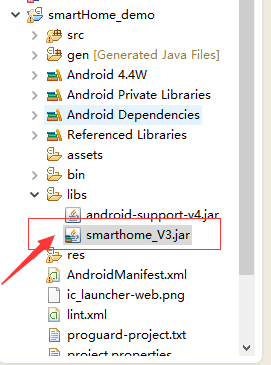
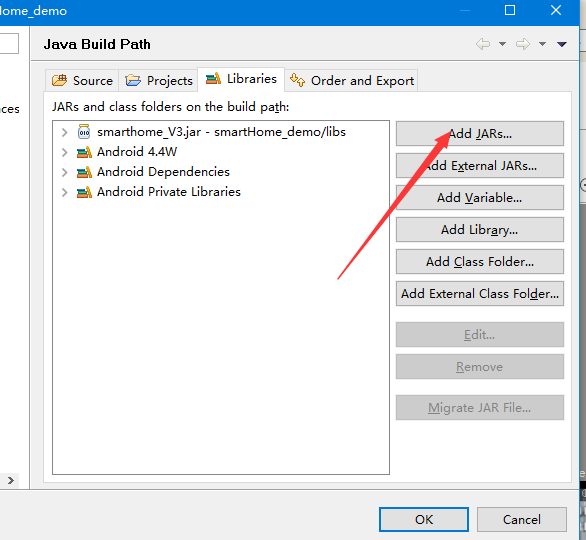
# 国赛库库文档说明

# 1 导入开发资源

1,将提供smarthome\_V3.jar资源包复制到你的工程下的libs目录下，如下图。



2, 在Eclipse设置引入jar文件。如下图。



# 2.智能家居(ControlUtils)

## 2.1 socket长连接

|  |
| --- |
| 方法： SocketClient.getInstance().creatConnect(); ; |
| 简介：socket进行长连接 |
| 参数： 无 |
| 返回值：String status 类型 Success: 成功 Failure： 失败； |
| 调用示例：   ControlUtils.setUser(String UserName, String Password, String ip)   SocketClient.getInstance().creatConnect();  SocketClient.getInstance().login( new LoginCallback() {  @Override  public void onEvent(final String status) {  runOnUiThread( new Runnable() {  @Override  public void run() {  if (status.equals( "Success" )) {  startActivity( new Intent( LoginActivity.this, MainActivity.class ) );  finish();  } else {  Toast.makeText( LoginActivity.this, "失败！", Toast.LENGTH\_SHORT ).show();  }  }  } );  }  } ); |

## 2.2 单控发送数据

|  |
| --- |
| 方法： control(String SensorType, String Channel, String Command) |
| 简介：单控发送数据 |
| 参数：  Type 字段说明 ：Control 控制  SensorType 字段说明：   |  |  |  |  | | --- | --- | --- | --- | | 传感器类型 | SensorType | Channel | Command | | 温度 | Temperature |  |  | | 湿度 | Humidity |  |  | | 光照 | Illumination |  |  | | 烟雾 | Smoke |  |  | | 燃气 | Gas |  |  | | PM2.5 | PM25 |  |  | | 二氧化碳 | CO2 |  |  | | 气压 | AirPressure |  |  | | 人体红外 | StateHumanInfrared |  |  | | 风扇 | Fan | CHANNEL\_ALL | OPEN 开 CLOSE 关 | | 射灯 | Lamp | CHANNEL\_ALL | OPEN 开 CLOSE 关 | | 报警灯 | WarningLight | CHANNEL\_ALL | OPEN 开 CLOSE 关 | | 窗帘 | Curtain | CHANNEL\_1开CHANNEL\_2 停  CHANNEL\_3 关 | OPEN | | 红外发射1 | INFRARED\_1\_SERVE | CHANNEL\_1通道 CHANNEL\_2通道  CHANNEL\_3通道 | OPEN 开 CLOSE 关 | | 门禁控制 | RFID\_Open\_Door | CHANNEL\_ALL | OPEN | |
| 返回值：boolean true: 成功false: 失败； |
| 调用示例：  ControlUtils.control( ConstantUtil.Curtain, ConstantUtil.CHANNEL\_1, ConstantUtil.OPEN ); |

