第 4 章　项目实现(Implement)——编码和测试

- **本章目标：**

  - 应用程序启动后 Splash 效果的实现方式
  - 应用给定的图片资源，选择合适的布局方案和控件，实现界面的布局
  - 应用后台服务端和客户端之间的数据传输方式
  - 应用关系数据库的增删改查等常用操作
  - 应用日志文件在项目中的作用和用法
  - 应用多线程和 Handler 的常见用法
  - 应用 Gesture 技术在触屏操作上的用法
  - 应用绘制图形相关的第三方库的用法
  - 应用常见模块(如登录注册模块和系统设置模块)的实现方式
  - 应用定时采集各种传感器数据的实现方式

- **课时分配：**

  - 90 课时

子任务 1　使用 JSON 封装数据实现客户端与服务端的通信

- **子任务目标：**

  - 应用 JSON 协议对数据进行封装，实现客户端与服务端的通信

- **课时分配：**

  - 8 课时

4.1.1　任务构思(Conceive)

在智能农业实战项目中，客户端和服务端需要通信。通信方式可以采取 HTTP 或 SOCKET 方式，但不管采用哪种通信方式，都需要通信协议，在本任务中使用 JSON 和

Soap 两种通信协议。简单地说,JSON 就是 Javascript 中的对象和数组,通过这两种结构可以表示各种复杂的结构。

4.1.2 任务设计(Design)

在 Android 系统中,把数据封装成 JSON 格式可以通过如下 3 步实现:

(1)创建 JSONObject 对象;

(2)调用 put(Object key, Object value)方法给变量赋值;

(3)调用 toString()方法把 JSONObject 对象转换为 JSON 格式的字符串。

在 Android 系统中,对 JSON 数据的解析可以通过如下 3 步实现:

(1)创建 JSONObject 对象;

(2)调用 has(String key)方法,判断 JSONObject 是否含有变量 key(可省略);

(3)调用 getString(String key)方法,获取变量 key 的值。

4.1.3 任务实现(Implement)

智能农业实战项目的该子任务包含的角色为开发工程师。角色扮演的过程如表 4-1-1所示。

表 4-1-1　　角色扮演过程

步骤	角色	工作	输出
1	开发工程师	编码实现子任务的业务功能	实现子任务业务功能的 Android 源代码
2	开发工程师	两两交叉代码评审	代码评审纪要
3	开发工程师	根据代码评审纪要重构代码	重构后的 Android 源代码

4.1.4 任务运作(Operate)

所有团队成员召开计划会议,并将会议纪要记录在表 4-1-2 中。

表 4-1-2　　会议纪要记录表

5W2H	结论
WHAT	
HOW	
WHY	
WHEN	
WHERE	
WHO	
HOW MUCH	

在计划会议结束之前,所有团队成员把子任务进一步细分为更小的任务,把这些任

务写在便签上，并把便签贴在看板上的“ToDo”列。

所有团队成员每天早上召开 15 分钟的站立会议，每人汇报三个问题，并更新看板上的“ToDo”、“Doing”和“Done”三列。

所有团队成员相互协作，编码实现子任务的业务功能，详细的示例代码请见 Android 项目：JSON，参考代码如下：

(1)创建布局文件：activity_main. xml

```
<RelativeLayout xmlns:android= "http://schemas.android.com/apk/res/android"
    xmlns:tools= "http://schemas.android.com/tools"
    android:layout_width= "match_parent"
    android:layout_height= "match_parent"
    android:background= "#8B6969"
    tools:context= ".MainActivity">

    <TextView
        android:id= "@+id/textView1"
        android:layout_width= "wrap_content"
        android:layout_height= "wrap_content"
        android:layout_alignParentLeft= "true"
        android:layout_alignParentTop= "true"
        android:layout_marginLeft= "17dp"
        android:layout_marginTop= "27dp"
        android:textSize= "18dp"
        android:textColor= "#00CD00"
        android:text= "json--->java"/>

    <TextView
        android:id= "@+id/textView2"
        android:layout_width= "wrap_content"
        android:layout_height= "wrap_content"
        android:layout_alignLeft= "@+id/textView1"
        android:layout_below= "@+id/textView1"
        android:layout_marginLeft= "38dp"
        android:layout_marginTop= "37dp"
        android:textSize= "20dp"
        android:text= "TextView"/>

    <TextView
        android:id= "@+id/textView5"
        android:layout_width= "wrap_content"
        android:layout_height= "wrap_content"
```

```
    android:layout_alignLeft="@+id/textView2"
    android:layout_below="@+id/textView2"
    android:layout_marginTop="38dp"
    android:textSize="20dp"
    android:text="TextView"/>

<TextView
    android:id="@+id/textView6"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignLeft="@+id/textView5"
    android:layout_below="@+id/textView5"
    android:layout_marginTop="35dp"
    android:textSize="20dp"
    android:text="TextView"/>

<TextView
    android:id="@+id/textView7"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignLeft="@+id/textView6"
    android:layout_below="@+id/textView6"
    android:layout_marginTop="33dp"
    android:textSize="20dp"
    android:text="TextView"/>

<TextView
    android:id="@+id/textView4"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignLeft="@+id/textView7"
    android:layout_below="@+id/textView3"
    android:layout_marginTop="36dp"
    android:textSize="20dp"
    android:text="TextView"/>

<TextView
    android:id="@+id/textView3"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignLeft="@+id/textView1"
```

```
        android:layout_below="@+id/textView7"
        android:layout_marginTop="82dp"
        android:textSize="18dp"
        android:textColor="#00CD00"
        android:text="java--->json"/>

    <TextView
        android:id="@+id/textView8"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignLeft="@+id/textView7"
        android:layout_below="@+id/textView7"
        android:layout_marginTop="30dp"
        android:textSize="20dp"
        android:text="TextView"/>

</RelativeLayout>
```

(2)创建 Activity 类:MainActivity.java

```
public class MainActivity extends Activity {
    //二氧化碳,光照强度,温度,PM2.5,体感,json 数据显示
    private TextView tco2, tlight, ttemper, tPM, tfeel, tjson;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main );
        tco2=(TextView) this.findViewById(R.id.textView2);
        tlight=(TextView) this.findViewById(R.id.textView5);
        ttemper=(TextView) this.findViewById(R.id.textView6);
        tPM=(TextView) this.findViewById(R.id.textView7);
        tfeel=(TextView) this.findViewById(R.id.textView8);
        tjson=(TextView) this.findViewById(R.id.textView4);
        //将数据显示到界面上
        JavatoJson();
        JsontoJava();
    }

    //把数据封装成 Json 格式
    public void JavatoJson() {
        JSONObject jsonObject=new JSONObject();
        try {
```

```
            jsonObject.put("CO2:", 0.039);
            jsonObject.put("Light", 5000);
            jsonObject.put("Temperature", 24.5d);
            jsonObject.put("PM25", 60);
            jsonObject.put("feel", true);
            tjson.setText(jsonObject.toString());
        }catch (JSONException e) {
            e.printStackTrace();
        }
    }

    //解析 Json 格式的数据
    public void JsontoJava() {
        String jsonStr="{CO2: 0.039,'Light':5000,Temperature:24.5d,PM25:60,feel:true}";

        JSONObjectjsonObj;
        try {
            jsonObj=new JSONObject(jsonStr);
            String CO2=jsonObj.getString("CO2");
            String Light=jsonObj.getString("Light");
            String Temperature=jsonObj.getString("Temperature");
            String PM25=jsonObj.getString("PM25");
            boolean feel=jsonObj.getBoolean("feel");

            tco2.setText("CO2 浓度: " + CO2 + " %");
            tlight.setText("光照强度: " + Light + " cd/m2");
            ttemper.setText("温度:" + Temperature + " ℃ ");
            tPM.setText("PM2.5: " + PM25 + " ug/m3");
            if (feel) {
                tfeel.setText("feel: 倍爽");
            }else {
                tfeel.setText("feel: 不爽");
            }
        }catch (JSONException e) {
            e.printStackTrace();
        }
    }
}
```

(3)在 AndroidManifest.xml 中注册 Activity

```
<? xml version= "1.0"encoding= "utf-8"? >
<manifest xmlns:android= "http://schemas.android.com/apk/res/android"
```

```
package= "com.lenovo.demo"
    android:versionCode= "1"
    android:versionName= "1.0">

    <uses-sdk
        android:minSdkVersion= "8"
        android:targetSdkVersion= "18"/>

    <application
        android:allowBackup= "true"
        android:icon= "@drawable/app"
        android:label= "@string/app_name">
        <activity
            android:name= ".activity.MainActivity"
            android:label= "@string/app_name">
            <intent-filter>
                <actionandroid:name= "android.intent.action.MAIN"/>

                <categoryandroid:name= "android.intent.category.LAUNCHER"/>
            </intent-filter>
        </activity>
    </application>

</manifest>
```

确保以上每一步都编译通过：在 IDE 集成开发环境中，没有出现红叉的错误提示。应用程序的运行效果如图 4-1-1 所示。

图 4-1-1　应用程序的运行效果图

开发工程师之间两两交叉代码评审，并根据评审纪要重构代码。

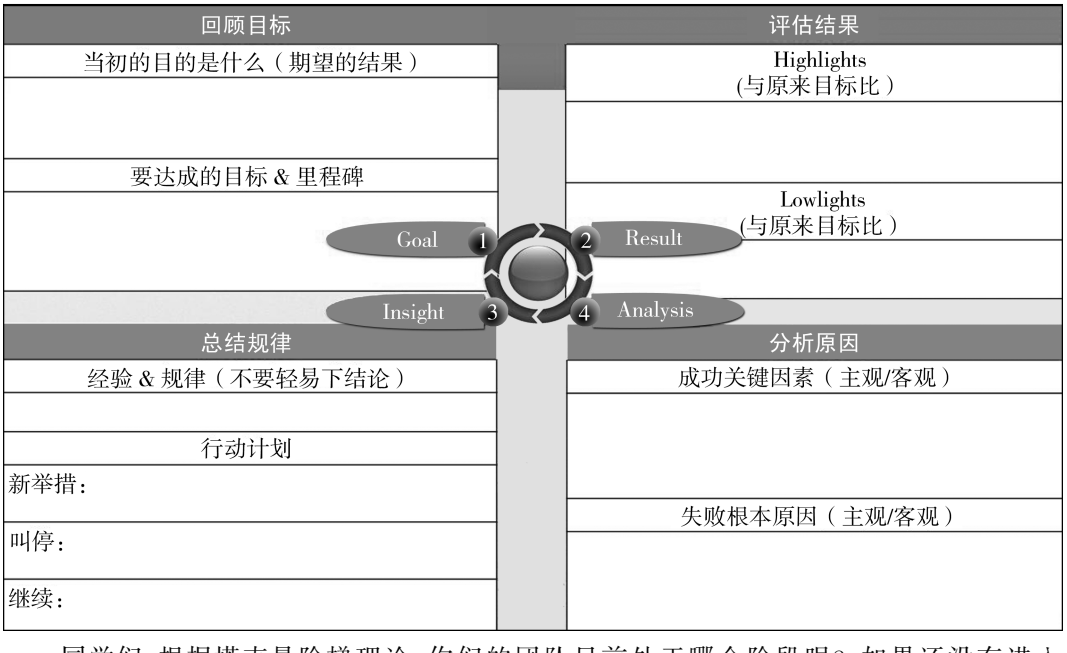
所有团队成员召开评审会，并填写表 4-1-3。

表 4-1-3　　评审建议表

序号	建议描述	是否接纳	原因
1		□接纳　□部分接纳　□未接纳	
2		□接纳　□部分接纳　□未接纳	
3		□接纳　□部分接纳　□未接纳	
4		□接纳　□部分接纳　□未接纳	
5		□接纳　□部分接纳　□未接纳	
6		□接纳　□部分接纳　□未接纳	
7		□接纳　□部分接纳　□未接纳	
8		□接纳　□部分接纳　□未接纳	
9		□接纳　□部分接纳　□未接纳	
10		□接纳　□部分接纳　□未接纳	

所有团队成员召开复盘会，并填写表 4-1-4。

表 4-1-4　　复盘会总结表



同学们，根据塔克曼阶梯理论，你们的团队目前处于哪个阶段呢？如果还没有进入成熟阶段，那就让我们一起努力吧，在接下来的任务中，更加注重相互配合和协作，及时发现并改正团队配合过程中出现的问题，早日让我们的团队进入成熟阶段！

4.1.5　任务扩展(Extend)

课后练习:使用 JSON 封装一个普通对象,实现客户端与服务端的通信。

子任务 2　使用 SOAP 协议实现客户端与服务端的通信

- **子任务目标:**

  ➢ 应用 SOAP 协议,实现客户端和服务端的数据通信

- **课时分配:**

  ➢ 8 课时

4.2.1　任务构思(Conceive)

在智能农业实战项目中,通过 SOAP 协议来实现客户端与服务端的通信:在客户端创建请求对象,通过线程将该消息发送给服务端,同时服务端给客户端发送反馈信息,然后客户端对该信息进行解析和处理。整个过程分为:注册、登录和获取相关的环境数据。

4.2.2　任务设计(Design)

在智能农业实战项目中,SOAP 协议主要运用在以下 3 个模块中:

(1)注册:在服务端注册账号,在拥有账号的基础上,才能进行登录操作。

(2)登录:在客户端进行登录,通过服务端的认证后,才能进行客户端 App 的后续操作。

(3)获取:从服务端获取环境信息并显示在客户端的界面上。

4.2.3　任务实现(Implement)

智能农业实战项目的该子任务包含的角色为开发工程师。角色扮演的过程如表 4-2-1所示。

表 4-2-1　　角色扮演过程

步骤	角色	工作	输出
1	开发工程师	编码实现子任务的业务功能	实现子任务业务功能的 Android 源代码
2	开发工程师	两两交叉代码评审	代码评审纪要
3	开发工程师	根据代码评审纪要重构代码	重构后的 Android 源代码

4.2.4　任务运作(Operate)

所有团队成员召开计划会议,并将会议纪要记录在表 4-2-2 中。

表 4-2-2　　会议纪要记录表

5W2H	结论
WHAT	
HOW	
WHY	
WHEN	
WHERE	
WHO	
HOW MUCH	

在计划会议结束之前，所有团队成员把子任务进一步细分为更小的任务，把这些任务写在便签上，并把便签贴在看板上的“ToDo”列。

所有团队成员每天早上召开 15 分钟的站立会议，每人汇报三个问题，并更新看板上的“ToDo”、“Doing”和“Done”三列。

所有团队成员相互协作，编码实现子任务的业务功能，详细的示例代码请见 Android 项目：SOAP，参考代码如下：

(1)创建布局文件：user_register_dialog. xml

```
<? xml version= "1.0"encoding= "utf-8"? >
<LinearLayout xmlns:android= "http://schemas.android.com/apk/res/android"
    android:layout_width= "600dip"
    android:layout_height= "500dip"
    android:background= "@drawable/dialog_bg"
    android:orientation= "vertical"
    android:padding= "20dp">
    <RelativeLayout
        android:layout_width= "fill_parent"
        android:layout_height= "90dip">

        <TextView
            android:layout_width= "wrap_content"
            android:layout_height= "wrap_content"
            android:layout_centerInParent= "true"
            android:text= "@string/register"
            android:textColor= "#292929"
            android:textSize= "25sp"/>
        <Button
            android:id= "@+id/close"
            android:layout_width= "wrap_content"
            android:layout_height= "wrap_content"
```

```
        android:layout_alignParentRight= "true"
        android:layout_centerVertical= "true"
        android:layout_marginRight= "20dp"
        android:background= "@drawable/dialog_close"/>
</RelativeLayout>
<View
    android:layout_width= "wrap_content"
    android:layout_height= "1dp"
    android:background= "@drawable/dialog_line"/>
<LinearLayout
    android:layout_width= "0dip"
    android:layout_height= "0dip"
    android:focusable= "true"
    android:focusableInTouchMode= "true"/>
<EditText
    android:id= "@+id/account_edit_text"
    android:layout_width= "match_parent"
    android:layout_height= "60dip"
    android:layout_marginLeft= "60dip"
    android:layout_marginRight= "60dip"
    android:layout_marginTop= "20dip"
    android:background= "@drawable/username_edit_bg"
    android:hint= "@string/username_hint"
    android:singleLine= "true"/>
<EditText
    android:id= "@+id/password_edit_text"
    android:layout_width= "match_parent"
    android:layout_height= "60dip"
    android:layout_marginLeft= "60dip"
    android:layout_marginRight= "60dip"
    android:layout_marginTop= "25dip"
    android:background= "@drawable/password_edit_bg"
    android:hint= "@string/password_hint"
    android:inputType= "textPassword"
    android:singleLine= "true"/>
<EditText
    android:id= "@+id/email_edit_text"
    android:layout_width= "match_parent"
    android:layout_height= "60dip"
    android:layout_marginLeft= "60dip"
    android:layout_marginRight= "60dip"
```

```
        android:layout_marginTop="25dip"
        android:background="@drawable/email_edit_bg"
        android:hint="@string/password_hint"
        android:inputType="textEmailAddress"
        android:singleLine="true"/>
    <Button
        android:id="@+id/ok_button"
        android:layout_width="fill_parent"
        android:layout_height="60dip"
        android:layout_marginLeft="60dip"
        android:layout_marginRight="60dip"
        android:layout_marginTop="25dip"
        android:background="@drawable/btn_green"
        android:text="@android:string/ok"
        android:textColor="@color/white"
        android:textSize="25sp"/>
</LinearLayout>
```

(2)创建布局文件:user_login. xml

```
<? xml version="1.0"encoding="utf-8"? >
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:background="@drawable/login_bg"
    android:gravity="center_horizontal"
    android:orientation="vertical">

    <TextView
        android:id="@+id/set_ip_text_view"
        android:layout_width="150dip"
        android:layout_height="50dip"
        android:layout_marginLeft="500dip"
        android:layout_marginTop="50dip"
        android:gravity="center"
        android:text="@string/setting_ip"
        android:textColor="@color/white"
        android:textSize="24sp"/>
    <LinearLayout
        android:layout_width="500dip"
        android:layout_height="wrap_content"
        android:layout_marginTop="110dip"
        android:background="@drawable/dialog_bg"
```

```
android:gravity="top|center_horizontal"
android:orientation="vertical"
android:padding="50dip">

<EditText
    android:id="@+id/account_edit_text"
    android:layout_width="match_parent"
    android:layout_height="60dip"
    android:background="@drawable/username_edit_bg"
    android:hint="@string/username_hint"
    android:singleLine="true"/>
<EditText
    android:id="@+id/password_edit_text"
    android:layout_width="match_parent"
    android:layout_height="60dip"
    android:layout_marginTop="25dip"
    android:background="@drawable/password_edit_bg"
    android:hint="@string/password_hint"
    android:inputType="textPassword"
    android:singleLine="true"/>
<Button
    android:id="@+id/login_button"
    android:layout_width="fill_parent"
    android:layout_height="60dip"
    android:layout_marginBottom="15dip"
    android:layout_marginTop="25dip"
    android:background="@drawable/btn_blue"
    android:text="@string/login"
    android:textColor="@color/white"
    android:textSize="25sp"/>
<LinearLayout
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:layout_marginTop="20dip"
    android:gravity="center"
    android:orientation="horizontal">
    <CheckBox
        android:id="@+id/recode_pwd_check_box"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/recode_password"
```

```
            android:textColor="@color/blue"
            android:textSize="20sp"/>
        <TextView
            android:id="@+id/forget_pwd_text_view"
            android:layout_width="0dip"
            android:layout_height="wrap_content"
            android:layout_marginLeft="20dip"
            android:layout_weight="1"
            android:text="@string/forget_password"
            android:textColor="@color/blue"
            android:textSize="20sp"/>

        <TextView
            android:id="@+id/register_text_view"
            android:layout_width="120dip"
            android:layout_height="40dip"
            android:layout_gravity="right|center_vertical"
            android:layout_marginLeft="20dip"
            android:background="@drawable/blue_textview_bg"
            android:gravity="center_vertical|center_horizontal"
            android:paddingLeft="5dip"
            android:text="@string/register"
            android:textColor="@color/blue"
            android:textSize="20sp"/>
        </LinearLayout>
    </LinearLayout>

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="v0.4"
        android:textColor="#33ff0000"/>

    </LinearLayout>
```

(3)添加布局文件:envir_fragment.xml

```
<? xml version="1.0"encoding="utf-8"? >
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical">
```

```
    <GridView
        android:id="@+id/sensor_grid_view"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:layout_marginTop="40dip"
        android:gravity="center_horizontal"
        android:horizontalSpacing="20dp"
        android:numColumns="3"
        android:padding="20dip"
        android:verticalSpacing="40dp">
    </GridView>

</LinearLayout>
```

