(Response<BaseListResponse<ClockVo>> response) {
                  }

                  @Override
                  public void onError(Response<BaseListResponse<ClockVo>> response) {
                        super.onError(response);
                  }
            });
```

---

8. Timing Countdown for a specific device

You can create a timing countdown for a specific device.

8.1 Add a timing countdown

Your operation on the device will take effect once timing countdown is finished.

@param state: the status of the device is to be when countdown is finished

@param duration : the duration of timing countdown. The unit is second,

such as 10seconds; if 10 minutes, the value is 600.

[Sample Code]

```
/**
 *
 * @param devId         device's devId
 * @param state         desired state in the future
 * @param duration
 */
EHomeInterface.getINSTANCE().addTimerClockWithDeviceModel(devId, state, duration);
```

8.2 Obtain a timing countdown

Get a timing countdown under a specific device, and then, you can show its value in the APP.

[Sample Code]

```
/**
 *
 * @param tag
 * @param deviceId
 * @param clockCallback
 */
EHomeInterface.getINSTANCE().getTimerClockWithDeviceModel(mContext,
deviceVo.getDevice().getId(), new ClockCallback() {
            @Override
            public void onSuccess(Response<BaseListResponse<ClockVo>> response) {

            }
            @Override
            public void onError(Response<BaseListResponse<ClockVo>> response) {
                super.onError(response);
            }
        });
```

8.3 Update a timing countdown

Once you update a timing countdown, it will become a new one.

[Sample Code]

```
/**
 *
 * @param devId
 * @param state
 * @param duration
 */
EHomeInterface.getINSTANCE().updateTimerClockWithDeviceModel(devId, state, duration);
```

8.4 Remove a timing countdown

[Sample Code]

```
/**
 *
 * @param devId
 */
EHomeInterface.getINSTANCE().removeTimerClockWithDeviceModel(devId);
```

9. Necessary Event

All event meanings can be found in the corresponding class file. Please check the usage of Eventbus at first glance. For all the usage of these EventBus, associated Event should be handled properly.

Turn on event

```
@Subscribe(threadMode = ThreadMode.MAIN, priority = 1, sticky = true)
public void onEventMainThread(MqttReceiveOnEvent event) {}
```

Turn off event

```
@Subscribe(threadMode = ThreadMode.MAIN, priority = 1, sticky = true)
public void onEventMainThread(MqttReceiveOffEvent event) {}
```

Unbind event

```
@Subscribe(threadMode = ThreadMode.MAIN, priority = 1, sticky = true)
```

```
public void onEventMainThread(MqttReceiveUnbindEvent event) {}
```

Offline event

```
@Subscribe(threadMode = ThreadMode.MAIN, priority = 1, sticky = true)
public void onEventMainThread(MqttReceiveStatusEvent event) {}
```

Bind success event

```
@Subscribe(threadMode = ThreadMode.MAIN, priority = 1, sticky = true)
public void onEventMainThread(MqttBindSuccessEvent event) {}
```

Already bind event

```
@Subscribe(threadMode = ThreadMode.MAIN, priority = 1, sticky = true)
public void onEventMainThread(MqttAlreadyBindEvent event) {}
```

Shared device: turn on event

```
@Subscribe(threadMode = ThreadMode.MAIN, priority = 1, sticky = true)
public void onEventMainThread(MqttReceiveSharedOnEvent event) {}
```

Shared device: turn off event

```
@Subscribe(threadMode = ThreadMode.MAIN, priority = 1, sticky = true)
public void onEventMainThread(MqttReceiveSharedOffEvent event) {}
```

Shared device: offline event

```
@Subscribe(threadMode = ThreadMode.MAIN, priority = 1, sticky = true)
public void onEventMainThread(MqttReceiveSharedStatusEvent event) {}
```

Set countdown success event

```
@Subscribe(threadMode = ThreadMode.MAIN, priority = 1, sticky = true)
public void onEventMainThread(MqttSetCdSuccessEvent event) {}
```

Cancel countdown success event

```
@Subscribe(threadMode = ThreadMode.MAIN, priority = 1, sticky = true)
public void onEventMainThread(MqttCancelCdSuccessEvent event) {}
```

Add/open timer success event

```
@Subscribe(threadMode = ThreadMode.MAIN, priority = 1, sticky = true)
public void onEventMainThread(MqttSetTimerSuccessEvent event) {}
```

```
Delete/close timer success event
```

```
@Subscribe(threadMode = ThreadMode.MAIN, priority = 1, sticky = true)
public void onEventMainThread(MqttCancelTimerSuccessEvent event) {}
```

Welcome to contact us:

- Android SDK Contributors: Zheng Li, Pan Zhao, Shiwen Ning
- Email : zhouwei20150901@icloud.com, whatie@qq.com

LICENSE