## 2.3传感器数据主动上传

|  |
| --- |
| 方法： ControlUtils.getData(); |
| 简介：传感器数据上传 |
| 参数： DeviceBean  Value：getTemperature() //温度  Value：getHumidity() //湿度  Value：getIllumination() //光照  Value：getSmoke() //烟雾  Value：getGas() //燃气  Value：getPM25() //PM2.5  Value：getCo2() //二氧化碳  Value：getStateHumanInfrared() //人体红外 |
| 返回值： onResult(final DeviceBean.Device bean) |
| 调用示例：  SocketClient.*getInstance*().getData( **new** DataCallback<DeviceBean>() {  @Override  **public** **void** onResult(**final** DeviceBean bean) {  runOnUiThread( **new** Runnable() {  @Override  **public** **void** run() {  **if** (!TextUtils.*isEmpty*(bean.*getTemperature*() )){  mTempTv.setText( bean.*getTemperature*() );  }  **if** (!TextUtils.*isEmpty*( bean.*getHumidity*() )){  mHumidityTv.setText( bean.*getHumidity*() );  }  **if** (!TextUtils.*isEmpty*( bean.*getGas*() )){  mGasTv.setText( bean.*getGas*() );  }  **if** (!TextUtils.*isEmpty*( bean.*getIllumination*() )){  mIlluminationTv.setText( bean.*getIllumination*() );  }  **if** (!TextUtils.*isEmpty*( bean.*getPM25*() )){  mPm25Tv.setText( bean.*getPM25*() );  }  **if** (!TextUtils.*isEmpty*( bean.*getSmoke*() )){  mSmokeTv.setText( bean.*getSmoke*() );  }  **if** (!TextUtils.*isEmpty*( bean.*getCo2*() )){  mCo2Tv.setText( bean.*getCo2*() );  }  **if** (!TextUtils.*isEmpty*( bean.*getAirPressure*() )){  mPressureTv.setText( bean.*getAirPressure*() );  }  **if** (!TextUtils.*isEmpty*( bean.*getStateHumanInfrared*() ) && bean.*getStateHumanInfrared*().equals( ConstantUtil.*CLOSE* )){  mStateHumanInfraredTv.setText( "无人" );  }**else**{  mStateHumanInfraredTv.setText( "有人" );  }  **if** (!TextUtils.*isEmpty*( bean.*getLamp*() ) && bean.*getLamp*().equals( ConstantUtil.*CLOSE* )) {  mLampCb.setChecked( **false** );  mLampCb.setBackgroundResource( R.drawable.*lamp\_unpressed* );  } **else** {  mLampCb.setBackgroundResource( R.drawable.*lamp\_pressed* );  mLampCb.setChecked( **true** );  }  **if** (!TextUtils.*isEmpty*( bean.*getFan*() ) && bean.*getFan*().equals( ConstantUtil.*CLOSE* )) {  mWindSpeedCb.setChecked( **false** );  mWindSpeedCb.setBackgroundResource( R.drawable.*wind\_speed\_unpressed* );  } **else** {  mWindSpeedCb.setBackgroundResource( R.drawable.*wind\_speed\_pressed* );  mWindSpeedCb.setChecked( **true** );  }  **if** (!TextUtils.*isEmpty*( bean.*getCurtain*() ) && bean.*getCurtain*().equals( ConstantUtil.*CHANNEL\_1* )) {  mCurtainOpenImg.setBackgroundResource( R.drawable.*curtain\_open\_pressed* );  mCurtainStopImg.setBackgroundResource( R.drawable.*curtain\_stop\_unpressed* );  mCurtainCloseImg.setBackgroundResource( R.drawable.*curtain\_close\_unpressed* );  } **else** **if** (!TextUtils.*isEmpty*( bean.*getCurtain*() ) && bean.*getCurtain*().equals( ConstantUtil.*CHANNEL\_2* )) {  mCurtainOpenImg.setBackgroundResource( R.drawable.*curtain\_open\_unpressed* );  mCurtainStopImg.setBackgroundResource( R.drawable.*curtain\_stop\_pressed* );  mCurtainCloseImg.setBackgroundResource( R.drawable.*curtain\_close\_unpressed* );  } **else** **if** (!TextUtils.*isEmpty*( bean.*getCurtain*() ) && bean.*getCurtain*().equals( ConstantUtil.*CHANNEL\_3* )) {  mCurtainOpenImg.setBackgroundResource( R.drawable.*curtain\_open\_unpressed* );  mCurtainStopImg.setBackgroundResource( R.drawable.*curtain\_stop\_unpressed* );  mCurtainCloseImg.setBackgroundResource( R.drawable.*curtain\_close\_pressed* );  }  **if** (!TextUtils.*isEmpty*( bean.*getWarningLight*() ) && bean.*getWarningLight*().equals( ConstantUtil.*CLOSE* )) {  mAlarmCb.setChecked( **false** );  mAlarmCb.setBackgroundResource( R.drawable.*alarm\_unpressed* );  } **else** {  mAlarmCb.setChecked( **true** );  mAlarmCb.setBackgroundResource( R.drawable.*alarm\_pressed* );  }  }  } );  }  } ); |