(4)添加 Java 类:UserLoginActivity

```
/**
 * 用户登录界面,其中还包含了设置 IP 的对话框界面
 */
public class UserLoginActivity extends AppBaseActivity
{
    private static final String TAG ="UserLoginActivity";

    //设置 IP 的文本框
    private TextView mSetIPTV;
    //注册的文本框
    private TextView mRegisterTV;
    //找回密码的文本框
    private TextView mFindPwdTV;

    //用户名
    private EditText mUsernameET;
    //密码
    private EditText mPasswordET;
    //登录按钮
    private Button mLoginBtn;
    private CheckBox mSavePwdCheckBox;

    //用户登录请求对象
    private LoginRequest mLoginRequest;

    @Override
    protectedvoid onCreate(Bundle savedInstanceState)
```

```
{
    super.onCreate(savedInstanceState);
    setContentView(R.layout.user_login);

    initView();
    //创建用户登录请求对象
    mLoginRequest=new LoginRequest("");
    mLoginRequest.setOnResponseEventListener(new BaseRequest.OnResponseEventListener() {
        @Override
        public void onResponse(BaseRequest request, RequestResult result) {
            //处理用户登录请求的结果
            dismissLoadDialog();
            if (mLoginRequest.isSuccess()) {
                //然后进入主界面
                Intent intent=new Intent(UserLoginActivity.this, MainActivity.class);
                startActivity(intent);
                finish();
            }else {
                //登录失败则提示用户
                showAlertDialog(getString(R.string.prompt), getString(R.string.login
            _failed));
            }
        }
    });
}

private void initView()
{
    mSetIPTV=(TextView) findViewById(R.id.set_ip_text_view);
    mSetIPTV.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            //显示设置 IP 的对话框
            showSetIpDialog();
        }
    });

    mRegisterTV=(TextView) findViewById(R.id.register_text_view);
    mRegisterTV.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
```

```
                //显示注册对话框
                UserRegisterDialog dialog=new UserRegisterDialog(UserLoginActivity.this, mApp);
            dialog.show();
        }
    });

    mFindPwdTV=(TextView) findViewById(R.id.forget_pwd_text_view);
    mFindPwdTV.setEnabled(false);
    mFindPwdTV.setVisibility(View.INVISIBLE);
    mSavePwdCheckBox=(CheckBox) findViewById(R.id.recode_pwd_check_box);
    mSavePwdCheckBox.setChecked(mApp.isSavePwd());
    mSavePwdCheckBox.setOnCheckedChangeListener(new CompoundButton.OnCheckedChangeListener() {
        @Override
        public void onCheckedChanged(CompoundButton arg0, booleanarg1) {
            mApp.setSavePwd(arg1);
        }
    });

    mUsernameET=(EditText) findViewById(R.id.account_edit_text);
    mUsernameET.setText(mApp.getUsername());
    mPasswordET=(EditText) findViewById(R.id.password_edit_text);
    if (mApp.isSavePwd()) {
        mPasswordET.setText(mApp.getPassword());
    }
    mLoginBtn=(Button) findViewById(R.id.login_button);
    mLoginBtn.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            //开始登录
            userLoginEntry();
        }
    });
}

//开始登录
private void userLoginEntry()
{
    String username=mUsernameET.getText().toString().trim();
    String password=mPasswordET.getText().toString().trim();

    //先检查用户名和密码
```

```
        if (username.equals("")) {
            showAlertDialog(getString(R.string.prompt), getString(R.string.username_not_empty));
            return;
        }
        if (password.equals("")) {
            showAlertDialog(getString(R.string.prompt), getString(R.string.password_not_empty));
            return;
        }
        //先保存用户名和密码
        mApp.setUsername(username);
        if (mApp.isSavePwd()) {
            mApp.setPassword(password);
        }
        //将用户名和密码封装到请求中
        mLoginRequest.setUsername(username);
        mLoginRequest.setPassword(password);
        //开始向服务器端发送请求
        startRequest(mLoginRequest);
        //显示等待框
        showLoadDialog();
    }

    // show setting IP address dialog
    private void showSetIpDialog()
    {
        final Dialog myDialog=new Dialog(this);// 创建一个对话框
        myDialog.show();
        myDialog.getWindow().setBackgroundDrawable(new ColorDrawable(0)); // 设置背景颜色
        //设置对话框的界面布局
        Windowwin=myDialog.getWindow();
        win.setContentView(R.layout.set_ip_dialog);
        //设置关闭按钮
        win.findViewById(R.id.close).setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                myDialog.dismiss();
            }
        });

        final EditText ip1ET=(EditText) win.findViewById(R.id.ip1_edit_text);
        final EditText ip2ET=(EditText) win.findViewById(R.id.ip2_edit_text);
```

```
    final EditText ip3ET=(EditText) win.findViewById(R.id.ip3_edit_text);
    final EditText ip4ET=(EditText) win.findViewById(R.id.ip4_edit_text);
    //显示上次设置的 ip 地址
    String ipStr[]=mApp.getServerIpStr().split("\\.");
    if (ipStr.length>= 4)
    {
        ip1ET.setText(ipStr[0]);
        ip2ET.setText(ipStr[1]);
        ip3ET.setText(ipStr[2]);
        ip4ET.setText(ipStr[3]);
        }

        win.findViewById(R.id.ok_button).setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                //点击 OK 以后,设置 IP 地址
                try
                {
                    int ip1, ip2, ip3, ip4;
                    ip1=Integer.parseInt(ip1ET.getText().toString().trim());
                    ip2=Integer.parseInt(ip2ET.getText().toString().trim());
                    ip3=Integer.parseInt(ip3ET.getText().toString().trim());
                    ip4=Integer.parseInt(ip4ET.getText().toString().trim());
                    String ipStr=ip1 + "." + ip2 + "." + ip3 + "." + ip4;
                    //保存新的 IP 地址
                    mApp.setServerIpStr(ipStr);
                    Log.d(TAG, "set server IP:" + ipStr);
                    myDialog.dismiss();
                }catch (Exception e) {
                    showAlertDialog(getString(R.string.prompt),getString(R.string.
                invalid_ip));
                }
            }
        });
    }
}
```

(5)添加 Java 类:EnvirFragment.java

```
/**
*环境指标子窗体
*/
public class EnvirFragment extends AppBaseFragment
```

```
{
    public static final String TAG ="EnvirFragment";
    //所有传感器数据的集合
    private ArrayList<SensorBean>mSensorList;
    //传感器数据值适配器
    private SensorGridAdapter mAdp;
    //存放所有传感器 view 的容器
    private GridView mGridView;

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        mSensorList=new ArrayList<SensorBean>();
        mSensorList.add(new SensorBean(getString(R.string.CO2_title)));// CO2 浓度
        mSensorList.add(new SensorBean(getString(R.string.light_title)));// 光照强度
        mSensorList.add(new SensorBean(getString(R.string.air_tmper_title)));// 空气温度
        mSensorList.add(new SensorBean(getString(R.string.air_humid_title)));// 空气湿度
        mSensorList.add(new SensorBean(getString(R.string.soil_tmper_title)));// 土壤温度
        mSensorList.add(new SensorBean(getString(R.string.soil_humid_title)));// 土壤湿度

        mAdp=new SensorGridAdapter(getActivity(), mSensorList);
    }

    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle
savedInstanceState) {
        returninflater.inflate(R.layout.envir_fragment , container, false);
    }

    @Override
    public void onActivityCreated(Bundle savedInstanceState)
    {
        super.onActivityCreated(savedInstanceState);

        mGridView=(GridView) getView().findViewById(R.id.sensor_grid_view);
        mGridView.setAdapter(mAdp);
        updateView();
    }

    @Override
    public void onDestroy() {
```

```
    super.onDestroy();
}

@Override
public void onDestroyView() {
    super.onDestroyView();
}

//刷新环境指标子窗体界面
public void updateView()
{
    SensorValue value=mApp.getCurSensorValue();
    SensorConfig config=mApp.getSensorConfig();
    if (value != null&&config != null)
    {
        // C02 浓度
        SensorBeanco2=mSensorList.get(0);
        co2.setValue(value.getCo2());
        co2.setMinValue(config.minCo2);
        co2.setMaxValue(config.maxCo2);

        //光照强度
        SensorBean light=mSensorList.get(1);
        light.setValue(value.getLight());
        light.setMinValue(config.minLight);
        light.setMaxValue(config.maxLight);

        //空气温度
        SensorBean airT=mSensorList.get(2);
        airT.setValue(value.getAirTemper());
        airT.setMinValue(config.minAirTemperature);
        airT.setMaxValue(config.maxAirTemperature);

        //空气湿度
        SensorBean airH=mSensorList.get(3);
        airH.setValue(value.getAirHumid());
        airH.setMinValue(config.minAirHumidity);
        airH.setMaxValue(config.maxAirHumidity);

        //土壤温度
        SensorBean soilT=mSensorList.get(4);
```

```
            soilT.setValue(value.getSoilTemper());
            soilT.setMinValue(config.minSoilTemperature);
            soilT.setMaxValue(config.maxSoilTemperature);

            //土壤湿度
            SensorBean soilH=mSensorList.get(5);
            soilH.setValue(value.getSoilHumid());
            soilH.setMinValue(config.minSoilHumidity);
            soilH.setMaxValue(config.maxSoilHumidity);

            mAdp.notifyDataSetChanged();
        }
    }
}
```

(6)添加 Java 类:UserRegisterDialog.java

```
/**
*用户注册对话框
*/
public class UserRegisterDialog extends AppBaseDialog
{
    //用户注册的请求对象
    private RegisterRequest mRegisterRequest;

    private EditText mUsernameET;
    private EditText mPasswordET;
    private EditText mEmailET;

    private Button mCloseBtn;
    private Button mPostBtn;

    public UserRegisterDialog(Activity act, ClientApp app)
    {
        super(act, app);
    }

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        this.setContentView(R.layout.user_register_dialog);
        getWindow().setBackgroundDrawable(new ColorDrawable(0));
        initView();
```

```
    mRegisterRequest=new RegisterRequest("");
    mRegisterRequest.setOnResponseEventListener(new BaseRequest.OnResponseEventListener() {
        @Override
        public void onResponse(BaseRequest request, RequestResult result) {
            //处理用户注册请求的结果
            dismissLoadDialog();
            if (mRegisterRequest.isSuccess()) {
                //如果成功,则提示用户
                showAlertDialog(getString(R.string.prompt),
                    getString(R.string.register_success));
                UserRegisterDialog.this.dismiss();
            }else {
                showAlertDialog(getString(R.string.prompt), getString(R.string.
            register_failed));
            }
        }
    });
}

private void initView()
{
    mCloseBtn=(Button) findViewById(R.id.close);
    mCloseBtn.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            UserRegisterDialog.this.dismiss();
        }
    });

    mUsernameET=(EditText) findViewById(R.id.account_edit_text);
    mPasswordET=(EditText) findViewById(R.id.password_edit_text);
    mEmailET=(EditText) findViewById(R.id.email_edit_text);

    mPostBtn=(Button) findViewById(R.id.ok_button);
    mPostBtn.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            userRegisterEnter();
        }
    });
```

```
    }

    private void userRegisterEnter()
    {
        String username＝mUsernameET.getText().toString().trim();
        String password＝mPasswordET.getText().toString().trim();
        String email＝mEmailET.getText().toString().trim();
        //检查用户名,密码和邮箱地址
        if (username.equals("")) {
            showAlertDialog(getString(R.string.prompt), getString(R.string.username_
        not_empty));
            return;
        }
        if (password.equals("")) {
            showAlertDialog(getString(R.string.prompt), getString(R.string.password_
        not_empty));
            return;
        }
        if (email.equals("")) {
            showAlertDialog(getString(R.string.prompt), getString(R.string.email_not
        _empty));
            return;
        }
        if (! Util.isEmail(email)) {
            showAlertDialog(getString(R.string.prompt), getString(R.string.invalid
        _email));
            return;
        }

        mRegisterRequest.setUsername(username);
        mRegisterRequest.setPassword(password);
        mRegisterRequest.setEmail(email);
        //开启发起请求
        startRequest(mRegisterRequest);
        //显示等待提示框
        showLoadDialog();
    }
}
```

(7)添加 Java 类:RequestThread.java

```
/**
 * 请求执行线程,支持 http 和 socket 两种通信方式,其端口设置见 AppConfig 文件
```

```
 * 该线程既可以执行一次请求然后就结束，也可以循环重复执行同一个请求，isLoop 变量可以进行控制
 */
public class RequestThread extends Thread
{
    private static final String TAG = "RequestThread";
    //日志开关
    private static final boolean LOG_ENABLE = false;

    //支持 http 和 socket 两种通信方式
    public static final String COMMUN_HTTP = "http";
    public static final String COMMUN_SOCKET = "socket";
    //消息 id，请求完成以后，需要将此 id 发送给 ui 主线程处理
    public static final int MSG_REQUEST_RESULT = 0×10;
    // ui 主线程的 Handler
    private Handler mHandler;
    // app 上下文
    private Context mContext;
    //通信方式
    private String mCommunType;
    //请求实体对象
    private BaseRequest mRequest;
    // app 对象
    private ClientApp mApp;

    //线程是否已经被取消的控制变量
    private volatile boolean mCancel=false;
    //线程是否应该循环执行的控制变量
    private volatile boolean isLoop=false;
    //循环执行的时间间隔
    private volatile int loopPeriod=1000;// 默认为 1 秒
    //线程是否已经被暂停的控制变量
    private volatile boolean isPause=false;

    public RequestThread(String communType, Context context, Handler handler)
    {
        mContext=context;
        mHandler=handler;
        this.mCommunType=communType;
        mApp=(ClientApp) context.getApplicationContext();
    }

    //请求执行线程装载请求实体
```

```
publicvoid setRequest(BaseRequest mRequest) {
    this.mRequest=mRequest;
}

//停止请求线程
public void stopRequestThread()
{
    mCancel=true;
    isLoop=false;
}

//请求线程是否已经停止
public boolean isCancel()
{
    returnmCancel;
}

//打印日志
private void showLog(String msg)
{
    if (LOG_ENABLE) {
        Log.d(TAG, msg);
    }
}

protected void runBefore()
{
}

@Override
public void run()
{
    super.run();
    runBefore();
    do
    {
        if (!isPause)
        {
            RequestResult result=RequestResult.RESULT_FAIL;
            //先判断网络状态
            if (NetUtil.isNetworkAvailable(mContext))
            {
                try
```

```
{
    if (mRequest ! = null&&mApp ! = null)
    {
        //获取协议类型
        String protocolType=mRequest.getProtocolType();
        //获取 action 名称
        String actionName=mRequest.getActionName();
        //获取请求 body
        String requestBody=mRequest.getBody();
        if (protocolType ! = null&&actionName ! = null&&
    requestBody ! = null)
        {
            //采用 http POST 的通信方式
            if (mCommunType.equals(COMMUN_HTTP))
            {
                // url 组装
                String url="http://" + mApp.getServerIpStr() + ":"
                + AppConfig.HTTP_SERVER_PORT;
                url += "/type/" + protocolType;// 设置协议类型
                url += "/action/" + actionName;// 设置 action 名称
                showLog("Http Url:" + url);
                showLog("Http body:" + requestBody);
                Stringresponse="";
                response = NetUtil.sendSoap(url, actionName,
            requestBody);
                //采用 http post 的方式发送 soap 报文
                //将服务器端回应的结果保存到请求对象中
                mRequest.setResponseStr(response);
                showLog("Http response:" + response);
                result=RequestResult.RESULT_SUCCESS;
            }
            else if (mCommunType.equals(COMMUN_SOCKET))// 采用
        socket 通信方式
            {
                String outMsg="";
                //封装请求数据
                // body 字段的内容需要经过 Base64 编码
                String bodyBase64=Base64.encodeToString(
                    requestBody.getBytes(), Base64.DEFAULT);
                JSONObjectjsonObj=new JSONObject();
                try
                {
                jsonObj.put("type", protocolType);// 设置协议类型
```

```
        jsonObj.put("action", actionName);// 设置 action 名称
        jsonObj.put("body", bodyBase64);// 设置 body 内容
        outMsg=jsonObj.toString();
    }catch (JSONException e) {
        e.printStackTrace();
    }

    String receiveMsg="";
    //采用 socket 通信方式
    Socket clientSocket=null;
    try
    {
        clientSocket=new Socket(mApp.getServerIpStr(),
            AppConfig.SOCKET_SERVER_PORT);
        clientSocket.setSoTimeout(30 * 1000);// 设置超时
        BufferedReader in=new BufferedReader(
            new InputStreamReader ( clientSocket.
       getInputStream()));
            //先向服务器端发送数据,以后回车为结束标记
            PrintWriterout=new PrintWriter(
                clientSocket.getOutputStream());
            out.println(outMsg);
            out.flush();

            //再读取服务器端的回应数据,一行一行读取
            Stringline;
            while ((line=in.readLine()) ! = null)
            {
                if (line.equals("")) {
                    break;
                }
                receiveMsg + = line;
            }
        }catch (UnknownHostException e) {
            e.printStackTrace();
        }catch (IOException e) {
            e.printStackTrace();
        }finally
        {
            if (clientSocket ! = null)
            {
                clientSocket.close();
            }
```

```
                    }

                    //数据解析,Base64 解码
                    bytetmpB [ ] = Base64. decode ( receiveMsg,
                Base64. DEFAULT );
                    StringreceBody=new String(tmpB);
                    //将服务器端回应的结果保存到请求对象中
                    mRequest. setResponseStr(receBody);
                    showLog("socket response:" + receBody);
                    result=RequestResult. RESULT_SUCCESS ;
                }
            }
        }
    }catch (Exception e)
    {
        result=RequestResult. RESULT_FAIL ;
        e. printStackTrace();
    }
}
else
{
    result=RequestResult. RESULT_NO_NET ;
}

//请求完成以后,将此结果发送给 ui 主线程处理
if (! mCancel&&mHandler ! = null)
{
    Message msg=new Message();
    msg. what= MSG_REQUEST_RESULT ;
    msg. obj=this;
    msg. arg1=result. ordinal();
    mHandler. sendMessage(msg);
}

//如果时循环执行线程,则延时
if (isLoop)
{
    try {
        Thread. sleep (loopPeriod);
    }catch (InterruptedException e) {
        e. printStackTrace();
    }
}
```

```
            }
            else
            {
                //如果线程被暂停,则线程延时
                try {
                    Thread.sleep(loopPeriod);
                    continue;
                }catch (InterruptedException e) {
                    e.printStackTrace();
                }
            }
        }while (isLoop);
    }

    //该函数只能由 ui 主线程来调用,其他线程不能调用,目的是让 ui 主线程处理请求结果,以便界面刷新
    public void hanlderResult(RequestResult result)
    {
        if (! mCancel&&mRequest ! = null)
        {
            mRequest.parseResult(result);
        }
    }

    //判断线程是否暂停
    public void pause() {
        isPause=true;
    }

    //线程重新开始
    public void restart() {
        isPause=false;
    }

    //设置线程是否为循环执行
    public void setLoop(boolean isLoop, int loopPeriod)
    {
        this.isLoop=isLoop;
        this.loopPeriod=loopPeriod;
    }
}
```

(8)添加 Java 类:LoginRequest.java

```
/* *
```

```
 * 用户登录请求类
 */
public class LoginRequest extends AgricultureRequest
{
    // action name
    private static final String ACTION = "login";

    private String username;
    private String password;

    public LoginRequest(String username) {
        super(username);
    }

    @Override
    protected String getActionName() {
        return ACTION ;
    }

    public String getUsername() {
        return username;
    }

    public void setUsername(String username) {
        this.username = username;
    }

    public String getPassword() {
        return password;
    }

    public void setPassword(String password) {
        this.password = password;
    }

    // soap 协议时的获取 body 函数
    @Override
    protected String onGetSoapBody()
    {
        // 用户名
        String soapBody = Util.getXmlElementStr("username", username);
```

```
        // 密码
        soapBody += Util.getXmlElementStr("password", password);
        return Util.getSoapXml(soapBody);
    }
}
```

确保以上每一步都编译通过，在 IDE 集成开发环境中，没有出现红叉的错误提示。应用程序的运行效果如图 4-2-1 所示。

图 4-2-1　应用程序运行效果图

点击用户注册按钮，并输入用户名和密码，如图 4-2-2 所示。

图 4-2-2　用户注册界面

注册成功后，如图 4-2-3 所示。

图 4-2-3　用户注册成功界面

用注册的账号进行登录，如图 4-2-4 所示。

图 4-2-4　登录界面

登录成功后，进入客户端 App 的主界面，如图 4-2-5 所示。

图 4-2-5　App 主界面

开发工程师之间两两交叉代码评审，并根据评审纪要重构代码。

所有团队成员召开评审会，并填写表 4-2-3。

表 4-2-3　　评审建议表

序号	建议描述	是否接纳	原因
1		□接纳　□部分接纳　□未接纳	
2		□接纳　□部分接纳　□未接纳	
3		□接纳　□部分接纳　□未接纳	
4		□接纳　□部分接纳　□未接纳	
5		□接纳　□部分接纳　□未接纳	
6		□接纳　□部分接纳　□未接纳	
7		□接纳　□部分接纳　□未接纳	
8		□接纳　□部分接纳　□未接纳	
9		□接纳　□部分接纳　□未接纳	
10		□接纳　□部分接纳　□未接纳	

所有团队成员召开复盘会，并填写表 4-2-4。

表 4-2-4　　复盘会总结表

回顾目标	评估结果
当初的目的是什么（期望的结果）	Highlights (与原来目标比）
要达成的目标 & 里程碑	Lowlights (与原来目标比）
Goal 1　2 Result	
Insight 3　4 Analysis	
总结规律	**分析原因**
经验 & 规律（不要轻易下结论）	成功关键因素（主观/客观）
行动计划	失败根本原因（主观/客观）
新举措：	
叫停：	
继续：	

同学们，根据塔克曼阶梯理论，你们的团队目前处于哪个阶段呢？如果还没有进入成熟阶段，那就让我们一起努力吧，在接下来的任务中，更加注重相互配合和协作，及时发现并改正团队配合过程中出现的问题，早日让我们的团队进入成熟阶段！

