# 微兔-大华电子秤开发文档 By 周全

1.连接电子秤两种类型：

首先：导入电子秤electriscale.jar包

1.1:初始化第一次连接

ElectricScaleManager.*getInstance*().findElectricScale(this);

在需要扫描的时候开启扫描附近的电子秤设备进行连接，默认为大华Tm系列，网关是以太网。

1.2：已经连接过的再次重新连接

ElectricScaleManager.*getInstance*().findElectricScaleReconnect(this);

开启这个方法来判断当前电子秤是否和设备连接成功，操作任何电子秤的数据操作，在当前页面必须每次调用一次来操作电子秤，确保对电子秤的操作能成功。

2.监听当前电子秤的操作状态：

ElectricScaleManager.*getInstance*().findElectricScaleReconnect(this);  
ElectricScaleManager.*getInstance*().setOnBaseElectronicScaleOperationListen(new ElectricScaleManager.OnBaseElectronicScaleOperationListen(){  
 @Override  
 public void electronicScaleConnectSuccess() {  
 Logger.*d*("ElectricScaleManager 当前设备连接成功");  
 handler.sendEmptyMessage(FLAG\_CONNECT\_SUCCESS);  
 }  
 @Override  
 public void electronicScaleConnectFailed() {  
 Logger.*d*("ElectricScaleManager 当前设备连接失败");  
 handler.sendEmptyMessage(FLAG\_RECONNECT\_FAILED);  
 }  
 @Override  
 public void electronicScaleReconnectSuccess() {  
 Logger.*d*("ElectricScaleManager 当前设备重新连接成功");  
 handler.sendEmptyMessage(FLAG\_RECONNECT\_SUCCESS);  
 }  
  
 @Override  
 public void electronicScaleReconnectFailed() {  
 Logger.*d*("ElectricScaleManager 当前设备重新连接失败");  
 handler.sendEmptyMessage(FLAG\_RECONNECT\_FAILED);  
 }  
 @Override  
 public void electronicScaleOperationSuccessful(String msg) {  
 Logger.*d*("ElectricScaleManager 当前设备发送数据成功");  
 }  
 @Override  
 public void electronicScaleOperationFailed() {  
 Logger.*d*("ElectricScaleManager 当前设备发送数据失败");  
 }  
 @Override  
 public void electronicScaleMessageSuccessful() {  
 Logger.*d*("ElectricScaleManager 当前设备正在发送数据中");  
 }  
});

根据当前状态来监听电子秤是否操作成功，连接成功，在这些状态方法里面来处理当前的操作

在每个页面都应该执行这个方法来监听当前电子秤的状态，方便处理业务逻辑。

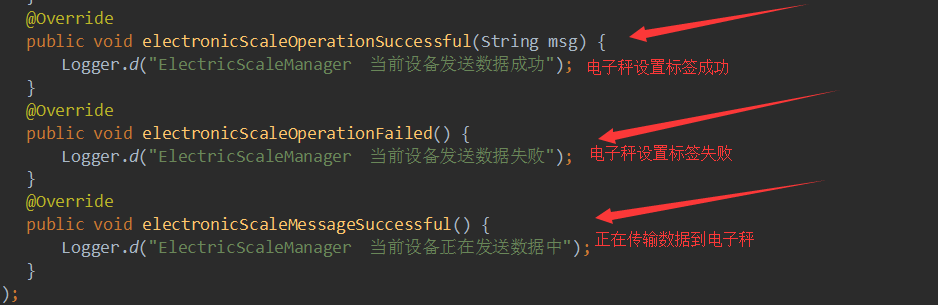
1. 设置电子秤标签尺寸：

ElectricScaleManager.*getInstance*().initElectricScaleLabelSize(this);

调用上面方法初始化电子秤标签

当前设备标签尺寸只支持40mm\*30mm。

注意同时也要调用监听方法来判断是否操作成功



1. 传秤，传递商品数据到电子秤上面：

ElectricScaleDataParam这个类用来传输数据

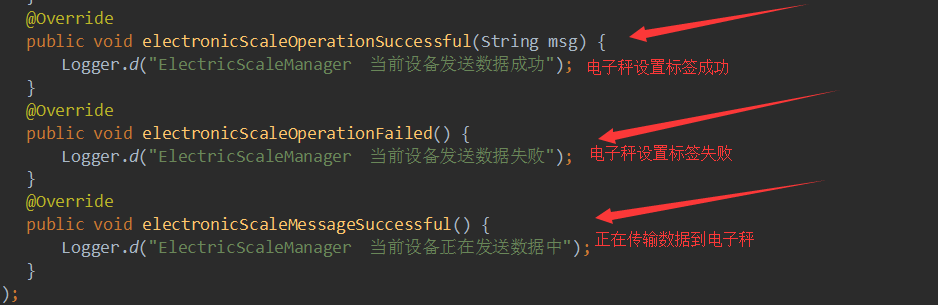
private String barcode; // 商品条形码  
private BigDecimal sellingPrice; // 销售价格  
private String name; // 商品名称  
private String plu;//plu编码