## 2.4情景模式

* 光照模式：光照度大于设定值100就打开灯光，反之关闭；

示例：

if (!TextUtils.isEmpty( DeviceBean.getIllumination() ) && Double.parseDouble( DeviceBean.getIllumination() ) >= 100) {

if (!TextUtils.isEmpty( DeviceBean.getLamp() ) && DeviceBean.getLamp().equals( ConstantUtil.CLOSE ))

ControlUtils.control( ConstantUtil.Lamp, ConstantUtil.CHANNEL\_ALL, ConstantUtil.OPEN );

} else {//灯光开，窗户关

if (!TextUtils.isEmpty( DeviceBean.getLamp() ) && !DeviceBean.getLamp().equals( ConstantUtil.CLOSE ))

ControlUtils.control( ConstantUtil.Lamp, ConstantUtil.CHANNEL\_ALL, ConstantUtil.CLOSE );

}

* 温度模式：温度少于10度就打开空调，反之关闭；

示例：

if (!TextUtils.isEmpty( DeviceBean.getTemperature() ) && Double.parseDouble( DeviceBean.getTemperature() ) <= 10) {

ControlUtils.control( ConstantUtil.INFRARED\_1\_SERVE, ConstantUtil.CHANNEL\_2, ConstantUtil.OPEN );

} else {

ControlUtils.control( ConstantUtil.INFRARED\_1\_SERVE, ConstantUtil.CHANNEL\_2, ConstantUtil.CLOSE );

}

* 安防模式：人体红外检测到有人就开启报警器，反正关闭；

示例：  
 if (!TextUtils.isEmpty( DeviceBean.getStateHumanInfrared() ) && DeviceBean.getStateHumanInfrared().equals( ConstantUtil.CLOSE )) {//人体红外无人时关闭

if (!TextUtils.isEmpty( DeviceBean.getWarningLight() ) && !DeviceBean.getWarningLight().equals( ConstantUtil.CLOSE ))

ControlUtils.control( ConstantUtil.WarningLight, ConstantUtil.CHANNEL\_ALL, ConstantUtil.CLOSE );

} else if (!TextUtils.isEmpty( DeviceBean.getStateHumanInfrared() ) && !DeviceBean.getStateHumanInfrared().equals( ConstantUtil.CLOSE )) {//有人时打开报警灯

if (!TextUtils.isEmpty( DeviceBean.getWarningLight() ) && DeviceBean.getWarningLight().equals( ConstantUtil.CLOSE ))

ControlUtils.control( ConstantUtil.WarningLight, ConstantUtil.CHANNEL\_ALL, ConstantUtil.OPEN );

}

* 离家模式：灯光关闭，窗帘关闭，红外设备关闭；

示例：

private int i = 0;