4.2.5　任务扩展(Extend)

课后练习：使用 SOAP 协议在客户端和服务端之间传递一个普通的对象。

子任务 3　使用 Socket 实现客户端与服务端的通信

• 子任务目标：

➢ 应用 Socket 和 SocketServer，实现客户端与服务端的数据通信

• 课时分配：

➢ 8 课时

4.3.1　任务构思(Conceive)

在智能农业实战项目中，服务端从沙盘获取数据，然后将数据传给客户端，客户端提供客户浏览数据的功能，并且设定好环境指标保存在服务端，服务端将沙盘传来的数据与环境指标进行对比，如果沙盘中的指标不在环境指标范围内，则发出告警通知。因此，

客户端与服务端需要进行数据通信。除了 HTTP、SOAP 协议外，使用 Socket 和 SocketServer 也可以实现客户端与服务端的数据通信。

4.3.2　任务设计(Design)

使用 Socket 实现客户端与服务端的通信要分别实现客户端 App 和服务端 App。

服务端 App：

(1)指定 Socket 通信的端口 8891。

(2)写一个 SocketServerThread 类，继承 Thread，实现一个微型 Socket Server。在这个类中，当监听到有客户端连接时就创建客户端，然后接收客户端发来的数据。

(3)写一个 NotificationServerThread 类，继承 Thread。实现一个微型的 Push Server。在这个类中，当有客户端发来提示信息时，服务端能监听到，然后接收并发出提示信息。

(4)服务端要能够从请求中解析出用户名和密码，然后在数据库中进行查询，检查其是否正常，并且从客户端请求 body 中取得用户名、密码和邮箱，然后向数据库中插入一条记录，并将结果返回给 Socket 服务器。

客户端 App：

(1)指定 Socket 通信的端口 8891。

(2)采用 Socket 的通信方式向服务器发送登录请求。

(3)采用 Socket 的通信方式向服务器发送注册请求。

(4)向服务器发送找回密码请求。如果密码成功发送到用户邮箱，则服务器返回报文中会携带该邮箱地址，并给出提示。

4.3.3　任务实现(Implement)

智能农业实战项目的该子任务包含的角色为开发工程师。角色扮演的过程如表 4-3-1所示。

表 4-3-1　　角色扮演过程

步骤	角色	工作	输出
1	开发工程师	编码实现子任务的业务功能	实现子任务业务功能的 Android 源代码
2	开发工程师	两两交叉代码评审	代码评审纪要
3	开发工程师	根据代码评审纪要重构代码	重构后的 Android 源代码

4.3.4　任务运作(Operate)

所有团队成员召开计划会议，并将会议纪要记录在表 4-3-2 中。

表 4-3-2　　会议纪要记录表

5W2H	结论
WHAT	
HOW	
WHY	
WHEN	
WHERE	
WHO	
HOW MUCH	

在计划会议结束之前，所有团队成员把子任务进一步细分为更小的任务，把这些任务写在便签上，并把便签贴在看板上的“ToDo”列。

所有团队成员每天早上召开 15 分钟的站立会议，每人汇报三个问题，并更新看板上的“ToDo”、“Doing”和“Done”三列。

所有团队成员相互协作，编码实现子任务的业务功能，详细的示例代码请见 Android 项目：SocketServer 和 SocketClient，参考代码如下：

(1)在服务端创建 Java 文件：SimpleServer.java

```
public class SimpleServer {
    public static void main(String[] args) throws IOException {
        //创建一个 ServerSocket,用于监听客户端 socket 的连接请求
        ServerSocket ss =new ServerSocket(30000);
        //采用循环不断接受来自客户端的请求,服务器端也对应产生一个 Socket
        while(true) {
            //服务端等待连接
            Socket s = ss.accept();
            OutputStream os = s.getOutputStream();
            os.write("欢迎来到智能农业系统\n".getBytes("utf-8"));
            //获取输入流
            BufferedReader br =new BufferedReader(new InputStreamReader(
                s.getInputStream()));
            String line = br.readLine();
            System.out.println(line);
            os.close();
            s.close();
        }
    }
}
```

(2)在客户端创建布局文件：activity_main.xml

```
<? xml version= "1.0"encoding= "utf-8"? >
```

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:gravity="center"
    android:orientation="vertical">

    <TextView
        android:id="@+id/huanyin"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginLeft="18dp"
        android:layout_marginTop="22dp"
        android:textSize="30sp"/>

    <TextView
        android:id="@+id/air"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentLeft="true"
        android:layout_below="@+id/huanyin"
        android:layout_marginLeft="45dp"
        android:layout_marginTop="37dp"
        android:text="沙盘现在:\n空气湿度 26 度\n空气温度 35 度\n土壤温度 6 度\n土壤
    湿度 8 度"
        android:textSize="25sp"/>

</RelativeLayout>
```

(3)添加 Java 类:MainActivity.java

```
public class MainActivity extends Activity {
    private TextView huanyin;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main );
        // 这个 textview 用来显示服务器传过来的数据
        huanyin = (TextView) this.findViewById(R.id.huanyin );
        // 创建线程进行通信
        new Thread() {
            @Override
            public void run() {
```

```
                try {
                    // 在实例化的 socket 对象中指明服务器的 ip 和端口号
                    Socketsocket = new Socket("192.168.130.108", 30000);
                    // 延迟的时间
                    socket.setSoTimeout(10000);
                    // 获取服务器传过来的数据(inputStream()是输入流)
                    BufferedReaderbr = new BufferedReader(
                        new InputStreamReader(socket.getInputStream()));
                    Stringline = br.readLine();
                    huanyin.setText(line);
                    // 向服务器传数据(outputStream()是输出流)
                    OutputStreamos = socket.getOutputStream();
                    PrintWriterpw = new PrintWriter(os);
                    // 从 strings.xml 中获取 name 为 airinfo 的字符串的内容。
                    Stringinfo = getString(R.string.airinfo);
                    pw.write(info);
                    // 刷新缓冲区
                    pw.flush();
                    // 当完成了对一个流的读取或者写入后,就应该调用 close 方法将它关
                闭,这样可以释放流所占用的操作系统资源。
                    br.close();
                    socket.close();

                }catch (IOException e) {
                    e.printStackTrace();
                }
            }
        }.start();
    }
}
```

(4)修改 AndroidManifest.xml 文件

```
<manifest xmlns:android= "http://schemas.android.com/apk/res/android"
    package= "com.lenovo.demo"
    android:versionCode= "1"
    android:versionName= "1.0">

    <uses-sdk
        android:minSdkVersion= "15"
        android:targetSdkVersion= "19"/>

    <application
```

```
        android:allowBackup="true"
        android:icon="@drawable/app"
        android:label="@string/app_name">
        <activity android:name=".activity.MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN"/>
                <category android:name="android.intent.category.LAUNCHER"/>
            </intent-filter>
        </activity>
    </application>
    <uses-permission android:name="android.permission.INTERNET"/>
</manifest>
```

确保以上每一步都编译通过：在 IDE 集成开发环境中，没有出现红叉的错误提示。应用程序的运行效果如图 4-3-1 所示。

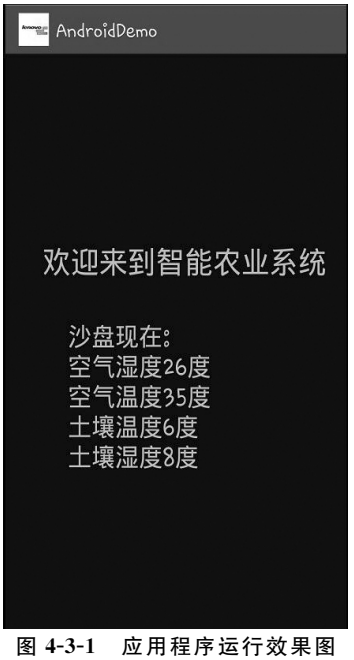


图 4-3-1　应用程序运行效果图

智能农业实战项目完整源代码中的核心代码片段如下：

```
while (! socket.isClosed())
{
    //读取客户端发送的内容
    String receiveStr;
    receiveStr=in.readLine();
    if(receiveStr==null)
      break;
    if(receiveStr! =null && ! receiveStr.equals(""))
```

```
    {
    String action="";
    String type="";
    String strBody="";
    //从内容中解析协议类型,action 和协议 body
try
{
    JSONObject jsonRespObj=new JSONObject(receiveStr);
    if(jsonRespObj != null)
    {
        //解析协议类型
        if(jsonRespObj.has("type")){
            type=jsonRespObj.getString("type");
        }
        //解析 action
        if(jsonRespObj.has("action")){
            action=jsonRespObj.getString("action");
        }
        //解析协议 body
        if(jsonRespObj.has("body")){
            String tmpStr=jsonRespObj.getString("body");
            //base64 解码
            byte tmpB[]=Base64.decode(tmpStr, Base64.DEFAULT);
            strBody=new String(tmpB);
        }
        //根据协议类型和 action,解析协议内容并返回相关结果
        String socketResp=BaseAction.disposeAction(type, action, strBody, mContext);
        out.println(Base64.encodeToString(socketResp.getBytes(), Base64.DEFAULT));
    }
}
while (! socket.isClosed())
{
//读取客户端发送的内容
  String receiveStr;
  receiveStr=in.readLine();
  if(receiveStr==null)
    break;
  if(receiveStr! =null && ! receiveStr.equals(""))
  {
    if(receiveStr.equals("getNotification"))
    {
```

```
            String socketResp="";
            String notifiMsg="";
            if(isCanStartNoitifiServer()){
                checkAllSensorStatus();
                notifiMsg=getNotificationMsg();
            }
            JSONObject jsonResponse=new JSONObject();
            try {
            jsonResponse.put("notifiId", mNotifiId);
            jsonResponse.put("notifiMsg", notifiMsg);
            socketResp=jsonResponse.toString();
            } catch (JSONException e) {
            e.printStackTrace();
            }
            out.println(Base64.encodeToString(socketResp.getBytes(), Base64.DEFAULT));
            }
            out.flush();
        }
    }
```

开发工程师之间两两交叉代码评审，并根据评审纪要重构代码。

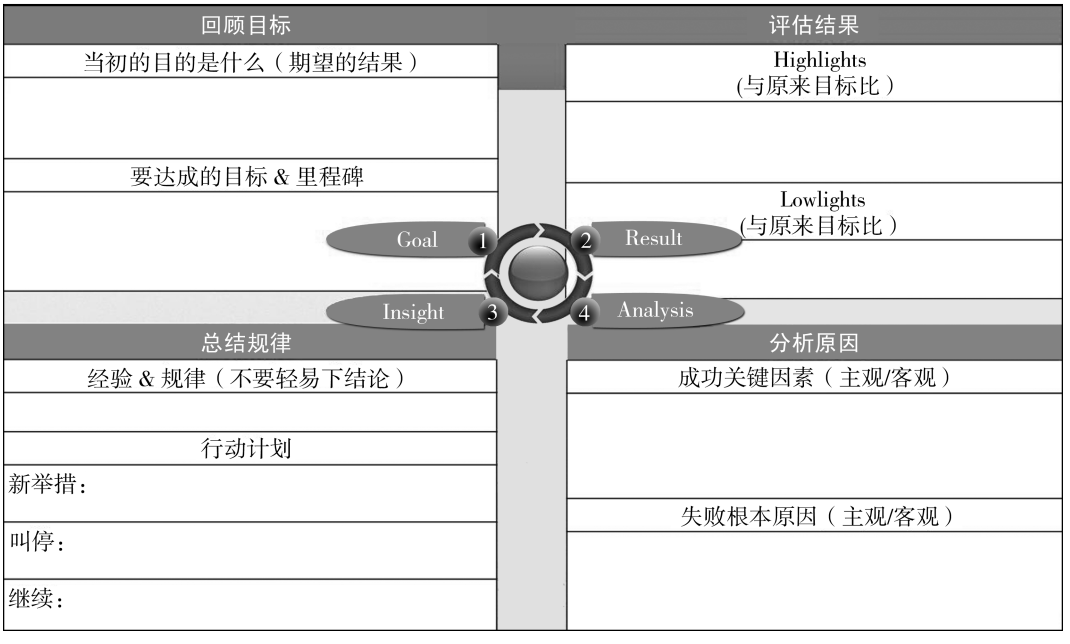
所有团队成员召开评审会，并填写表 4-3-3。

表 4-3-3　　评审建议表

序号	建议描述	是否接纳	原因
1		□接纳　□部分接纳　□未接纳	
2		□接纳　□部分接纳　□未接纳	
3		□接纳　□部分接纳　□未接纳	
4		□接纳　□部分接纳　□未接纳	
5		□接纳　□部分接纳　□未接纳	
6		□接纳　□部分接纳　□未接纳	
7		□接纳　□部分接纳　□未接纳	
8		□接纳　□部分接纳　□未接纳	
9		□接纳　□部分接纳　□未接纳	
10		□接纳　□部分接纳　□未接纳	

所有团队成员召开复盘会，并填写表 4-3-4。

表 4-3-4　　复盘会总结表



同学们，根据塔克曼阶梯理论，你们的团队目前处于哪个阶段呢？如果还没有进入成熟阶段，那就让我们一起努力吧，在接下来的任务中，更加注重相互配合和协作，及时发现并改正团队配合过程中出现的问题，早日让我们的团队进入成熟阶段！

4.3.5　任务扩展(Extend)

课后练习：使用 Socket 和 SocketServer 在服务端和客户端之间传送一个普通的对象。

子任务 4　实时采集传感器数据

• 子任务目标：

➢ 应用 Handler 对传感器的数据进行定时采集

• 课时分配：

➢ 4 课时

4.4.1　任务构思(Conceive)

在智能农业实战项目中，智能农业大棚中的环境指标数据不停地发生变化，从而传感器读到的数据也在不停地发生变化。所以需要一个定时器，每隔一秒对大棚中的传感

器数据进行定时采集。

4.4.2　任务设计(Design)

在 Android 系统中，Handler 是最常用的定时操作类。

4.4.3　任务实现(Implement)

智能农业实战项目的该子任务包含的角色为开发工程师。角色扮演的过程如表4-4-1所示。

表 4-4-1　　角色扮演过程

步骤	角色	工作	输出
1	开发工程师	编码实现子任务的业务功能	实现子任务业务功能的 Android 源代码
2	开发工程师	两两交叉代码评审	代码评审纪要
3	开发工程师	根据代码评审纪要重构代码	重构后的 Android 源代码

4.4.4　任务运作(Operate)

所有团队成员召开计划会议，并将会议纪要记录在表 4-4-2 中。

表 4-4-2　　会议纪要记录表

5W2H	结论
WHAT	
HOW	
WHY	
WHEN	
WHERE	
WHO	
HOW MUCH	

在计划会议结束之前，所有团队成员把子任务进一步细分为更小的任务，把这些任务写在便签上，并把便签贴在看板上的“ToDo”列。

所有团队成员每天早上召开 15 分钟的站立会议，每人汇报三个问题，并更新看板上的“ToDo”、“Doing”和“Done”三列。

所有团队成员相互协作，编码实现子任务的业务功能，详细的示例代码请见 Android 项目：Handler，参考代码如下：

(1)添加布局文件：activity_main. xml

```
<? xml version= "1.0"encoding= "utf-8"? >
<LinearLayout xmlns:android= "http://schemas.android.com/apk/res/android"
```

```
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical">
    <Button
    android:id="@+id/button1"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="@string/show_horizontal_dialog"/>

</LinearLayout>
```

(2)添加 Java 类:MainActivity.java

```
/**
* 主页面对应的 Activity
*/
public class MainActivity extends Activity implements OnClickListener
{
    private Button mButton1;
    private static final int MAX_PROGRESS = 100;
    private ProgressDialog mProgressDialog;
    private Handler mHandler;
    private int mProgress;

    @Override
    public void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        findViewsById();
        setListeners();
    }

    private void findViewsById() {
        mButton1 = (Button) findViewById(R.id.button1);
    }

    private void setListeners() {
        mButton1.setOnClickListener(this);
    }

    @Override
    public void onClick(View view)
    {
        switch (view.getId())
```

```
    {
        case R.id.button1:
            showProgressDialog(ProgressDialog.STYLE_HORIZONTAL);
            break;
        default:
            break;
    }
}

@SuppressWarnings("deprecation")
private void showProgressDialog(int style)
{
    mProgressDialog = new ProgressDialog(this);
    mProgressDialog.setIcon(R.drawable.wait);
    mProgressDialog.setTitle(getResources().getString(R.string.title));
    mProgressDialog.setMessage(getResources().getString(R.string.degree_intime));
    mProgressDialog.setProgressStyle(style);
    mProgressDialog.setMax(MAX_PROGRESS);
    mProgressDialog.setButton2(getResources().getString(R.string.cancle), new
DialogInterface.OnClickListener()
    {
        public void onClick(DialogInterface dialog, int whichButton)
        {
            // 删除消息队列中的消息来停止定时器
            mHandler.removeMessages(1);
            // 恢复进度初始值
            mProgress = 0;
            mProgressDialog.setProgress(0);
        }
    });

    mProgressDialog.setButton(getResources().getString(R.string.pause), new
DialogInterface.OnClickListener()
    {
        public void onClick(DialogInterface dialog, int whichButton)
        {
            // 删除消息队列中的消息来停止定时器
            mHandler.removeMessages(1);
        }
    });

    mProgressDialog.show();
```

```
        mHandler = new Handler()
        {
            @Override
            public void handleMessage(Message msg)
            {
                super.handleMessage(msg);

                if (mProgress >= MAX_PROGRESS)
                {
                    // 进度达到最大值,关闭对话框
                    mProgress = 0;
                    mProgressDialog.dismiss();
                }
                else
                {
                    mProgress++;
                    // 将进度递增 1
                    mProgressDialog.incrementProgressBy(1);
                    // 随机设置下一次递增进度(调用 handleMessage()方法)的时间间隔
                    // 第一个参数表示消息代码,第二个参数表示下一次调用 handleMessage
                ()方法要等待的毫秒数
                    mHandler.sendEmptyMessageDelayed(1, 500);
                }
            }
        };

        mProgress = (mProgress > 0) ? mProgress : 0;
        // 必须在对话框显示出来后再设置当前进度
        mProgressDialog.setProgress(mProgress);
        mHandler.sendEmptyMessage(1);
    }
}
```

(3)修改 AndroidManifest.xml 文件

```
<manifest xmlns:android= "http://schemas.android.com/apk/res/android"
    package= "com.lenovo.demo"
    android:versionCode= "1"
    android:versionName= "1.0">

    <uses-sdk
        android:minSdkVersion= "10"
        android:targetSdkVersion= "19"/>

    <application
```

```
        android:allowBackup="true"

        android:label="@string/app_name"
        android:theme="@style/AppTheme">
        <activity
            android:name=".activity.MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN"/>

                <category android:name="android.intent.category.LAUNCHER"/>
            </intent-filter>
        </activity>
    </application>

</manifest>
```

确保以上每一步都编译通过：在 IDE 集成开发环境中，没有出现红叉的错误提示。应用程度的运行效果如图 4-4-1 所示。

点击“增强光照强度”按钮，如图 4-4-2 所示。

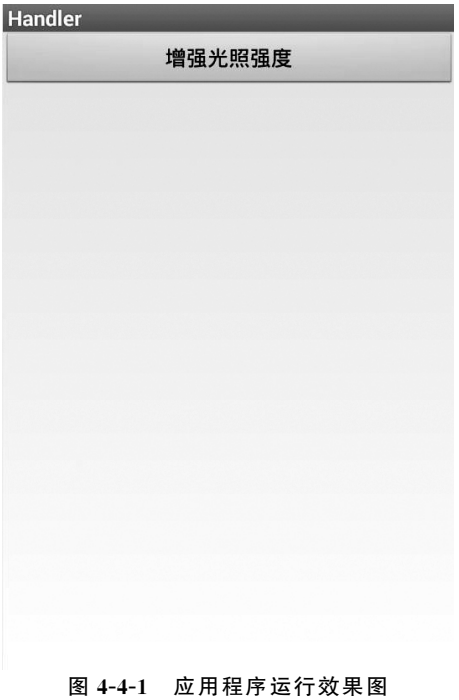


图 4-4-1　应用程序运行效果图

图 4-4-2　“增强光照强度”界面

智能农业实战项目完整源代码中的核心代码片段如下：

```
//获取外接多个传感模块地址，用于后续对该模块进行数据请求
String nameStr=msg.getData().getString("sensorName"); //获取传感器名称
addr=msg.getData().getByteArray("sensorAddr");//获取传感器地址
```