将所有需要传秤的商品数据添加到集合当中去，然后添加到方法里面，可以循环添加多条

ElectricScaleDataParam electricScaleDataParam=new ElectricScaleDataParam();  
electricScaleDataParam.setBarcode("0005");   
electricScaleDataParam.setName("苹果");  
electricScaleDataParam.setSellingPrice(*toBigDecimal*("19.99"));  
electricScaleDataParam.setPlu("0005");  
electricScaleDataParamList.add(electricScaleDataParam);

然后将electricScaleDataParamList 集合里面的数据添加调用下面方法

ElectricScaleManager.*getInstance*().electricScaleTransferData(electricScaleDataParamList);



5：注意事项

5.1 ：如果已经初始化电子秤连接成功过，之后在每个页面发送数据指令操作的时候必须先提前调用一次下面方法，确保当前设备和电子秤是连接成功状态，才能确保设置电子秤操作能够成功，在onCreate里面调用。

ElectricScaleManager.*getInstance*().findElectricScaleReconnect(this);

5.2：建议在每个页面都要执行下面方法

ElectricScaleManager.*getInstance*().findElectricScaleReconnect(this); //连接电子秤  
ElectricScaleManager.*getInstance*().setOnBaseElectronicScaleOperationListen(new ElectricScaleManager.OnBaseElectronicScaleOperationListen(){  
 @Override  
 public void electronicScaleConnectSuccess() {  
 Logger.*d*("ElectricScaleManager 当前设备连接成功");  
 handler.sendEmptyMessage(FLAG\_CONNECT\_SUCCESS);  
 }  
 @Override  
 public void electronicScaleConnectFailed() {  
 Logger.*d*("ElectricScaleManager 当前设备连接失败");  
 handler.sendEmptyMessage(FLAG\_RECONNECT\_FAILED);  
 }  
 @Override  
 public void electronicScaleReconnectSuccess() {  
 Logger.*d*("ElectricScaleManager 当前设备重新连接成功");  
 handler.sendEmptyMessage(FLAG\_RECONNECT\_SUCCESS);  
 }  
  
 @Override  
 public void electronicScaleReconnectFailed() {  
 Logger.*d*("ElectricScaleManager 当前设备重新连接失败");  
 handler.sendEmptyMessage(FLAG\_RECONNECT\_FAILED);  
 }  
 @Override  
 public void electronicScaleOperationSuccessful(String msg) {  
 Logger.*d*("ElectricScaleManager 当前设备发送数据成功");  
 }  
 @Override  
 public void electronicScaleOperationFailed() {  
 Logger.*d*("ElectricScaleManager 当前设备发送数据失败");  
 }  
 @Override  
 public void electronicScaleMessageSuccessful() {  
 Logger.*d*("ElectricScaleManager 当前设备正在发送数据中");  
 }  
});

5.3：在应用页面的onDestroy生命周期执行关掉connect连接，防止内存泄露

@Override  
protected void onDestroy() {  
 super.onDestroy();  
 ElectricScaleManager.*getInstance*().closeTcpConn();  
}

5.4：string字符串转换 BigDecimal 的工具类

public static BigDecimal toBigDecimal(String param){  
 if(*isEmpty*(param)){  
 return BigDecimal.*ZERO*;  
 }  
 try {  
 return new BigDecimal(param);  
 } catch (Exception e) {  
 e.printStackTrace();  
 return BigDecimal.*ZERO*;  
 }  
 }  
 public static boolean isEmpty(String str) {  
 if (str == null || "".equals(str) || "null".equalsIgnoreCase(str)) {  
 return true;  
 }  
 for (int i = 0; i < str.length(); i++) {  
 char c = str.charAt(i);  
 if (c != ' ' && c != '\t' && c != '\r' && c != '\n') {  
 return false;  
 }  
 }  
 return true;  
}