//离家模式

Runnable runnable = new Runnable() {//离家模式

@Override

public void run() {

// handler自带方法实现定时器

try {

mHandler.postDelayed( this, 5000 );

i++;

switch (i) {//

case 1://灯光

if (!TextUtils.isEmpty( DeviceBean.getLamp() ) && !DeviceBean.getLamp().equals( ConstantUtil.CLOSE ))

ControlUtils.control( ConstantUtil.Lamp, ConstantUtil.CHANNEL\_1, ConstantUtil.CLOSE );

break;

case 2: //窗帘

if (!TextUtils.isEmpty( DeviceBean.getCurtain() ) && DeviceBean.getCurtain().equals( ConstantUtil.CHANNEL\_1 ))

ControlUtils.control( ConstantUtil.Curtain, ConstantUtil.CHANNEL\_3, ConstantUtil.OPEN );//关

break;

case 3: //通道1

ControlUtils.control( ConstantUtil.INFRARED\_1\_SERVE, ConstantUtil.CHANNEL\_1, ConstantUtil.OPEN );

break;

case 4://通道2

ControlUtils.control( ConstantUtil.INFRARED\_1\_SERVE, ConstantUtil.CHANNEL\_2, ConstantUtil.OPEN );

break;

case 5://通道3

ControlUtils.control( ConstantUtil.INFRARED\_2\_SERVE, ConstantUtil.CHANNEL\_1, ConstantUtil.OPEN );

break;

case 6://通道4

ControlUtils.control( ConstantUtil.INFRARED\_2\_SERVE, ConstantUtil.CHANNEL\_2, ConstantUtil.OPEN );

break;

case 7://风扇

ControlUtils.control( ConstantUtil.Fan, ConstantUtil.CHANNEL\_1, ConstantUtil.CLOSE );

mHandler.removeCallbacks( runnable );

break;

}

} catch (Exception e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

};

* 睡眠模式：灯光关闭，窗帘关闭，红外设备关闭；

private int j = 0;

//睡眠模式

Runnable runnable1 = new Runnable() {//睡眠模式

@Override

public void run() {

// handler自带方法实现定时器

try {

mHandler.postDelayed( this, 2000 );

j++;

switch (j) {

case 1://灯光

if (!TextUtils.isEmpty( DeviceBean.getLamp() ) && !DeviceBean.getLamp().equals( ConstantUtil.CLOSE ))

ControlUtils.control( ConstantUtil.Lamp, ConstantUtil.CHANNEL\_ALL, ConstantUtil.CLOSE );

break;

case 2: //窗帘

if (!TextUtils.isEmpty( DeviceBean.getCurtain() ) && !DeviceBean.getCurtain().equals( ConstantUtil. CHANNEL\_1))

ControlUtils.control( ConstantUtil.Curtain, ConstantUtil.CHANNEL\_4, ConstantUtil.OPEN );//关

break;

case 3: //通道1

ControlUtils.control( ConstantUtil.INFRARED\_1\_SERVE, ConstantUtil.CHANNEL\_1, ConstantUtil.OPEN );

break;

case 4://通道2

ControlUtils.control( ConstantUtil.INFRARED\_1\_SERVE, ConstantUtil.CHANNEL\_2, ConstantUtil.OPEN );

break;

case 5://通道3

ControlUtils.control( ConstantUtil.INFRARED\_2\_SERVE, ConstantUtil.CHANNEL\_1, ConstantUtil.OPEN );

break;

case 6://通道4

ControlUtils.control( ConstantUtil.INFRARED\_2\_SERVE, ConstantUtil.CHANNEL\_2, ConstantUtil.OPEN );

break;

case 7://风扇

ControlUtils.control( ConstantUtil.Fan, ConstantUtil.CHANNEL\_1, ConstantUtil.CLOSE );

mHandler.removeCallbacks( runnable1 );

break;

}

} catch (Exception e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

};

起床模式：(窗帘开启)

示例：

if (!TextUtils.isEmpty( DeviceBean.getCurtain() ) && !DeviceBean.getCurtain().equals( ConstantUtil.CHANNEL\_1 )) {

ControlUtils.control( ConstantUtil.Curtain, ConstantUtil.CHANNEL\_1, ConstantUtil.OPEN );//开

}