```
if(addr! =null && nameStr! =null)
{
    mStartTime=new Date();//记录服务器开始时间
    Log.d(TAG,"get sensor name : "+nameStr+", addr :"+addr);
    //处理 CO2 和灯光传感器的地址
    if(nameStr.equals(Sensor.CO2_SENSOR)
        ||nameStr.equals(Sensor.LIGHT_SENSOR)){
        //根据传感器名称设置传感器地址
        Sensor sensor=Sensor.getSensorByName(gSensorList, nameStr);
        if(sensor ! = null){
            sensor.setAddr(addr);
        }
    }
    //处理空气温湿度传感器的地址,它们共用一个地址
    if(nameStr.equals(Sensor.AIR_TMPER_SENSOR)){
        Sensor sensor=Sensor.getSensorByName(gSensorList, nameStr);
        if(sensor ! = null){
            sensor.setAddr(addr);
        }
        sensor=Sensor.getSensorByName(gSensorList, Sensor.AIR_HUMID_SENSOR);
        if(sensor ! = null){
            sensor.setAddr(addr);
        }
    }
    //处理土壤温湿度传感器的地址,它们共用一个地址
    if(nameStr.equals(Sensor.SOIL_TMPER_SENSOR)){
        Sensor sensor=Sensor.getSensorByName(gSensorList, nameStr);
        if(sensor ! = null){
            sensor.setAddr(addr);
        }
        sensor=Sensor.getSensorByName(gSensorList, Sensor.SOIL_HUMID_SENSOR);
        if(sensor ! = null){
            sensor.setAddr(addr);
        }
    }
//处理控制器的地址
if(nameStr.equals(Controller.WATER_PUMP_CONTORL) &&gControllerList.get(0).getAddr()==null){
    //所有的控制器都共用一个地址
    for(Controller controller : gControllerList){
        controller.setAddr(addr);
        //获得控制器地址时,先关闭控制器
```

```
            adapter.controlAct(controller.getAddr(), controller.getSubDev(), 0x30);
            controller.setStatus(0);
            threadWait();//延时,预留时间给下位机处理
        }
    }
```

开发工程师之间两两交叉代码评审，并根据评审纪要重构代码。

所有团队成员召开评审会，并填写表 4-4-3。

表 4-4-3　　评审建议表

序号	建议描述	是否接纳	原因
1		□接纳　□部分接纳　□未接纳	
2		□接纳　□部分接纳　□未接纳	
3		□接纳　□部分接纳　□未接纳	
4		□接纳　□部分接纳　□未接纳	
5		□接纳　□部分接纳　□未接纳	
6		□接纳　□部分接纳　□未接纳	
7		□接纳　□部分接纳　□未接纳	
8		□接纳　□部分接纳　□未接纳	
9		□接纳　□部分接纳　□未接纳	
10		□接纳　□部分接纳　□未接纳	

所有团队成员召开复盘会，并填写表 4-4-4。

表 4-4-4　　复盘会总结表

回顾目标	评估结果
当初的目的是什么（期望的结果）	Highlights (与原来目标比）
要达成的目标 & 里程碑	Lowlights (与原来目标比）
总结规律	分析原因
经验 & 规律（不要轻易下结论）	成功关键因素（主观/客观）
行动计划	失败根本原因（主观/客观）
新举措：	
叫停：	
继续：	

同学们，根据塔克曼阶梯理论，你们的团队目前处于哪个阶段呢？如果还没有进入成熟阶段，那就让我们一起努力吧，在接下来的任务中，更加注重相互配合和协作，及时发现并改正团队配合过程中出现的问题，早日让我们的团队进入成熟阶段！

4.4.5 任务扩展(Extend)

课后练习：使用 Timer 和 TimerTask 类实现定时操作。

子任务 5 实现用户登录

• 子任务目标：

➢ 应用用户的注册登录功能

• 课时分配：

4 课时

4.5.1 任务构思(Conceive)

在智能农业实战项目中，用户需要先在客户端的 App 进行注册，注册成功后，信息保存在服务端的 App。当登录系统时，在客户端的 App 输入注册时的用户名和密码，即可成功登录系统，从而进行后续操作。

4.5.2 任务设计(Design)

在 Android 系统中，数据保存的方式有：SharedPreferencs、SQLite 数据库、文件和网络。对于简单的配置信息，使用 SharedPreferences 即可。

4.5.3 任务实现(Implement)

智能农业实战项目的该子任务包含的角色为开发工程师。角色扮演的过程如表 4-5-1所示。

表 4-5-1　　角色扮演过程

步骤	角色	工作	输出
1	开发工程师	编码实现子任务的业务功能	实现子任务业务功能的 Android 源代码
2	开发工程师	两两交叉代码评审	代码评审纪要
3	开发工程师	根据代码评审纪要重构代码	重构后的 Android 源代码

4.5.4 任务运作(Operate)

所有团队成员召开计划会议，并将会议纪要记录在表 4-5-2 中。

表 4-5-2　　会议纪要记录表

5W2H	结论
WHAT	
HOW	
WHY	
WHEN	
WHERE	
WHO	
HOW MUCH	

在计划会议结束之前，所有团队成员把子任务进一步细分为更小的任务，把这些任务写在便签上，并把便签贴在看板上的“ToDo”列。

所有团队成员每天早上召开 15 分钟的站立会议，每人汇报三个问题，并更新看板上的“ToDo”、“Doing”和“Done”三列。

所有团队成员相互协作，编码实现子任务的业务功能，详细的示例代码请见 Android 项目：RegisterLogin，参考代码如下：

(1)添加布局文件：activity_login. xml

```
<LinearLayout xmlns:android= "http://schemas.android.com/apk/res/android"
    android:layout_width= "fill_parent"
    android:layout_height= "fill_parent"
    android:orientation= "vertical">

    <TextView
    android:id= "@+id/TextView01"
        android:layout_width= "wrap_content"
        android:layout_height= "wrap_content"
        android:text= "@string/user_login"
        android:textColor= "#ff0000"
        android:textSize= "24sp">
    </TextView>

    <TextView
        android:id= "@+id/TextView02"
        android:layout_width= "wrap_content"
        android:layout_height= "wrap_content"
        android:text= "@string/user_name"
        android:textSize= "18sp">
    </TextView>
```

```
<EditText
    android:id="@+id/EditText01"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:hint="@string/user"
    android:text="">
</EditText>

<TextView
    android:id="@+id/TextView03"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="@string/pwd_login"
    android:textSize="18sp">
</TextView>

<EditText
    android:id="@+id/EditText02"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:hint="@string/pwd"
    android:password="true"
    android:text="">
</EditText>

<CheckBox
    android:id="@+id/ckb_save_password"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="@string/recode_pwd">
</CheckBox>

<LinearLayout
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:layout_marginTop="20dip"
    android:gravity="center"
    android:orientation="horizontal">

    <Button
        android:id="@+id/Button01"
```

```
            android:layout_width="120dip"
            android:layout_height="40dip"
            android:layout_gravity="right|center_vertical"
            android:text="@string/login"
            android:textSize="20sp">
        </Button>

        <Button
            android:id="@+id/Button02"
            android:layout_width="120dip"
            android:layout_height="40dip"
            android:layout_gravity="left|center_vertical"
            android:text="@string/register"
            android:textSize="20sp">
        </Button>
    </LinearLayout>

</LinearLayout>
```

(2)添加布局文件:activity_main.xml

```
<? xml version="1.0"encoding="UTF-8"? >
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical">

    <TextView
    android:id="@+id/TextView01"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="@string/user_login"
    android:textColor="#ff0000"
    android:textSize="24sp">
    </TextView>

    <TextView
        android:id="@+id/TextView02"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/user_name"
        android:textSize="18sp">
    </TextView>
```

```
<EditText
    android:id="@+id/EditText01"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="">
</EditText>

<TextView
    android:id="@+id/TextView03"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="@string/pwd_login"
    android:textSize="18sp">
</TextView>

<EditText
    android:id="@+id/EditText02"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="">
</EditText>

<CheckBox
    android:id="@+id/ckb_save_password"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="@string/recode_pwd">
</CheckBox>

<LinearLayout
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:layout_marginTop="20dip"
    android:gravity="center"
    android:orientation="horizontal">

    <Button
        android:id="@+id/Button01"
        android:layout_width="120dip"
        android:layout_height="40dip"
```

```
            android:layout_gravity="right|center_vertical"
            android:text="@string/login"
            android:textSize="20sp">
        </Button>

        <Button
            android:id="@+id/Button02"
            android:layout_width="120dip"
            android:layout_height="40dip"
            android:layout_gravity="left|center_vertical"
            android:text="@string/cancel"
            android:textSize="20sp">
        </Button>
    </LinearLayout>

</LinearLayout>
```

(3)添加 Java 类:LoginActivity.java

```
public class LoginActivity extends Activity {

    String pass_user = "";
    String saved_user, saved_pass;
    EditTextet 01, et02;
    Button btn01, btn02;
    CheckBox ckb_save;
    String input_user, input_pass;
    public static final String SHAREDPREFERENCES_NAME = "SaveUsers";
    public static int MODE = Context.MODE_WORLD_READABLE + Context.MODE_WORLD_WRITEABLE;
    SharedPreferencessharedPreferences;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_login);
        Intent intent = getIntent();

        Bundle bundle = intent.getExtras();
        if (bundle ! = null) {
            pass_user = bundle.getString("user_name");
        }

    sharedPreferences = getSharedPreferences( SHAREDPREFERENCES_NAME , MODE );
```

```
saved_pass = sharedPreferences.getString("password", "");
saved_user = sharedPreferences.getString("user_name", "");
et01 = (EditText) findViewById(R.id.EditText01);
et02 = (EditText) findViewById(R.id.EditText02);
btn01 = (Button) findViewById(R.id.Button01);
btn02 = (Button) findViewById(R.id.Button02);
ckb_save = (CheckBox) findViewById(R.id.ckb_save_password);
et01.setText(pass_user);
boolean t = sharedPreferences.getBoolean("save", false);
if (sharedPreferences.getBoolean("save", false)) {
    et01.setText(saved_user);
    et02.setText(saved_pass);
    ckb_save.setChecked(true);
}
btn02.setOnClickListener(new OnClickListener() {

    @Override
    public void onClick(View arg0) {
        Intent intent = new Intent(LoginActivity.this, MainActivity.class);
        startActivity(intent);
    }

});
btn01.setOnClickListener(new OnClickListener() {

    @Override
    public void onClick(View arg0) {
        input_user = et01.getText().toString().trim();
        input_pass = et02.getText().toString().trim();
        if (input_user.equals("")) {
            Toast.makeText(getApplicationContext(), "用户名不能为空", Toast.
        LENGTH_SHORT).show();
        }else {
            if (input_pass.equals("")) {
                Toast.makeText(getApplicationContext(), "密码不能为空", Toast.
            LENGTH_SHORT)
                        .show();
            }else {
                if (input_user.equals(saved_user) && input_pass.equals(saved_
            pass)) {
                    Toast.makeText(getApplicationContext(), "登陆成功",
```

```
                    Toast. LENGTH_SHORT )
                            .show();

                        if (ckb_save.isChecked()) {
                            SharedPreferences.Editor editor = sharedPreferences.edit
                        ();
                            editor.putBoolean("save", true);
                            editor.putString("user_name", input_user);
                            editor.putString("password", input_pass);
                        }else {
                            SharedPreferences.Editor editor = sharedPreferences.edit
                        ();
                            editor.putBoolean("save", false);
                        }
                    }else {
                        Toast.makeText(getApplicationContext(), "用户名密码错误"
                    + saved_pass,
                            Toast.LENGTH_SHORT).show();
                    }
                }
            }
        }

    });
  }
}
```

确保以上每一步都编译通过：在 IDE 集成开发环境中，没有出现红叉的错误提示。应用程序的运行效果如图 4-5-1 所示。点击注册按钮，如图 4-5-2 所示。

图 4-5-1　应用程序运行效果图

图 4-5-2　注册界面

输入符合要求的注册信息，再次点击“注册”按钮，如图 4-5-3 所示。

在主界面输入注册信息，点击“登录”按钮，如图 4-5-4 所示。

图 4-5-3　注册成功界面

图 4-5-4　登录成功界面

智能农业实战项目源代码中的核心代码片段如下：

```
protected String soapPorcess(String param)
{
    String username="";
    String password="";
    //从完整的 soap 协议内容中解析出 soap body 内容
    Element soapBodyEle=Util.getSoapBodyElement(param);
    if(soapBodyEle ! = null)
    {
        //解析用户名
        username=Util.getChildElementValueStr(soapBodyEle, "username");
        //解析用户密码
        password=Util.getChildElementValueStr(soapBodyEle, "password");
    }
    String retValue="failed";
    //通过用户名和密码,验证某个用户是否存在数据库中
    if (DatabaseUtil.query(username, password,context)) {
        retValue="ok";
    }
    //返回结果
    return Util.getSoapXml(Util.getXmlElementStr("result", retValue));
}
JSONObject jsonRequest=new JSONObject(param);
//处理注册请求
```

```
String username="";
String password="";
String email="";
//解析用户名
if(jsonRequest.has("username")){
username=jsonRequest.getString("username");
}
//解析用户密码
if(jsonRequest.has("password")){
    password=jsonRequest.getString("password");
}
//解析邮箱地址
if(jsonRequest.has("email")){
    email=jsonRequest.getString("email");
}
//向数据库中插入一条用户记录
if (DatabaseUtil.insert(username, password,email, context) ! = -1) {
    jsonResponse.put("result", "ok");
} else {
    jsonResponse.put("result", "failed");
}
//返回结果
return jsonResponse.toString();
```

开发工程师之间两两交叉代码评审，并根据评审纪要重构代码。

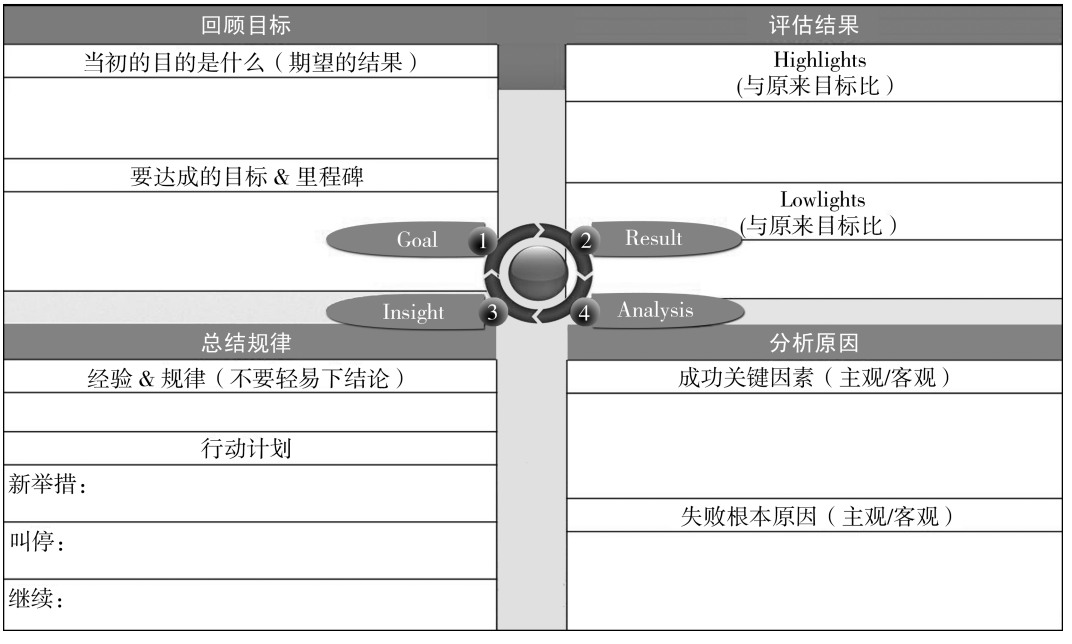
所有团队成员召开评审会，并填写表 4-5-3。

表 4-5-3　　评审建议表

序号	建议描述	是否接纳	原因
1		□接纳　□部分接纳　□未接纳	
2		□接纳　□部分接纳　□未接纳	
3		□接纳　□部分接纳　□未接纳	
4		□接纳　□部分接纳　□未接纳	
5		□接纳　□部分接纳　□未接纳	
6		□接纳　□部分接纳　□未接纳	
7		□接纳　□部分接纳　□未接纳	
8		□接纳　□部分接纳　□未接纳	
9		□接纳　□部分接纳　□未接纳	
10		□接纳　□部分接纳　□未接纳	

所有团队成员召开复盘会，并填写表 4-5-4。

表 4-5-4　　复盘会总结表



同学们，根据塔克曼阶梯理论，你们的团队目前处于哪个阶段呢？如果还没有进入成熟阶段，那就让我们一起努力吧，在接下来的任务中，更加注重相互配合和协作，及时发现并改正团队配合过程中出现的问题，早日让我们的团队进入成熟阶段！

4.5.5　任务扩展(Extend)

课后练习：使用 SQLite 数据库保存用户的注册信息。

子任务 6　实现用户权限

- **子任务目标：**
  - 应用用户、角色和权限的关联操作
- **课时分配：**
  - 4 课时

4.6.1　任务构思(Conceive)

在智能农业实战项目中，当用户是 admin 时，拥有管理员权限，可以进行手动控制和设置各个传感器的阀值。其他用户只能获取和展示传感器数据，不能进行手动控制和设置。

4.6.2　任务设计(Design)

在手动控制界面和系统设置界面里，当用户点击选择项时，先判断用户名是否为“admin”，如果不是则提示用户没有操作权限。

4.6.3　任务实现(Implement)

智能农业实战项目的该子任务包含的角色为开发工程师。角色扮演的过程如表4-6-1所示。

表 4-6-1　　角色扮演过程

步骤	角色	工作	输出
1	开发工程师	编码实现子任务的业务功能	实现子任务业务功能的 Android 源代码
2	开发工程师	两两交叉代码评审	代码评审纪要
3	开发工程师	根据代码评审纪要重构代码	重构后的 Android 源代码

4.6.4　任务运作(Operate)

所有团队成员召开计划会议，并将会议纪要记录在表4-6-2中。

表 4-6-2　　会议纪要记录表

5W2H	结论
WHAT	
HOW	
WHY	
WHEN	
WHERE	
WHO	
HOW MUCH	

在计划会议结束之前，所有团队成员把子任务进一步细分为更小的任务，把这些任务写在便签上，并把便签贴在看板上的“ToDo”列。

所有团队成员每天早上召开15分钟的站立会议，每人汇报三个问题，并更新看板上的“ToDo”、“Doing”和“Done”三列。

所有团队成员相互协作，编码实现子任务的业务功能，详细的代码请见智能农业实战项目完整源代码，主要代码如下：

```
if(mUsername.equals("admin")){
    //调用设置土壤温湿度阀值的对话框
    SettingSoilDialog dialog=new SettingSoilDialog(getActivity(), mApp);
    dialog.show();
} else {
```

```
        showNoAuthAlertDialog();
    }
    //提示用户缺少控制权限
    private void showNoAuthAlertDialog(){
        showAlertDialog(getString(R.string.prompt),getString(R.string.no_setting_authority_exp));
    }
```

开发工程师之间两两交叉代码评审，并根据评审纪要重构代码。

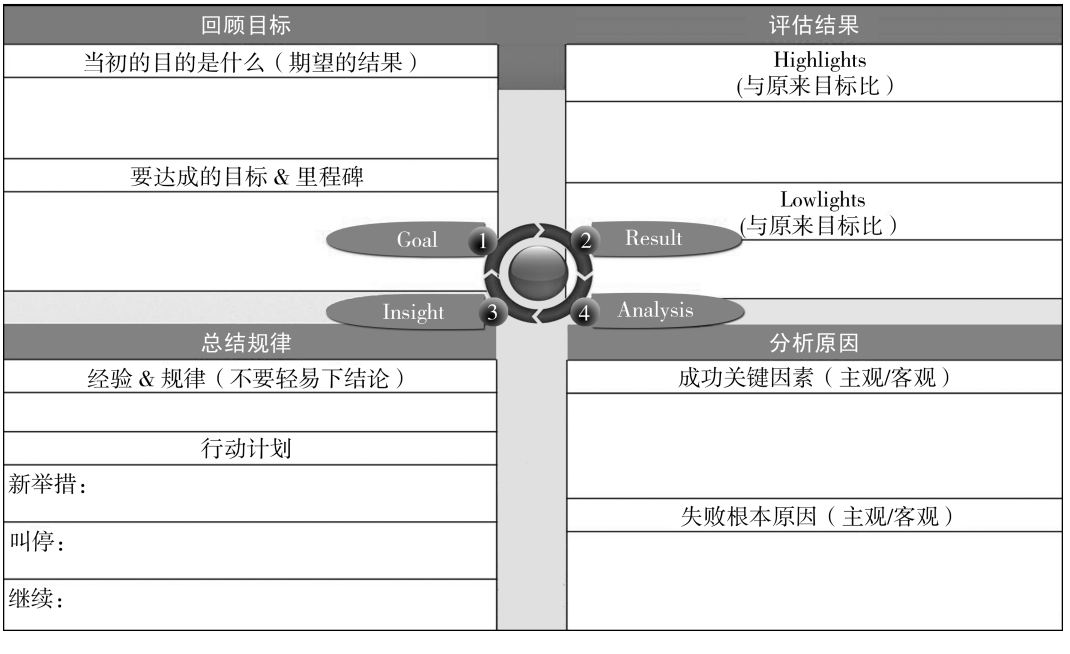
所有团队成员召开评审会，并填写表 4-6-3。

表 4-6-3　　评审建议表

序号	建议描述	是否接纳	原因
1		□接纳　□部分接纳　□未接纳	
2		□接纳　□部分接纳　□未接纳	
3		□接纳　□部分接纳　□未接纳	
4		□接纳　□部分接纳　□未接纳	
5		□接纳　□部分接纳　□未接纳	
6		□接纳　□部分接纳　□未接纳	
7		□接纳　□部分接纳　□未接纳	
8		□接纳　□部分接纳　□未接纳	
9		□接纳　□部分接纳　□未接纳	
10		□接纳　□部分接纳　□未接纳	

所有团队成员召开复盘会，并填写表 4-6-4。

表 4-6-4　　复盘会总结表



同学们，根据塔克曼阶梯理论，你们的团队目前处于哪个阶段呢？如果还没有进入成熟阶段，那就让我们一起努力吧，在接下来的任务中，更加注重相互配合和协作，及时发现并改正团队配合过程中出现的问题，早日让我们的团队进入成熟阶段！

4.6.5　任务扩展(Extend)

课后练习：在系统中设置多个角色，并分别赋予不同的操作权限。

子任务7　实现用户管理

- **子任务目标：**
  - 应用用户注册、登录和找回密码等功能模块
- **课时分配：**
  - 4课时

4.7.1　任务构思(Conceive)

在智能农业实战项目中，服务器提供了用户注册、登录和找回密码等接口，用户数据存储在后台数据库。

4.7.2　任务设计(Design)

服务端采用Android系统自带的SQLite数据库类实现了用户数据的存储，用户表包括用户名、密码和邮箱地址。

4.7.3　任务实现(Implement)

智能农业实战项目的该子任务包含的角色为开发工程师。角色扮演的过程如表4-7-1所示。

表4-7-1　　角色扮演过程

步骤	角色	工作	输出
1	开发工程师	编码实现子任务的业务功能	实现子任务业务功能的Android源代码
2	开发工程师	两两交叉代码评审	代码评审纪要
3	开发工程师	根据代码评审纪要重构代码	重构后的Android源代码

4.7.4　任务运作(Operate)

所有团队成员召开计划会议，并将会议纪要记录在表4-7-2中。

表 4-7-2　　会议纪要记录表

5W2H	结论
WHAT	
HOW	
WHY	
WHEN	
WHERE	
WHO	
HOW MUCH	

在计划会议结束之前，所有团队成员把子任务进一步细分为更小的任务，把这些任务写在便签上，并把便签贴在看板上的“ToDo”列。

所有团队成员每天早上召开 15 分钟的站立会议，每人汇报三个问题，并更新看板上的“ToDo”、“Doing”和“Done”三列。

所有团队成员相互协作，编码实现子任务的业务功能，详细的代码请见智能农业实战项目完整源代码，核心代码片段如下：

```
public static long insert(String username, String password,String email, Context context) {
    // 创建 ContentValues 对象
    ContentValues values=new ContentValues();
    // 向该对象中插入键值对，其中键是列名，值是希望插入到这一列的值，值必须和数据库当中的数据类型一致
    values.put("username", username);
    values.put("userpwd", password);
    values.put("email", email);
    SQLiteDatabase sqliteDatabase=getDatabase(context);
    // 调用 insert 方法，就可以将数据插入到数据库当中
    // 第一个参数:表名称
    // 第二个参数:SQl 不允许一个空列，如果 ContentValues 是空的，那么这一列被明确指明为 NULL 值
    // 第三个参数:ContentValues 对象
    return sqliteDatabase.insert("user", null, values);
}
public static boolean query(String username, String pwd, Context context) {
    SQLiteDatabase sqliteDatabase=getDatabase(context);
    Cursor cursor=sqliteDatabase.query("user", new String[] { "username",
        "userpwd" }, "username=? and userpwd=?", new String[] {
        username, pwd }, null, null, null);
    if (cursor.moveToNext())
        return true;
    return false;
}
```

开发工程师之间两两交叉代码评审，并根据评审纪要重构代码。

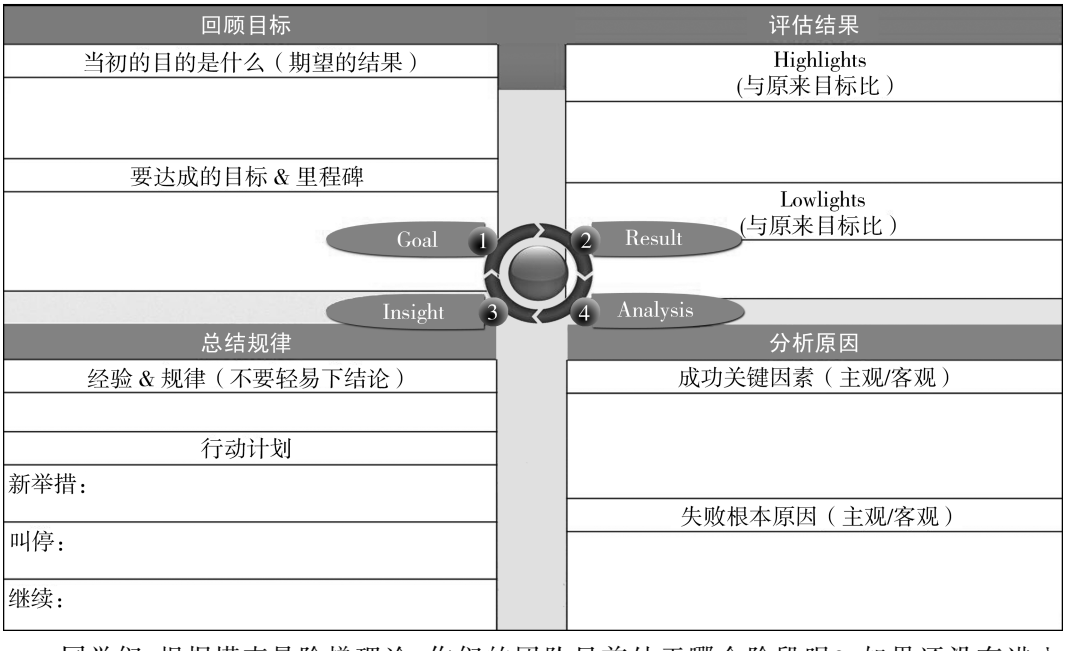
所有团队成员召开评审会，并填写表 4-7-3。

表 4-7-3　　评审建议表

序号	建议描述	是否接纳	原因
1		□接纳　□部分接纳　□未接纳	
2		□接纳　□部分接纳　□未接纳	
3		□接纳　□部分接纳　□未接纳	
4		□接纳　□部分接纳　□未接纳	
5		□接纳　□部分接纳　□未接纳	
6		□接纳　□部分接纳　□未接纳	
7		□接纳　□部分接纳　□未接纳	
8		□接纳　□部分接纳　□未接纳	
9		□接纳　□部分接纳　□未接纳	
10		□接纳　□部分接纳　□未接纳	

所有团队成员召开复盘会，并填写表 4-7-4。

表 4-7-4　　复盘会总结表



同学们，根据塔克曼阶梯理论，你们的团队目前处于哪个阶段呢？如果还没有进入成熟阶段，那就让我们一起努力吧，在接下来的任务中，更加注重相互配合和协作，及时发现并改正团队配合过程中出现的问题，早日让我们的团队进入成熟阶段！

4.7.5 任务扩展(Extend)

课后练习:根据用户的某个信息在数据库中查找到对应的用户。

子任务 8 实现找回密码功能

- **子任务目标:**

  - 应用发送电子邮件帮助用户找回密码

- **课时分配:**

  - 4 课时

4.8.1 任务构思(Conceive)

在智能农业实战项目中,登录界面包括的信息有:用户名和忘记密码按钮。找回密码界面需要对输入的信息进行过滤,对不满足要求的输入在客户端进行处理。对满足要求的数据,进行封装后,发往服务端进行鉴权,如果确实存在该用户,则发密码到该用户的邮箱。

4.8.2 任务设计(Design)

如果用户名和密码不符合要求,则提示用户,采用 HTTP 或 SOCKET 的通信方式向服务器发送请求。如果密码成功发送到用户邮箱,则服务器返回报文中会携带该邮箱地址,并给出提示。

4.8.3 任务实现(Implement)

智能农业实战项目的该子任务包含的角色为开发工程师。角色扮演的过程如表 4-8-1所示。

表 4-8-1　　角色扮演过程

步骤	角色	工作	输出
1	开发工程师	编码实现子任务的业务功能	实现子任务业务功能的 Android 源代码
2	开发工程师	两两交叉代码评审	代码评审纪要
3	开发工程师	根据代码评审纪要重构代码	重构后的 Android 源代码

4.8.4 任务运作(Operate)

所有团队成员召开计划会议,并将会议纪要记录在表 4-8-2 中。

表 4-8-2　　会议纪要记录表

5W2H	结论
WHAT	
HOW	
WHY	
WHEN	
WHERE	
WHO	
HOW MUCH	

在计划会议结束之前，所有团队成员把子任务进一步细分为更小的任务，把这些任务写在便签上，并把便签贴在看板上的“ToDo”列。

所有团队成员每天早上召开 15 分钟的站立会议，每人汇报三个问题，并更新看板上的“ToDo”、“Doing”和“Done”三列。

所有团队成员相互协作，编码实现子任务的业务功能，详细的代码请见智能农业实战项目完整源代码，核心代码片段如下：

```
//从完整的 soap 协议内容中解析出 soap body 内容
Element soapBodyEle=Util.getSoapBodyElement(param);
//解析用户名
if(soapBodyEle ! = null){
    username=Util.getChildElementValueStr(soapBodyEle, "username");
}
//通过用户名查找该用户的注册邮箱
String email=DatabaseUtil.queryUserEmail(username, context);
//通过用户名查找该用户的密码
String password=DatabaseUtil.queryUserPassword(username, context);
//如果密码和注册邮箱都找到，则将用户的用户名和密码发送至用户邮箱
if(email! =null && ! email.equals("")){
    //返回结果和用户邮箱地址
    retSoapBody=Util.getXmlElementStr("result", "ok");
    retSoapBody + = Util.getXmlElementStr("email", email);
    sendPasswordToEmail(username,email,password);
} else {
    retSoapBody=Util.getXmlElementStr("result", "failed");
}
//返回结果
return Util.getSoapXml(retSoapBody);
if(jsonRespObj.has("result"))
{
    String result=jsonRespObj.getString("result");
    result=result.toLowerCase();
    //判断请求是否成功
    if(result.equals("ok")){
        isSuccess=true;
    } else {
```

```
            isSuccess=false;
    }
    }
    //从回应中解析出用户邮箱,便于提示给用户
    if(jsonRespObj.has("email"))
    {
        email=jsonRespObj.getString("email");
    }
```

开发工程师之间两两交叉代码评审,并根据评审纪要重构代码。

所有团队成员召开评审会,并填写表 4-8-3。

表 4-8-3　　评审建议表

序号	建议描述	是否接纳	原因
1		□接纳 □部分接纳 □未接纳	
2		□接纳 □部分接纳 □未接纳	
3		□接纳 □部分接纳 □未接纳	
4		□接纳 □部分接纳 □未接纳	
5		□接纳 □部分接纳 □未接纳	
6		□接纳 □部分接纳 □未接纳	
7		□接纳 □部分接纳 □未接纳	
8		□接纳 □部分接纳 □未接纳	
9		□接纳 □部分接纳 □未接纳	
10		□接纳 □部分接纳 □未接纳	

所有团队成员召开复盘会,并填写表 4-8-4。

表 4-8-4　　复盘会总结表

回顾目标	评估结果
当初的目的是什么（期望的结果）	Highlights (与原来目标比）
要达成的目标 & 里程碑	Lowlights (与原来目标比）
Goal 1　2 Result	
Insight 3　4 Analysis	
总结规律	**分析原因**
经验 & 规律（不要轻易下结论）	成功关键因素（主观/客观）
行动计划	失败根本原因（主观/客观）
新举措：	
叫停：	
继续：	

同学们，根据塔克曼阶梯理论，你们的团队目前处于哪个阶段呢？如果还没有进入成熟阶段，那就让我们一起努力吧，在接下来的任务中，更加注重相互配合和协作，及时发现并改正团队配合过程中出现的问题，早日让我们的团队进入成熟阶段！

4.8.5　任务扩展(Extend)

课后练习：实现邮件服务器端和客户端的配置。

子任务9　实现日志功能

- **子任务目标：**
  - 应用服务端应用程序的日志记录功能
- **课时分配：**
  - 4课时

4.9.1　任务构思(Conceive)

在智能农业实战项目中，服务端实现日志记录功能，用于记录客户端的信息、操作类型及操作时间等相关信息。服务端实现日志的浏览功能（在服务端主界面单击浏览日志按钮，打开日志浏览界面）。日志内容包括：客户IP地址、用户名、操作类型和操作对象。

4.9.2　任务设计(Design)

通过对文本文件的读写即可实现简单的日志系统，设计文件中每行表示一条日志，每条日志中的字段以“[]”扩起来。比如：

[2014-06-05 12:12:12][192.168.1.1][admin][login][admin][User login success]，

日志文件路径：

public final static String LOG_FILE_PATH="/sdcard/lenovo/agriculture_server.txt"。

4.9.3　任务实现(Implement)

智能农业实战项目的该子任务包含的角色为开发工程师。角色扮演的过程如表4-9-1所示。

表4-9-1　　角色扮演过程

步骤	角色	工作	输出
1	开发工程师	编码实现子任务的业务功能	实现子任务业务功能的Android源代码
2	开发工程师	两两交叉代码评审	代码评审纪要
3	开发工程师	根据代码评审纪要重构代码	重构后的Android源代码

4.9.4 任务运作(Operate)

所有团队成员召开计划会议,并将会议纪要记录在表 4-9-2 中。

表 4-9-2　　会议纪要记录表

5W2H	结论
WHAT	
HOW	
WHY	
WHEN	
WHERE	
WHO	
HOW MUCH	

在计划会议结束之前,所有团队成员把子任务进一步细分为更小的任务,把这些任务写在便签上,并把便签贴在看板上的“ToDo”列。

所有团队成员每天早上召开 15 分钟的站立会议,每人汇报三个问题,并更新看板上的“ToDo”、“Doing”和“Done”三列。

所有团队成员相互协作,编码实现子任务的业务功能,详细的代码请见智能农业实战项目完整源代码,核心代码片段如下:

```
//创建文件输入流
fileIs=new FileInputStream(AppConfig.LOG_FILE_PATH);
//创建 reader buffer
BufferedReader buf=new BufferedReader(new InputStreamReader(fileIs));
String readString=null;
//一行一行读取日志文本
while((readString=buf.readLine())! =null)
{
    //以“][”为分隔符,取出日志各字段内容
    String logInfos[]=readString.split("\\]\\[");
    LogBean logBean=new LogBean();
    logBean.setDate(logInfos[0].replace("[", ""));//日志日期时间
    logBean.setClientIP(logInfos[1]);//客户端 IP 地址
    logBean.setUsername(logInfos[3]);//用户名
    logBean.setOperationType(logInfos[4]);//操作类型
    logBean.setOperationObj(logInfos[5]);//操作对象
    logBean.setLogMsg(logInfos[6].replace("]", ""));//日志内容
    dataLists.add(logBean);
}
```

开发工程师之间两两交叉代码评审,并根据评审纪要重构代码。

所有团队成员召开评审会，并填写表 4-9-3。

表 4-9-3　　评审建议表

序号	建议描述	是否接纳	原因
1		□接纳　□部分接纳　□未接纳	
2		□接纳　□部分接纳　□未接纳	
3		□接纳　□部分接纳　□未接纳	
4		□接纳　□部分接纳　□未接纳	
5		□接纳　□部分接纳　□未接纳	
6		□接纳　□部分接纳　□未接纳	
7		□接纳　□部分接纳　□未接纳	
8		□接纳　□部分接纳　□未接纳	
9		□接纳　□部分接纳　□未接纳	
10		□接纳　□部分接纳　□未接纳	

所有团队成员召开复盘会，并填写表 4-9-4。

表 4-9-4　　复盘会总结表

回顾目标	评估结果
当初的目的是什么（期望的结果）	Highlights (与原来目标比）
要达成的目标 & 里程碑	Lowlights (与原来目标比）
Goal 1　2 Result	
Insight 3　4 Analysis	
总结规律	**分析原因**
经验 & 规律（不要轻易下结论）	成功关键因素（主观/客观）
行动计划	失败根本原因（主观/客观）
新举措：	
叫停：	
继续：	

同学们，根据塔克曼阶梯理论，你们的团队目前处于哪个阶段呢？如果还没有进入成熟阶段，那就让我们一起努力吧，在接下来的任务中，更加注重相互配合和协作，及时发现并改正团队配合过程中出现的问题，早日让我们的团队进入成熟阶段！

4.9.5 任务扩展(Extend)

课后练习:在程序中设置一个 Bug,根据日志文件中的信息找到 Bug 产生的原因。

子任务 10 实现 Splash

• 子任务目标:

➢ 在应用程序启动时,应用 Splash 的效果

• 课时分配:

➢ 4 课时

4.10.1 任务构思(Conceive)

在智能农业实战项目中,当安装应用程序后用户第一次打开应用程序时,为了给用户一些应用程序的提示信息并增强用户体验,可以添加 Splash 的效果。在 Android 系统中,实现 Splash 可以通过 ViewSwitcher,在运行 Splash 的期间加载数据或展示应用程序的功能和特色。

4.10.2 任务设计(Design)

在 Android 系统中,用 ViewSwitcher 实现 Splash 主要通过下面 3 个步骤:

(1)将 Activity 的 RootView 设置为 ViewSwitcher,把一个布局(如 ImageView)作为第一个子 View。

(2)将 Activity 的主页面布局作为第二个子 View。

(3)应用程序启动时,先显示作为 Splash 的第一个子 View,展示完之后再显示主页面布局。

4.10.3 任务实现(Implement)

智能农业实战项目的该子任务包含的角色为开发工程师。角色扮演的过程如表 4-10-1所示。

表 4-10-1 角色扮演过程

步骤	角色	工作	输出
1	开发工程师	编码实现子任务的业务功能	实现子任务业务功能的 Android 源代码
2	开发工程师	两两交叉代码评审	代码评审纪要
3	开发工程师	根据代码评审纪要重构代码	重构后的 Android 源代码

4.10.4　任务运作(Operate)

所有团队成员召开计划会议，并将会议纪要记录在表 4-10-2 中。

表 4-10-2　　会议纪要记录表

5W2H	结论
WHAT	
HOW	
WHY	
WHEN	
WHERE	
WHO	
HOW MUCH	

在计划会议结束之前，所有团队成员把子任务进一步细分为更小的任务，把这些任务写在便签上，并把便签贴在看板上的“ToDo”列。

所有团队成员每天早上召开 15 分钟的站立会议，每人汇报三个问题，并更新看板上的“ToDo”、“Doing”和“Done”三列。

所有团队成员相互协作，编码实现子任务的业务功能，详细的示例代码请见 Android 项目 Splash，参考代码如下：

(1)创建布局文件：activity_main. xml

```
<? xml version= "1.0"encoding= "utf-8"? >
<FrameLayout xmlns:android= "http://schemas.android.com/apk/res/android"
    android:layout_width= "fill_parent"
    android:layout_height= "fill_parent"
    android:orientation= "vertical">

    <LinearLayout
        android:id= "@+id/linearLayout01"
        android:layout_width= "fill_parent"
        android:layout_height= "wrap_content"
        android:orientation= "vertical">

        <android.support.v4.view.ViewPager
            android:id= "@+id/guidePages"
            android:layout_width= "fill_parent"
            android:layout_height= "wrap_content"/>
    </LinearLayout>
```

```
    <LinearLayout
        android:id="@+id/linearLayout01"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:orientation="vertical">

        <RelativeLayout
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:orientation="vertical">

            <LinearLayout
                android:id="@+id/viewGroup"
                android:layout_width="fill_parent"
                android:layout_height="wrap_content"
                android:layout_alignParentBottom="true"
                android:layout_marginBottom="60dp"
                android:gravity="center_horizontal"
                android:orientation="horizontal">
            </LinearLayout>
        </RelativeLayout>
    </LinearLayout>

</FrameLayout>
```

(2)创建布局文件:item01.xml

```
<? xml version="1.0"encoding="utf-8"? >
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical">

    <ImageView
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:background="@drawable/bg"/>

</LinearLayout>
```

(3)创建布局文件:item02.xml

```
<? xml version="1.0"encoding="utf-8"? >
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
```

```
    android:layout_height="fill_parent"
    android:orientation="vertical">

    <ImageView
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:background="@drawable/s1"/>

</LinearLayout>
```

(4)创建布局文件:item03.xml

```
<? xml version="1.0"encoding="utf-8"? >
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical">

    <ImageView
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:background="@drawable/s2"/>

</LinearLayout>
```

(5)创建布局文件:item04.xml

```
<? xml version="1.0"encoding="utf-8"? >
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:background="@drawable/s3">

    <Button
    android:id="@+id/NavigateHome"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentRight="true"
        android:layout_alignParentTop="true"
        android:background="#008B00"
        android:text="Start"/>

</RelativeLayout>
```

(6)创建 Java 类:MainActivity.java

```
/* *
* 主页面对应的 Activity
* /
public class MainActivity extends Activity {
    private ViewPager viewPager;
    private ArrayList<View> pageViews;
    private ImageView imageView;
    private ImageView[] imageViews;
    // 包裹滑动图片 LinearLayout
    private ViewGroup main;
    // 包裹小圆点的 LinearLayout
    private ViewGroup group;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        // 设置无标题窗口
        requestWindowFeature(Window.FEATURE_NO_TITLE);
        LayoutInflaterinflater = getLayoutInflater();
        pageViews = new ArrayList<View>();

        pageViews.add(inflater.inflate(R.layout.item01, null));
        pageViews.add(inflater.inflate(R.layout.item02, null));
        pageViews.add(inflater.inflate(R.layout.item03, null));
        // pageViews.add(inflater.inflate(R.layout.item04, null));

        Viewfind = inflater.inflate(R.layout.item04, null);
        Buttonbutton = (Button) find.findViewById(R.id.NavigateHome);
        button.setOnClickListener(new OnClickListener() {

            @Override
            public void onClick(View v) {
                Intentintent = new Intent(MainActivity.this,
                    MainActivity.class);
                startActivity(intent);
                finish();
            }
        });

        pageViews.add(find);
```

```
        imageViews = new ImageView[pageViews.size()];
        main = (ViewGroup) inflater.inflate(R.layout.activity_main , null);
        group = (ViewGroup) main.findViewById(R.id.viewGroup );
        viewPager = (ViewPager) main.findViewById(R.id.guidePages );

        for (int i = 0; i < pageViews.size(); i++) {
            imageView = new ImageView(MainActivity.this);
            imageView.setLayoutParams(new LayoutParams(30, 30));
            imageView.setPadding(20, 0, 20, 0);
            imageViews[i] = imageView;

            if (i == 0) {
                // 默认选中第一张图片
                imageViews[i].setBackgroundResource(R.drawable.point_on );
            }else {
                imageViews[i].setBackgroundResource(R.drawable.point );
            }

            group.addView(imageViews[i]);
        }

        setContentView(main);

        viewPager.setAdapter(new GuidePageAdapter());
        viewPager.setOnPageChangeListener(new GuidePageChangeListener());

    }

    // 指引页面数据适配器
    class GuidePageAdapter extends PagerAdapter {

        @Override
        public int getCount() {
            return pageViews.size();
        }

        @Override
        public boolean isViewFromObject(View arg0, Object arg1) {
            return arg0 == arg1;
        }
```

```
        @Override
        public int getItemPosition(Object object) {
            return super.getItemPosition(object);
        }

        @Override
        public void destroyItem(View arg0, int arg1, Object arg2) {
            ((ViewPager)arg0).removeView(pageViews.get(arg1));
        }

        @Override
        public Object instantiateItem(View arg0, int arg1) {
            ((ViewPager)arg0).addView(pageViews.get(arg1));
            return pageViews.get(arg1);
        }

        @Override
        public void restoreState(Parcelable arg0, ClassLoader arg1) {

        }

        @Override
        public Parcelable saveState() {
            return null;
        }

        @Override
        public void startUpdate(View arg0) {

        }

        @Override
        public void finishUpdate(View arg0) {

        }
    }

    // 指引页面更改事件监听器
    class GuidePageChangeListener implements OnPageChangeListener {
```

```
        @Override
        public void onPageScrollStateChanged(int arg0) {

        }

        @Override
        public void onPageScrolled(int arg0, float arg1, int arg2) {

        }

        @Override
        public void onPageSelected(int arg0) {
            for (int i = 0; i < imageViews.length; i++) {
                imageViews[arg0].setBackgroundResource(R.drawable.point_on);

                if (arg0 != i) {
                    imageViews[i].setBackgroundResource(R.drawable.point);
                }
            }
        }
    }
}
```

确保以上每一步都编译通过：在 IDE 集成开发环境中，没有出现红叉的错误提示。应用程序的运行效果如图 4-10-1、4-10-2 和 4-10-3 所示。

图 4-10-1　应用程序运行效果图 1

图 4-10-2　应用程序运行效果图 2

图 4-10-3　应用程序运行效果图 3

开发工程师之间两两交叉代码评审，并根据评审纪要重构代码。

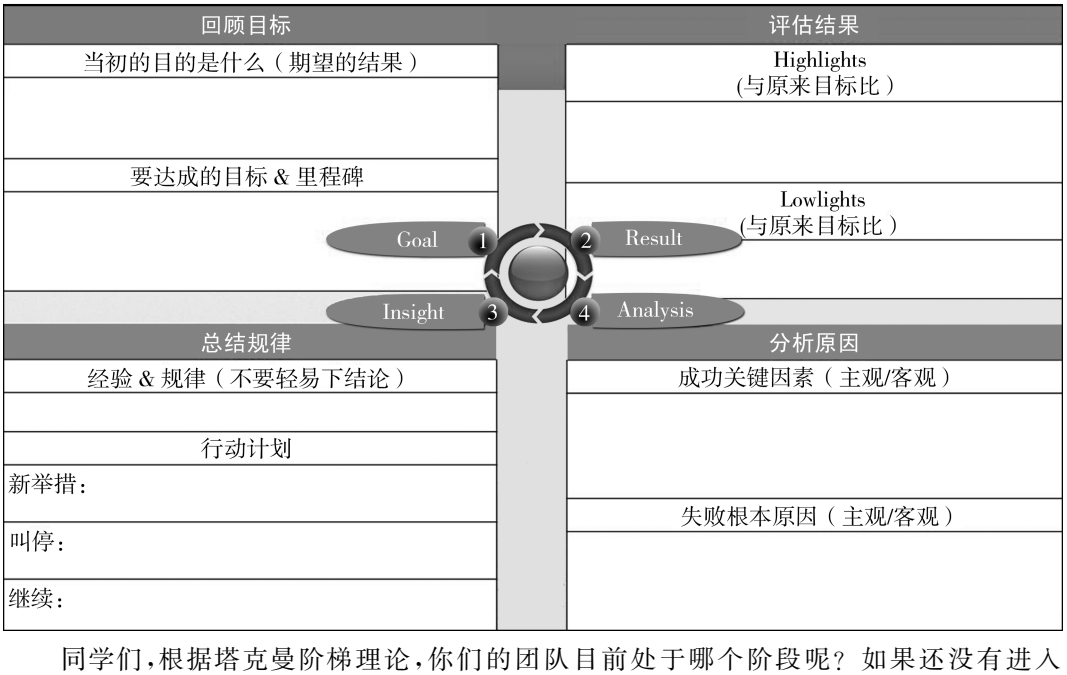
所有团队成员召开评审会，并填写表 4-10-3。

表 4-10-3　　评审建议表

序号	建议描述	是否接纳	原因
1		□接纳　□部分接纳　□未接纳	
2		□接纳　□部分接纳　□未接纳	
3		□接纳　□部分接纳　□未接纳	
4		□接纳　□部分接纳　□未接纳	
5		□接纳　□部分接纳　□未接纳	
6		□接纳　□部分接纳　□未接纳	
7		□接纳　□部分接纳　□未接纳	
8		□接纳　□部分接纳　□未接纳	
9		□接纳　□部分接纳　□未接纳	
10		□接纳　□部分接纳　□未接纳	

所有团队成员召开复盘会，并填写表 4-10-4。

表 4-10-4　　复盘会总结表



同学们，根据塔克曼阶梯理论，你们的团队目前处于哪个阶段呢？如果还没有进入成熟阶段，那就让我们一起努力吧，在接下来的任务中，更加注重相互配合和协作，及时发现并改正团队配合过程中出现的问题，早日让我们的团队进入成熟阶段！

4.10.5　任务扩展(Extend)

课后练习：在 Splash 中添加播放音乐的效果。

子任务 11　实现手动控制

- **子任务目标：**

  ➢ 应用手动控制沙盘上的受控设备

- **课时分配：**

  ➢ 4 课时

4.11.1　任务构思(Conceive)

在智能农业系统中，当系统在手控制模式下，用户可以控制沙盘上的 4 个控制设备：风扇、水泵、LED 灯和蜂鸣器。

4.11.2　任务设计(Design)

先判断用户是否为“admin”，如果不是，则不具备控制权限。当用户点击控制器图表后，切换控制状态，然后将新状态传给服务器进行控制。

4.11.3　任务实现(Implement)

智能农业实战项目的该子任务包含的角色为开发工程师。角色扮演的过程如表 4-11-1所示。

表 4-11-1　　角色扮演过程

步骤	角色	工作	输出
1	开发工程师	编码实现子任务的业务功能	实现子任务业务功能的 Android 源代码
2	开发工程师	两两交叉代码评审	代码评审纪要
3	开发工程师	根据代码评审纪要重构代码	重构后的 Android 源代码

4.11.4　任务运作(Operate)

所有团队成员召开计划会议，并将会议纪要记录在表 4-11-2 中。

表 4-11-2　　会议纪要记录表

5W2H	结论
WHAT	
HOW	
WHY	
WHEN	

（续表）

5W2H	结论
WHERE	
WHO	
HOW MUCH	

在计划会议结束之前，所有团队成员把子任务进一步细分为更小的任务，把这些任务写在便签上，并把便签贴在看板上的“ToDo”列。

所有团队成员每天早上召开 15 分钟的站立会议，每人汇报三个问题，并更新看板上的“ToDo”、“Doing”和“Done”三列。

所有团队成员相互协作，编码实现子任务的业务功能，详细的代码请见智能农业实战项目完整源代码，核心代码片段如下：

```
protected String onGetJasonBody()
{
    JSONObject jsonObj＝new JSONObject();
    try
    {
        //是否要控制风扇
        if(isCtrlBlower){
            jsonObj.put("Blower", blower);
        }
        //是否要控制蜂鸣器
        if(isCtrlBuzzer){
            jsonObj.put("Buzzer", buzzer);
        }
        //是否要控制路灯
        if(isCtrlRoadlamp){
            jsonObj.put("Roadlamp", roadlamp);
        }
        //是否要控制水泵
        if(isCtrlWater){
            jsonObj.put("WaterPump", waterPump);
        }
        return jsonObj.toString();
    }
    catch (JSONException e) {
        e.printStackTrace();
    }
    return super.onGetJasonBody();
}
```

开发工程师之间两两交叉代码评审，并根据评审纪要重构代码。

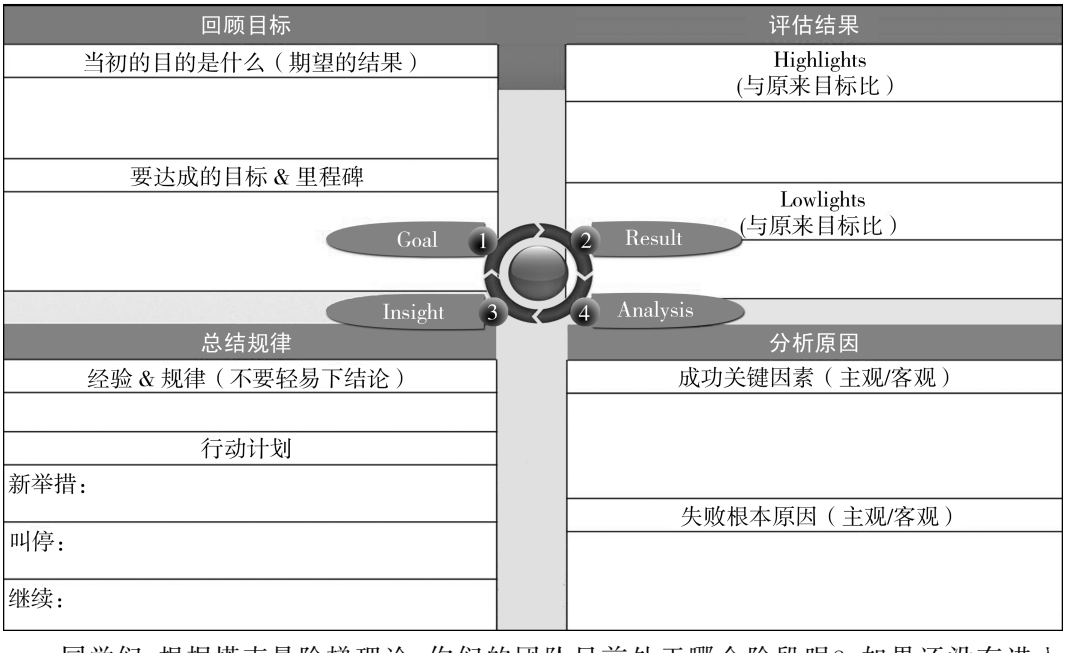
所有团队成员召开评审会，并填写表 4-11-3。

表 4-11-3　　评审建议表

序号	建议描述	是否接纳	原因
1		□接纳　□部分接纳　□未接纳	
2		□接纳　□部分接纳　□未接纳	
3		□接纳　□部分接纳　□未接纳	
4		□接纳　□部分接纳　□未接纳	
5		□接纳　□部分接纳　□未接纳	
6		□接纳　□部分接纳　□未接纳	
7		□接纳　□部分接纳　□未接纳	
8		□接纳　□部分接纳　□未接纳	
9		□接纳　□部分接纳　□未接纳	
10		□接纳　□部分接纳　□未接纳	

所有团队成员召开复盘会，并填写表 4-11-4。

表 4-11-4　　复盘会总结表



同学们，根据塔克曼阶梯理论，你们的团队目前处于哪个阶段呢？如果还没有进入成熟阶段，那就让我们一起努力吧，在接下来的任务中，更加注重相互配合和协作，及时发现并改正团队配合过程中出现的问题，早日让我们的团队进入成熟阶段！

4.11.5　任务扩展(Extend)

课后练习:将手动控制的功能改为自动控制沙盘上的受控设备。

子任务 12　实现实时环境指标界面

- **子任务目标:**

  - 实时环境指标界面的功能实现

- **课时分配:**

  - 4 课时

4.12.1　任务构思(Conceive)

在智能农业实战项目中,需要一个实时环境指标界面,该界面实时展示所有传感器的当前数据值,并能根据每个传感器的阀值判断是否应该告警,如果需要告警则显示红色背景,如果正常则显示绿色背景。

4.12.2　任务设计(Design)

在后台开启一个不断运行的线程,每隔 1 秒钟向服务器端请求一次传感器数据,然后交给 UI 线程进行展示。

4.12.3　任务实现(Implement)

智能农业实战项目的该子任务包含的角色为开发工程师。角色扮演的过程如表 4-12-1所示。

表 4-12-1　　角色扮演过程

步骤	角色	工作	输出
1	开发工程师	编码实现子任务的业务功能	实现子任务业务功能的 Android 源代码
2	开发工程师	两两交叉代码评审	代码评审纪要
3	开发工程师	根据代码评审纪要重构代码	重构后的 Android 源代码

4.12.4　任务运作(Operate)

所有团队成员召开计划会议,并将会议纪要记录在表 4-12-2 中。

表 4-12-2　　会议纪要记录表

5W2H	结论
WHAT	
HOW	
WHY	
WHEN	
WHERE	
WHO	
HOW MUCH	

在计划会议结束之前，所有团队成员把子任务进一步细分为更小的任务，把这些任务写在便签上，并把便签贴在看板上的“ToDo”列。

所有团队成员每天早上召开 15 分钟的站立会议，每人汇报三个问题，并更新看板上的“ToDo”、“Doing”和“Done”三列。

所有团队成员相互协作，编码实现子任务的业务功能，详细的代码请见智能农业实战项目完整源代码，核心代码片段如下：

```
//CO2 浓度
SensorBean co2=mSensorList.get(1);
co2.setValue(value.getCo2());
co2.setMinValue(config.minCo2);
co2.setMaxValue(config.maxCo2);
//光照强度
SensorBean light=mSensorList.get(2);
light.setValue(value.getLight());
light.setMinValue(config.minLight);
light.setMaxValue(config.maxLight);
//空气温度
SensorBean airT=mSensorList.get(3);
airT.setValue(value.getAirTemper());
airT.setMinValue(config.minAirTemperature);
airT.setMaxValue(config.maxAirTemperature);
//空气湿度
SensorBean airH=mSensorList.get(4);
airH.setValue(value.getAirHumid());
airH.setMinValue(config.minAirHumidity);
airH.setMaxValue(config.maxAirHumidity);
//土壤温度
SensorBean soilT=mSensorList.get(5);
soilT.setValue(value.getSoilTemper());
soilT.setMinValue(config.minSoilTemperature);
```

```
soilT.setMaxValue(config.maxSoilTemperature);
//土壤湿度
SensorBean soilH=mSensorList.get(6);
soilH.setValue(value.getSoilHumid());
soilH.setMinValue(config.minSoilHumidity);
soilH.setMaxValue(config.maxSoilHumidity);
```

开发工程师之间两两交叉代码评审，并根据评审纪要重构代码。

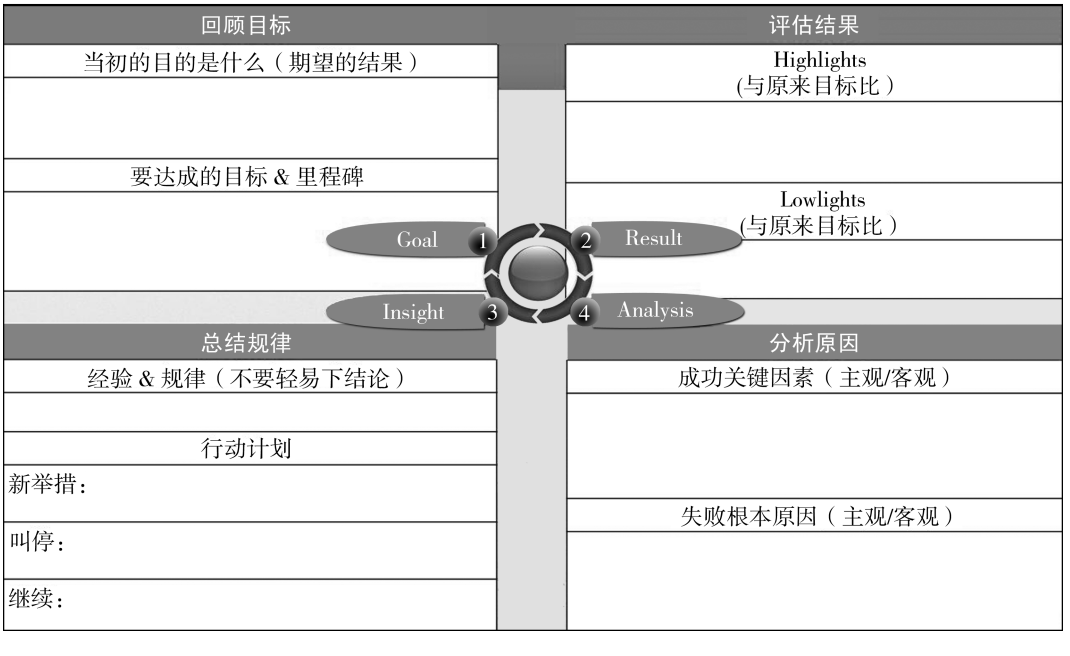
所有团队成员召开评审会，并填写表 4-12-3。

表 4-12-3　　评审建议表

序号	建议描述	是否接纳	原因
1		□接纳　□部分接纳　□未接纳	
2		□接纳　□部分接纳　□未接纳	
3		□接纳　□部分接纳　□未接纳	
4		□接纳　□部分接纳　□未接纳	
5		□接纳　□部分接纳　□未接纳	
6		□接纳　□部分接纳　□未接纳	
7		□接纳　□部分接纳　□未接纳	
8		□接纳　□部分接纳　□未接纳	
9		□接纳　□部分接纳　□未接纳	
10		□接纳　□部分接纳　□未接纳	

所有团队成员召开复盘会，并填写表 4-12-4。

表 4-12-4　　复盘会总结表



同学们，根据塔克曼阶梯理论，你们的团队目前处于哪个阶段呢？如果还没有进入成熟阶段，那就让我们一起努力吧，在接下来的任务中，更加注重相互配合和协作，及时发现并改正团队配合过程中出现的问题，早日让我们的团队进入成熟阶段！

4.12.5 任务扩展(Extend)

课后练习：优化实时环境指标界面的布局，使其具有更好的用户体验。

子任务 13 实现实时环境数据图表显示

- **子任务目标：**
  - 实时环境数据图表显示功能的实现
- **课时分配：**
  - 4 课时

4.13.1 任务构思(Conceive)

在智能农业实战项目中，需要一个实时环境数据图表界面，该界面共有 7 个图表，可以左右滑动进行切换。每隔 1 秒取一次实时数据并刷新一次，每次将数组内的数据向前移动一位，第一位丢弃，最后一位空出，并将读取到的数据置于数组的最后一位。

4.13.2 任务设计(Design)

在后台开启一个不断运行的线程，每隔 1 秒钟向服务器端请求一次传感器数据，每次将数组内的数据向前移动一位，第一位丢弃，最后一位空出，并将读取到的数据置于数组的最后一位。

4.13.3 任务实现(Implement)

智能农业实战项目的该子任务包含的角色为开发工程师。角色扮演的过程如表 4-13-1所示。

表 4-13-1　　角色扮演过程

步骤	角色	工作	输出
1	开发工程师	编码实现子任务的业务功能	实现子任务业务功能的 Android 源代码
2	开发工程师	两两交叉代码评审	代码评审纪要
3	开发工程师	根据代码评审纪要重构代码	重构后的 Android 源代码

4.13.4 任务运作(Operate)

所有团队成员召开计划会议，并将会议纪要记录在表 4-13-2 中。

表 4-13-2　　会议纪要记录表

5W2H	结论
WHAT	
HOW	
WHY	
WHEN	
WHERE	
WHO	
HOW MUCH	

在计划会议结束之前，所有团队成员把子任务进一步细分为更小的任务，把这些任务写在便签上，并把便签贴在看板上的“ToDo”列。

所有团队成员每天早上召开 15 分钟的站立会议，每人汇报三个问题，并更新看板上的“ToDo”、“Doing”和“Done”三列。

所有团队成员相互协作，编码实现子任务的业务功能，详细的代码请见智能农业实战项目完整源代码，核心代码片段如下：

```
//为各个图表添加新数据
mCo2Chart.majorValueList.add(sensor.getCo2());
mLightChart.majorValueList.add(sensor.getLight());
mAirTChart.majorValueList.add(sensor.getAirTemper());
mAirHChart.majorValueList.add(sensor.getAirHumid());
mSoilTChart.majorValueList.add(sensor.getSoilTemper());
mSoilHChart.majorValueList.add(sensor.getSoilHumid());
//如果数据点过多，则清除最老数据
if(mCo2Chart.majorValueList.size()>AppConfig.CHART_MAX_POINT){
    mCo2Chart.majorValueList.poll();
}
if(mLightChart.majorValueList.size()>AppConfig.CHART_MAX_POINT){
    mLightChart.majorValueList.poll();
}
if(mAirTChart.majorValueList.size()>AppConfig.CHART_MAX_POINT){
    mAirTChart.majorValueList.poll();
}
if(mAirHChart.majorValueList.size()>AppConfig.CHART_MAX_POINT){
    mAirHChart.majorValueList.poll();
}
if(mSoilTChart.majorValueList.size()>AppConfig.CHART_MAX_POINT){
    mSoilTChart.majorValueList.poll();
}
```

```
if(mSoilHChart.majorValueList.size()>AppConfig.CHART_MAX_POINT){
    mSoilHChart.majorValueList.poll();
}
//更新图表
mChartPagerAdp.notifyDataSetChanged();
```

开发工程师之间两两交叉代码评审，并根据评审纪要重构代码。

所有团队成员召开评审会，并填写表 4-13-3。

表 4-13-3　　评审建议表

序号	建议描述	是否接纳	原因
1		□接纳　□部分接纳　□未接纳	
2		□接纳　□部分接纳　□未接纳	
3		□接纳　□部分接纳　□未接纳	
4		□接纳　□部分接纳　□未接纳	
5		□接纳　□部分接纳　□未接纳	
6		□接纳　□部分接纳　□未接纳	
7		□接纳　□部分接纳　□未接纳	
8		□接纳　□部分接纳　□未接纳	
9		□接纳　□部分接纳　□未接纳	
10		□接纳　□部分接纳　□未接纳	

所有团队成员召开复盘会，并填写表 4-13-4。

表 4-13-4　　复盘会总结表

回顾目标	评估结果
当初的目的是什么（期望的结果）	Highlights (与原来目标比）
要达成的目标 & 里程碑	Lowlights (与原来目标比）
Goal 1　2 Result	
Insight 3　4 Analysis	
总结规律	**分析原因**
经验 & 规律（不要轻易下结论）	成功关键因素（主观/客观）
行动计划	失败根本原因（主观/客观）
新举措：	
叫停：	
继续：	

同学们，根据塔克曼阶梯理论，你们的团队目前处于哪个阶段呢？如果还没有进入成熟阶段，那就让我们一起努力吧，在接下来的任务中，更加注重相互配合和协作，及时发现并改正团队配合过程中出现的问题，早日让我们的团队进入成熟阶段！

4.13.5　任务扩展(Extend)

课后练习：把数组的实现方式改为用列表的方式进行实现。

子任务14　实现滑动切换环境指标子界面

- **子任务目标：**

  - 应用环境指标子界面的滑动切换功能

- **课时分配：**

  - 6课时

4.14.1　任务构思(Conceive)

在智能农业实战项目中，客户端的环境指标界面里有七个传感器实时数据曲线显示界面，这七个界面需要通过手势滑动或者点击标记点进行界面间的切换，并且需要根据当前的图表位置来高亮设置相应的标记点。

4.14.2　任务设计(Design)

在Android系统中，想要实现类似Launcher的界面滑动切换可以使用ViewPager通过以下5个步骤：

(1)首先ViewPager在处理滑动事件的时候要用到OnPageChangeListener；

(2)OnPageChangeListener这个接口需要实现三个方法：onPageScrollStateChanged，onPageScrolled，onPageSelected；

(3)在状态改变的时候调用onPageScrollStateChanged(int arg0)方法，其中arg0这个参数有三种状态：0，1，2。arg0 ==1的时候表示正在滑动，arg0==2的时候表示滑动完毕了，arg0==0的时候表示什么都没做；

(4)当页面在滑动的时候会调用onPageScrolled(int arg0，float arg1，int arg2)方法，在滑动被停止之前，此方法会一直得到调用。其中三个参数的含义分别为：arg0：当前页面及点击滑动的页面，arg1：当前页面偏移的百分比，arg2：当前页面偏移的像素位置；

(5)底部dot圆点的高亮设置和跳转通过protected void onLayout()方法实现。

4.14.3　任务实现(Implement)

智能农业实战项目的该子任务包含的角色为开发工程师。角色扮演的过程如表

4-14-1所示。

表 4-14-1　　角色扮演过程

步骤	角色	工作	输出
1	开发工程师	编码实现子任务的业务功能	实现子任务业务功能的 Android 源代码
2	开发工程师	两两交叉代码评审	代码评审纪要
3	开发工程师	根据代码评审纪要重构代码	重构后的 Android 源代码

4.14.4　任务运作(Operate)

所有团队成员召开计划会议,并将会议纪要记录在表 4-14-2 中。

表 4-14-2　　会议纪要记录表

5W2H	结论
WHAT	
HOW	
WHY	
WHEN	
WHERE	
WHO	
HOW MUCH	

在计划会议结束之前,所有团队成员把子任务进一步细分为更小的任务,把这些任务写在便签上,并把便签贴在看板上的“ToDo”列。

所有团队成员每天早上召开 15 分钟的站立会议,每人汇报三个问题,并更新看板上的“ToDo”、“Doing”和“Done”三列。

所有团队成员相互协作,编码实现子任务的业务功能,详细的示例代码请见 Android 项目:PageSwitch,参考代码如下:

(1)创建布局文件:layout.xml

```
<? xml version= "1.0"encoding= "utf-8"? >
<FrameLayout xmlns:android= "http://schemas.android.com/apk/res/android"
    android:layout_width= "fill_parent"
    android:layout_height= "fill_parent">

</FrameLayout>
```

(2)创建布局文件:main.xml

```
<? xml version= "1.0"encoding= "utf-8"? >
<RelativeLayout xmlns:android= "http://schemas.android.com/apk/res/android"
    android:layout_width= "fill_parent"
    android:layout_height= "fill_parent">
```

```
<com.lenovo.scroll.MyScrollLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/ScrollLayout"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent">

    <FrameLayout
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:background="@drawable/guide01">
    </FrameLayout>

    <FrameLayout
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:background="@drawable/guide02">
    </FrameLayout>

    <FrameLayout
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:background="@drawable/guide03">
    </FrameLayout>

    <FrameLayout
    android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:background="@drawable/guide04">
    </FrameLayout>

    <FrameLayout
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:background="@drawable/guide05">
    </FrameLayout>
</com.lenovo.scroll.MyScrollLayout>

<LinearLayout
    android:id="@+id/llayout"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignParentBottom="true"
```

```
android:layout_centerHorizontal="true"
android:layout_marginBottom="24.0dip"
android:orientation="horizontal">

<ImageView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_gravity="center_vertical"
    android:clickable="true"
    android:padding="15.0dip"
    android:src="@drawable/guide_round"/>

<ImageView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_gravity="center_vertical"
    android:clickable="true"
    android:padding="15.0dip"
    android:src="@drawable/guide_round"/>

<ImageView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_gravity="center_vertical"
    android:clickable="true"
    android:padding="15.0dip"
    android:src="@drawable/guide_round"/>

<ImageView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_gravity="center_vertical"
    android:clickable="true"
    android:padding="15.0dip"
    android:src="@drawable/guide_round"/>

<ImageView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_gravity="center_vertical"
    android:clickable="true"
    android:padding="15.0dip"
    android:src="@drawable/guide_round"/>
```

```
    </LinearLayout>

</RelativeLayout>
```

(3)创建布局文件:main_viewpage_layout.xml

```
<? xml version="1.0"encoding="utf-8"? >
<RelativeLayout
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    xmlns:android="http://schemas.android.com/apk/res/android">

    <android.support.v4.view.ViewPager
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:id="@+id/viewpager">
    </android.support.v4.view.ViewPager>

    <LinearLayout
        android:orientation="horizontal"
        android:id="@+id/llayout"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginBottom="24.0dip"
        android:layout_alignParentBottom="true"
        android:layout_centerHorizontal="true">

            <ImageViewandroid:clickable="true"
                android:padding="15.0dip"
                android:layout_gravity="center_vertical"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:src="@drawable/guide_round"/>
            <ImageViewandroid:clickable="true"
                android:padding="15.0dip"
                android:layout_gravity="center_vertical"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:src="@drawable/guide_round"/>
            <ImageViewandroid:clickable="true"
                android:padding="15.0dip"
                android:layout_gravity="center_vertical"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:src="@drawable/guide_round"/>
```

```
        <ImageViewandroid:clickable= "true"
            android:padding= "15.0dip"
            android:layout_gravity= "center_vertical"
            android:layout_width= "wrap_content"
            android:layout_height= "wrap_content"
            android:src= "@drawable/guide_round"/>
        <ImageViewandroid:clickable= "true"
            android:padding= "15.0dip"
            android:layout_gravity= "center_vertical"
            android:layout_width= "wrap_content"
            android:layout_height= "wrap_content"
            android:src= "@drawable/guide_round"/>
        </LinearLayout>

    </RelativeLayout>
```

(4)添加 Java 类:SwitchViewDemoActivity.java

```
public class SwitchViewDemoActivity extends Activity implements
        OnViewChangeListener, OnClickListener {

    private MyScrollLayout mScrollLayout;
    private ImageView[] mImageViews;
    private int mViewCount;
    private int mCurSel;

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        requestWindowFeature(Window.FEATURE_LEFT_ICON);
        setContentView(R.layout.main);
        getWindow().setFeatureDrawableResource(Window.FEATURE_LEFT_ICON, R.drawable.icon);
        init();
        Log.v("SwitchViewDemoActivity onClick()", "this is in SwitchViewDemoActivity onClick()");
    }

    private void init() {
        mScrollLayout=(MyScrollLayout) findViewById(R.id.ScrollLayout);
        LinearLayoutlinearLayout=(LinearLayout) findViewById(R.id.llayout);
        mViewCount=mScrollLayout.getChildCount();
        mImageViews=new ImageView[mViewCount];
        for (inti=0; i<mViewCount; i++) {
            mImageViews[i]=(ImageView) linearLayout.getChildAt(i);
            mImageViews[i].setEnabled(true);
            mImageViews[i].setOnClickListener(this);
```

```
            mImageViews[i].setTag(i);
        }
        mCurSel=0;
        mImageViews[mCurSel].setEnabled(false);
        mScrollLayout.SetOnViewChangeListener(this);
        Log.v("SwitchViewDemoActivity init()", "this is in SwitchViewDemoActivity init()");
    }

    //当前界面标记点高亮
    private void setCurPoint(intindex) {
        if (index< 0 || index>mViewCount - 1 || mCurSel == index) {
            return;
        }
        mImageViews[mCurSel].setEnabled(true);
        mImageViews[index].setEnabled(false);
        mCurSel=index;
    }

    @Override
    public void OnViewChange(intview) {

        setCurPoint(view);
    }

    @Override
    public void onClick(View v) {

        int pos=(Integer) (v.getTag());
        setCurPoint(pos);
        mScrollLayout.snapToScreen(pos);
    }
}
```

(5)添加 Java 类:MyScrollLayout.java

```
public class MyScrollLayout extends ViewGroup {

private static final String TAG ="ScrollLayout";
private VelocityTracker mVelocityTracker; // 用于判断甩动手势
private static final intSNAP_VELOCITY =600;
private Scroller mScroller; // 滑动控制器
private int mCurScreen;
private int mDefaultScreen=0;
private float mLastMotionX;
```

```
private OnViewChangeListener mOnViewChangeListener;

public MyScrollLayout(Context context) {
    super(context);

    init(context);
}

public MyScrollLayout(Context context, AttributeSet attrs) {
    super(context, attrs);

    init(context);
}

public MyScrollLayout(Context context, AttributeSet attrs, intdefStyle) {
    super(context, attrs, defStyle);

    init(context);
}

private void init(Context context) {
    mCurScreen=mDefaultScreen;
    mScroller=new Scroller(context);

}

// 实现点击 dot 点跳转页面
@Override
protected void onLayout(booleanchanged, intl, intt, intr, intb) {

    if (changed) {
        int childLeft=0;
        final int childCount=getChildCount();
        for (inti=0; i<childCount; i++) {
            final View childView=getChildAt(i);
            if (childView.getVisibility() != View.GONE) {
                final int childWidth=childView.getMeasuredWidth();
                childView.layout(childLeft, 0, childLeft + childWidth,
                        childView.getMeasuredHeight());
                childLeft += childWidth;
            }
```

```
        }
    }
}

@Override
protected void onMeasure(int widthMeasureSpec, int heightMeasureSpec) {

    super.onMeasure(widthMeasureSpec, heightMeasureSpec);
    final int width=MeasureSpec.getSize(widthMeasureSpec);
    final int count=getChildCount();
    for (inti=0; i<count; i++) {
        getChildAt(i).measure(widthMeasureSpec, heightMeasureSpec);
    }
    scrollTo(mCurScreen * width, 0);
}

public void snapToDestination() {
    final int screenWidth=getWidth();
    final int destScreen=(getScrollX() + screenWidth / 2) / screenWidth;
    snapToScreen(destScreen);
}

public void snapToScreen(int whichScreen) {
    // 获取屏幕截图
    whichScreen=Math.max(0, Math.min(whichScreen, getChildCount() - 1));
    if (getScrollX() != (whichScreen * getWidth())) {
        final int delta=whichScreen * getWidth() - getScrollX();
        mScroller.startScroll(getScrollX(), 0, delta, 0,
                Math.abs(delta) * 2);

        mCurScreen=whichScreen;
        invalidate(); // 刷新 layout
        if (mOnViewChangeListener != null) {
            mOnViewChangeListener.OnViewChange(mCurScreen);
        }
    }
}

@Override
public void computeScroll() {
```

```
        if (mScroller.computeScrollOffset()) {
            scrollTo(mScroller.getCurrX(), mScroller.getCurrY());
            postInvalidate();
        }
    }

    // onTouchEvent 方法
    @Override
    public boolean onTouchEvent(MotionEvent event) {

        final int action=event.getAction();
        final floatx=event.getX();
        switch (action) {
        case MotionEvent.ACTION_DOWN:
            Log.i("", "onTouchEvent  ACTION_DOWN");
            if (mVelocityTracker == null) {
                mVelocityTracker=VelocityTracker.obtain();
                mVelocityTracker.addMovement(event);
            }
            if (! mScroller.isFinished()) {
                mScroller.abortAnimation();
            }
            mLastMotionX=x;
            break;

        case MotionEvent.ACTION_MOVE:
            int deltaX=(int) (mLastMotionX - x);
            if (IsCanMove(deltaX)) {
                if (mVelocityTracker ! = null) {
                    mVelocityTracker.addMovement(event);
                }
                mLastMotionX=x;
                scrollBy(deltaX, 0);
            }

            break;
        // 事件操作
        case MotionEvent.ACTION_UP:
            int velocityX=0;
            if (mVelocityTracker ! = null) {
                mVelocityTracker.addMovement(event);
```

```
                mVelocityTracker.computeCurrentVelocity(1000);
                velocityX=(int) mVelocityTracker.getXVelocity();
            }
            if (velocityX> SNAP_VELOCITY&&mCurScreen> 0) {
                // 滑动足够距离向左
                Log.e(TAG, "snap left");
                snapToScreen(mCurScreen - 1);
            } elseif (velocityX< -SNAP_VELOCITY
                    &&mCurScreen< getChildCount() - 1) {
                // 滑动足够距离向右
                Log.e(TAG, "snap right");
                snapToScreen(mCurScreen + 1);
            } else {
                snapToDestination();
            }

            if (mVelocityTracker != null) {
                mVelocityTracker.recycle();
                mVelocityTracker=null;
            }

            break;
        }
        return true;
    }

    private boolean IsCanMove(int deltaX) {
        if (getScrollX() <= 0 &&deltaX< 0) {
            return false;
        }
        if (getScrollX() >= (getChildCount() - 1) * getWidth() &&deltaX> 0) {
            return false;
        }
        return true;
    }

    // 监听屏幕
    public void SetOnViewChangeListener(OnViewChangeListener listener) {
        mOnViewChangeListener=listener;
    }
}
```

(6)修改 AndroidManifest.xml

```
<? xml version= "1.0"encoding= "utf-8"? >
<manifest xmlns:android= "http://schemas.android.com/apk/res/android"
    package= "com.lenovo.demo"
    android:versionCode= "1"
    android:versionName= "1.0">

    <application
        android:icon= "@drawable/app"
        android:label= "@string/app_name">
        <activity
            android:name= ".activity.SwitchViewDemoActivity"
            android:configChanges= "orientation|keyboardHidden"
            android:label= "@string/app_name"
            android:screenOrientation= "portrait"
            android:theme= "@android:style/Theme.Black">
            <intent-filter>
                <action android:name= "android.intent.action.MAIN"/>

                <category android:name= "android.intent.category.LAUNCHER"/>
            </intent-filter>
        </activity>

    </application>

</manifest>
```

确保以上每一步都编译通过:在 IDE 集成开发环境中,没有出现红叉的错误提示。应用程序的运行效果如图 4-14-1 和图 4-14-2 所示。

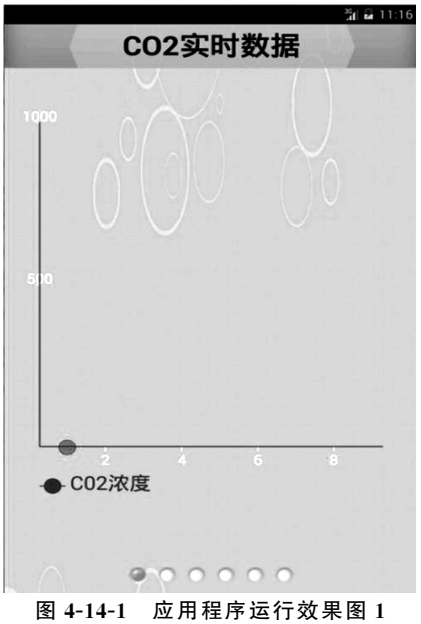


图 4-14-1　应用程序运行效果图 1

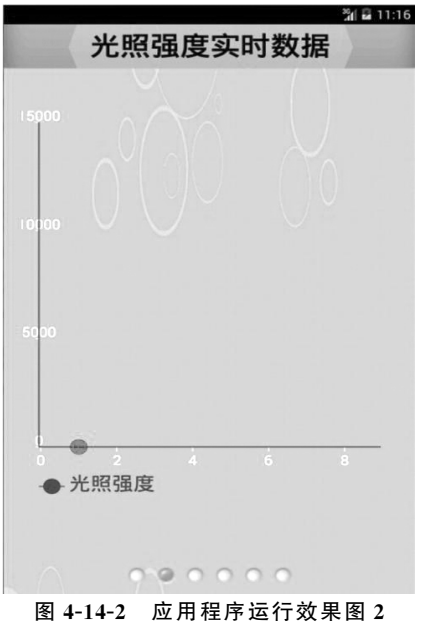


图 4-14-2　应用程序运行效果图 2

开发工程师之间两两交叉代码评审，并根据评审纪要重构代码。

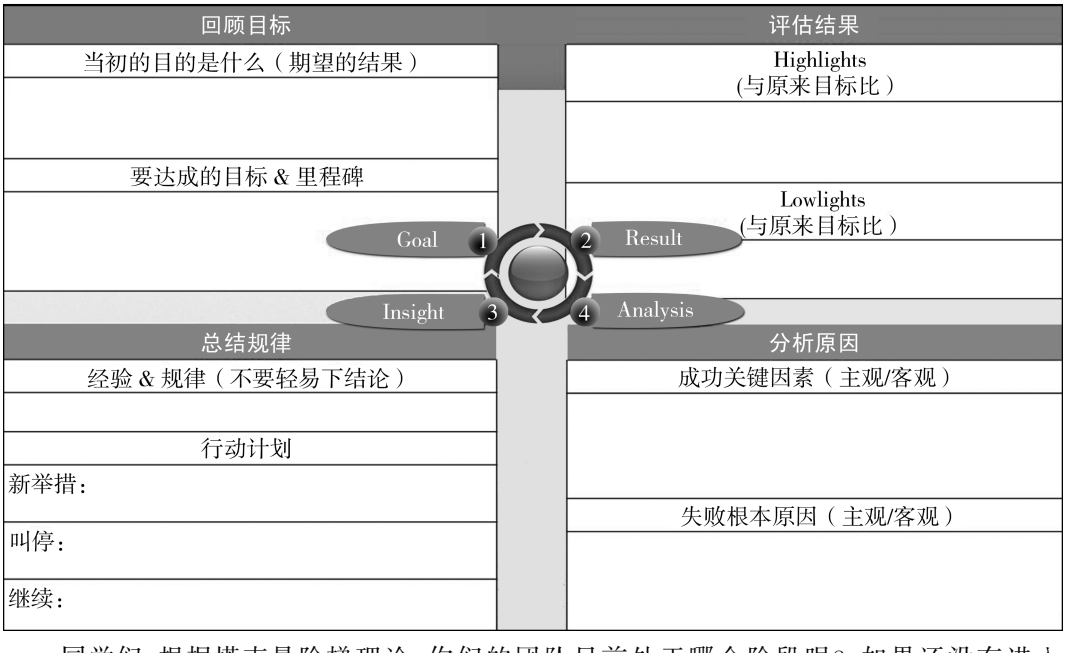
所有团队成员召开评审会，并填写表 4-14-3。

表 4-14-3　　评审建议表

序号	建议描述	是否接纳	原因
1		□接纳　□部分接纳　□未接纳	
2		□接纳　□部分接纳　□未接纳	
3		□接纳　□部分接纳　□未接纳	
4		□接纳　□部分接纳　□未接纳	
5		□接纳　□部分接纳　□未接纳	
6		□接纳　□部分接纳　□未接纳	
7		□接纳　□部分接纳　□未接纳	
8		□接纳　□部分接纳　□未接纳	
9		□接纳　□部分接纳　□未接纳	
10		□接纳　□部分接纳　□未接纳	

所有团队成员召开复盘会，并填写表 4-14-4。

表 4-14-4　　复盘会总结表



同学们，根据塔克曼阶梯理论，你们的团队目前处于哪个阶段呢？如果还没有进入成熟阶段，那就让我们一起努力吧，在接下来的任务中，更加注重相互配合和协作，及时发现并改正团队配合过程中出现的问题，早日让我们的团队进入成熟阶段！

4.14.5 任务扩展(Extend)

课后练习:当左右滑动到最后一个界面时,再继续滑动可以显示第一个界面。

子任务 15 实现显示历史数据

- **子任务目标:**

  - 应用历史数据的显示功能

- **课时分配:**

  - 4 课时

4.15.1 任务构思(Conceive)

在智能农业实战项目中,客户端将读取到的传感器数据存储到数据库,用户可以通过选择传感器类型和时间段,从数据库中读取历史数据,然后以图表的方式进行展示,支持左右滚动和缩放。

4.15.2 任务设计(Design)

当用户点击“查询”按钮以后,马上显示等待框,然后开启一个数据库查询线程,根据用户提交的条件查询相关传感器数据,最后将结果传给 UI 线程展示并关闭提示框。

4.15.3 任务实现(Implement)

智能农业实战项目的该子任务包含的角色为开发工程师。角色扮演的过程如表 4-15-1所示。

表 4-15-1　　角色扮演过程

步骤	角色	工作	输出
1	开发工程师	编码实现子任务的业务功能	实现子任务业务功能的 Android 源代码
2	开发工程师	两两交叉代码评审	代码评审纪要
3	开发工程师	根据代码评审纪要重构代码	重构后的 Android 源代码

4.15.4 任务运作(Operate)

所有团队成员召开计划会议,并将会议纪要记录在表 4-15-2 中。

表 4-15-2　　会议纪要记录表

5W2H	结论
WHAT	
HOW	
WHY	
WHEN	
WHERE	
WHO	
HOW MUCH	

在计划会议结束之前，所有团队成员把子任务进一步细分为更小的任务，把这些任务写在便签上，并把便签贴在看板上的“ToDo”列。

所有团队成员每天早上召开 15 分钟的站立会议，每人汇报三个问题，并更新看板上的“ToDo”、“Doing”和“Done”三列。

所有团队成员相互协作，编码实现子任务的业务功能，详细的代码请见智能农业实战项目完整源代码，核心代码片段如下：

```
//清除老数据
mChartPagerBean.majorValueList.clear();
mChartPagerBean.slaveValueList.clear();
//根据传感器类型设置相关曲线图表 Y 轴的最大值和值小值
switch(mCurrentType){
case R.string.CO2_title:
    mChartPagerBean.majorName=getString(R.string.CO2_title);
    mChartPagerBean.majorMin=0;//co2 的最小值
    mChartPagerBean.majorMax=1000;//co2 的最大值
    mChartPagerBean.majorWarningMin=mSensorConfig.minCo2;//co2 的告警最小值
    mChartPagerBean.majorWarningMax=mSensorConfig.maxCo2;//co2 的告警最大值
    break;
case R.string.light_title:
    mChartPagerBean.majorName=getString(R.string.light_title);
    mChartPagerBean.majorMin=0;//光照强度最小值
    mChartPagerBean.majorMax=15000;//光照强度最大值
    mChartPagerBean.majorWarningMin=mSensorConfig.minLight;//光照强度的告警最小值
    mChartPagerBean.majorWarningMax=mSensorConfig.maxLight;//光照强度的告警最大值
    break;
case R.string.air_tmper_title:
    mChartPagerBean.majorName=getString(R.string.air_tmper_title);
    mChartPagerBean.majorMin=0;//空气温度最小值
    mChartPagerBean.majorMax=100;//空气温度最大值
    mChartPagerBean.majorWarningMin=mSensorConfig.minAirTemperature;//空气温度的告警最小值
```

```
        mChartPagerBean.majorWarningMax=mSensorConfig.maxAirTemperature;//空气温度的告警最大值
        break;
    case R.string.air_humid_title:
        mChartPagerBean.majorName=getString(R.string.air_humid_title);
        mChartPagerBean.majorMin=0;//空气湿度最小值
        mChartPagerBean.majorMax=100;//空气湿度最大值
        mChartPagerBean.majorWarningMin=mSensorConfig.minAirHumidity;//空气湿度的告警最小值
        mChartPagerBean.majorWarningMax=mSensorConfig.maxAirHumidity;//空气湿度的告警最大值
        break;
    case R.string.soil_tmper_title:
        mChartPagerBean.majorName=getString(R.string.soil_tmper_title);
        mChartPagerBean.majorMin=0;//土壤温度最小值
        mChartPagerBean.majorMax=100;//土壤温度最大值
        mChartPagerBean.majorWarningMin=mSensorConfig.minSoilTemperature;//土壤温度的告警最小值
        mChartPagerBean.majorWarningMax=mSensorConfig.maxSoilTemperature;//土壤温度的告警最大值
        break;
    case R.string.soil_humid_title:
        mChartPagerBean.majorName=getString(R.string.soil_humid_title);
        mChartPagerBean.majorMin=0;//土壤湿度最小值
        mChartPagerBean.majorMax=100;//土壤湿度最大值
        mChartPagerBean.majorWarningMin=mSensorConfig.minSoilHumidity;//土壤湿度的告警最小值
        mChartPagerBean.majorWarningMax=mSensorConfig.maxSoilHumidity;//土壤湿度的告警最大值
        break;
    }
    mChartPagerBean.isHasSlave=false;
```

开发工程师之间两两交叉代码评审，并根据评审纪要重构代码。

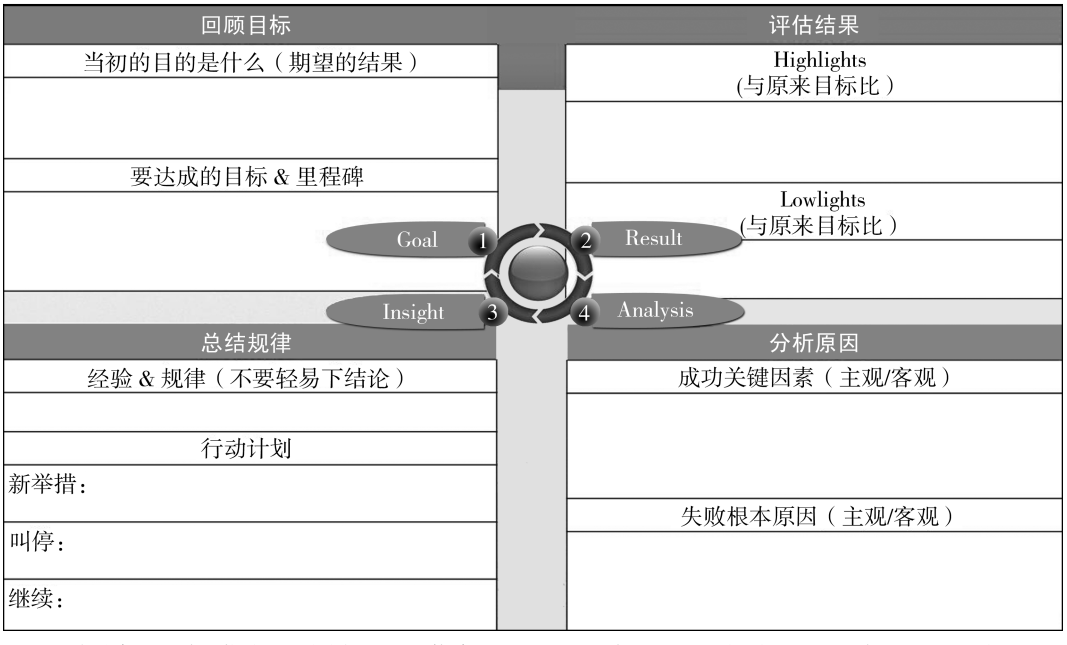
所有团队成员召开评审会，并填写表 4-15-3。

表 4-15-3　　评审建议表

序号	建议描述	是否接纳	原因
1		□接纳　□部分接纳　□未接纳	
2		□接纳　□部分接纳　□未接纳	
3		□接纳　□部分接纳　□未接纳	
4		□接纳　□部分接纳　□未接纳	
5		□接纳　□部分接纳　□未接纳	
6		□接纳　□部分接纳　□未接纳	
7		□接纳　□部分接纳　□未接纳	
8		□接纳　□部分接纳　□未接纳	
9		□接纳　□部分接纳　□未接纳	
10		□接纳　□部分接纳　□未接纳	

所有团队成员召开复盘会，并填写表 4-15-4。

表 4-15-4　　复盘会总结表



同学们，根据塔克曼阶梯理论，你们的团队目前处于哪个阶段呢？如果还没有进入成熟阶段，那就让我们一起努力吧，在接下来的任务中，更加注重相互配合和协作，及时发现并改正团队配合过程中出现的问题，早日让我们的团队进入成熟阶段！

4.15.5　任务扩展(Extend)

课后练习：使用 Google 的第三方绘图引擎 AchartEngine 绘制图形。

子任务 16　实现告警通知的产生

- **子任务目标：**

  ➢ 应用 Notification 控件实现告警通知的产生

- **课时分配：**

  ➢ 4 课时

4.16.1　任务构思(Conceive)

在智能农业项目中，当智能农业沙盘中的传感器检测到的环境指标不在设置的阀值范围内时，客户端的 App 要产生告警通知。

在 Android 系统中，实现提示信息的方式有：Dialog、Toast 和 Notification。

(1)Dialog 是以独占方式显示的，也就是说，如果不关闭 Dialog，就无法做其他事情。

(2)Toast 显示的提示信息,只能显示一小段时间,然后就消失了。

(3)Notification 显示在屏幕上方的状态栏中,不但可以长久保存,而且可以脱离应用程序存在。

因此,Notification 是实现智能农业项目中客户端告警通知的最好方式。

4.16.2 任务设计(Design)

在 Android 系统中,在状态栏生成并显示一个 Notification 可以通过如下 5 步实现:

(1)获得 NotificationManager 对象。

(2)创建 Notification 对象。发出通知的时间通常设置为当前时间。

(3)创建 PendingIntent 对象。创建该对象的过程中指定了 Intent 对象,使得下拉标题栏将其展开后,点击 Notification,将跳转到指定的 Activity。

(4)设置 Notification 的详细信息。详细信息包括:下拉标题栏将其展开后,Notification 的标题和文本。

(5)将 Notification 显示在标题栏。

4.16.3 任务实现(Implement)

智能农业实战项目的该子任务包含的角色为开发工程师。角色扮演的过程如表 4-16-1所示。

表 4-16-1　　角色扮演过程

步骤	角色	工作	输出
1	开发工程师	编码实现子任务的业务功能	实现子任务业务功能的 Android 源代码
2	开发工程师	两两交叉代码评审	代码评审纪要
3	开发工程师	根据代码评审纪要重构代码	重构后的 Android 源代码

4.16.4 任务运作(Operate)

所有团队成员召开计划会议,并将会议纪要记录在表 4-16-2 中。

表 4-16-2　　会议纪要记录表

5W2H	结论
WHAT	
HOW	
WHY	
WHEN	
WHERE	
WHO	
HOW MUCH	

在计划会议结束之前，所有团队成员把子任务进一步细分为更小的任务，把这些任务写在便签上，并把便签贴在看板上的“ToDo”列。

所有团队成员每天早上召开 15 分钟的站立会议，每人汇报三个问题，并更新看板上的“ToDo”、“Doing”和“Done”三列。

所有团队成员相互协作，编码实现子任务的业务功能，详细的示例代码请见 Android 项目：CreateNotification，参考代码如下：

(1)创建布局文件：activity_main. xml

```
<? xml version= "1.0"encoding= "utf-8"? >
<LinearLayout xmlns:android= "http://schemas.android.com/apk/res/android"
    android:layout_width= "match_parent"
    android:layout_height= "match_parent"
    android:orientation= "vertical">

<Button
    android:id= "@+id/button_light"
    android:layout_width= "wrap_content"
    android:layout_height= "wrap_content"
    android:drawableLeft= "@drawable/light"
    android:drawablePadding= "10dp"
    android:text= "@string/button_light"/>
<Button
    android:id= "@+id/button_pm25"
    android:layout_width= "wrap_content"
    android:layout_height= "wrap_content"
    android:drawableLeft= "@drawable/pm25"
    android:drawablePadding= "10dp"
    android:text= "@string/button_pm25"/>

</LinearLayout>
```

(2)创建 Activity 类：MainActivity. java

```
/ * *
 * 主页面对应的 Activity
 * /
public class MainActivity extends Activity implements OnClickListener
{
    private Button mButtonLight;
    private Button mButtonPM25;
    private Resources mResources;

    / * *
     * 创建 MainActivity 时被调用
```

```
 */
@Override
public void onCreate(Bundle savedInstanceState)
{
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);

    findViewsById();
    setListeners();

    mResources=getResources();
}

/**
 *根据 id 找到相应的 View
 */
private void findViewsById() {
    mButtonLight=(Button) findViewById(R.id.button_light);
    mButtonPM25=(Button) findViewById(R.id.button_pm25);
}

/**
 *设置 Listener
 */
private void setListeners() {
    mButtonLight.setOnClickListener(this);
    mButtonPM25.setOnClickListener(this);
}

/**
 *处理点击事件
 */
@Override
public void onClick(View view)
{
    switch (view.getId())
    {
        case R.id.button_light:
            showNotification(R.drawable.light,
                    mResources.getString(R.string.ticker_text_light),
                    mResources.getString(R.string.content_title_light),
                    mResources.getString(R.string.content_text_light),
                    R.id.button_light);
```

```
            break;
        case R.id.button_pm25:
            showNotification(R.drawable.pm25,
                    mResources.getString(R.string.ticker_text_pm25),
                    mResources.getString(R.string.content_title_pm25),
                    mResources.getString(R.string.content_text_pm25),
                    R.id.button_pm25);
            break ;
        default :
            break ;
    }
}

/**
 * 生成 Notification 并显示在标题栏
 *
 * @param resId Notification 文本前方的图像资源 ID
 * @param tickerText 显示在标题栏中的 Notification 文本
 * @param contentTitle 下拉标题栏将其展开后,Notification 的标题
 * @param contentText 下拉标题栏将其展开后,Notification 的文本
 * @param id 标识 Notification 的唯一 ID
 */
@SuppressWarnings("deprecation")
private void showNotification(intresId, String tickerText, String contentTitle,
        String contentText, int id)
{
    // 1.获得 NotificationManager 对象。
    NotificationManager notificationManager=(NotificationManager) getSystemService
(NOTIFICATION_SERVICE);
    /*
     * 2.创建 Notification 对象。生成 Notification 的时间通常设置为当前时间。
     */
    Notification notification=new ~~Notification~~(resId,
            tickerText, System.currentTimeMillis());
    /*
     * 3.创建 PendingIntent 对象。
     *创建该对象的过程中指定了 Intent 对象,使得下拉标题栏将其展开后,点击
Notification,将跳转到指定的 Activity。
     *该对象使得即使关闭应用程序,Notification 仍然会显示在状态栏中。
     */
    PendingIntent pendingIntent=PendingIntent.getActivity(this , 0,
            new Intent(this , MainActivity.class ), 0);
```

```
        // 4.设置 Notification 的详细信息
        notification.setLatestEventInfo(this , contentTitle, contentText,
                pendingIntent);

        // 5.将 Notification 显示在标题栏
        notificationManager.notify(id, notification);
    }
}
```

(3)在 AndroidManifest.xml 中注册 Activity

```
<manifest xmlns:android= "http://schemas.android.com/apk/res/android"
    package= "com.lenovo.demo"
    android:versionCode= "1"
    android:versionName= "1.0">
    <uses-sdk
        android:minSdkVersion= "15"
        android:targetSdkVersion= "19"/>
    <application
        android:allowBackup= "true"
        android:icon= "@drawable/app"
        android:label= "@string/app_name">
        <activityandroid:name= ".activity.MainActivity">
            <intent-filter>
                <action android:name= "android.intent.action.MAIN"/>

                <category android:name= "android.intent.category.LAUNCHER"/>
            </intent-filter>
        </activity>
    </application>

</manifest>
```

(4)修改 strings.xml

```
<resources>

    <string name= "app_name">NotificationDemo</string>
    <string name= "button_light">生成光照告警通知</string>
    <string name= "button_pm25">生成 PM2.5 告警通知</string>
    <string name= "ticker_text_light">光照告警了</string>
    <string name= "ticker_text_pm25">PM2.5 告警了</string>
    <string name= "content_title_light">光照告警通知</string>
    <string name= "content_title_pm25">PM2.5 告警通知</string>
    <string name= "content_text_light">光照强度超出阀值</string>
    <string name= "content_text_pm25">PM2.5 浓度超出阀值</string>
</resources>
```

确保以上每一步都编译通过：在 IDE 集成开发环境中，没有出现红叉的错误提示。

应用程序的运行效果如图 4-16-1 所示。

点击“生成光照告警通知”按钮，在状态栏显示“光照告警了”，运行效果如图 4-16-2 所示。

图 4-16-1　应用程序运行效果图

图 4-16-2　“光照告警了”运行效果图

点击“生成 PM 2.5 告警通知”按钮，在状态栏显示“光照告警了”，运行效果如图 4-16-3所示。

下拉状态栏将其展开，如图 4-16-4 所示。

图 4-16-3　“PM2.5 告警了”运行效果图

图 4-16-4“PM2.5 告警了”下拉状态栏展开图

点击两个 Notification 中的任意一个，将跳转到应用程序的主界面，如图 4-16-5 所示。

图 4-16-5　应用程序主界面

开发工程师之间两两交叉代码评审，并根据评审纪要重构代码。

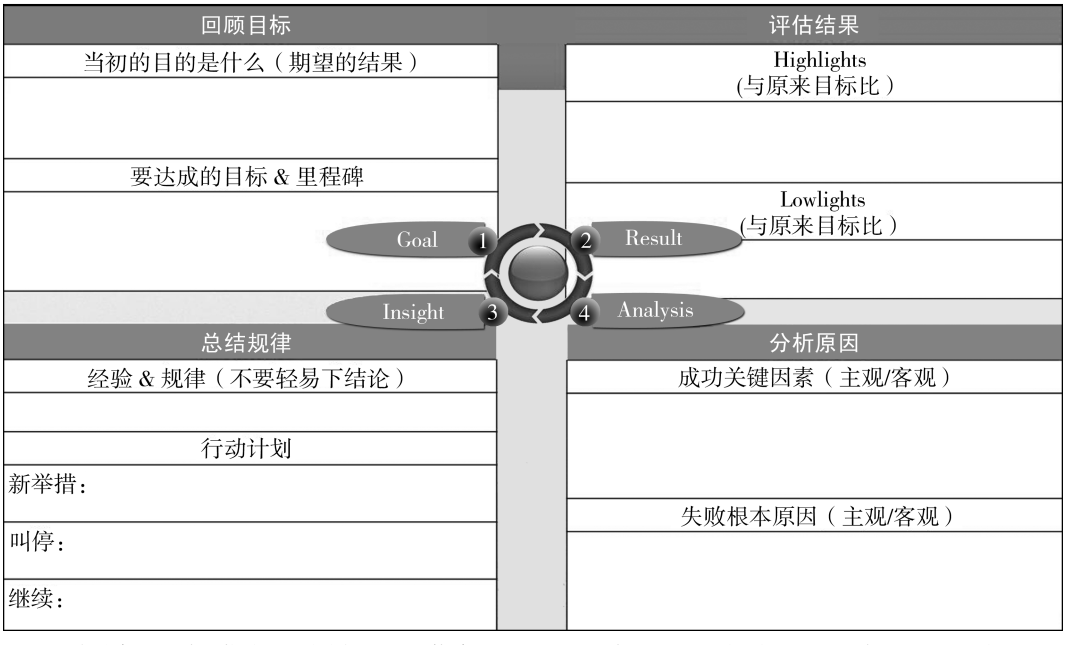
所有团队成员召开评审会，并填写表 4-16-3。

表 4-16-3　　评审建议表

序号	建议描述	是否接纳	原因
1		□接纳　□部分接纳　□未接纳	
2		□接纳　□部分接纳　□未接纳	
3		□接纳　□部分接纳　□未接纳	
4		□接纳　□部分接纳　□未接纳	
5		□接纳　□部分接纳　□未接纳	
6		□接纳　□部分接纳　□未接纳	
7		□接纳　□部分接纳　□未接纳	
8		□接纳　□部分接纳　□未接纳	
9		□接纳　□部分接纳　□未接纳	
10		□接纳　□部分接纳　□未接纳	

所有团队成员召开复盘会，并填写表 4-16-4。

表 4-16-4　　复盘会总结表



同学们，根据塔克曼阶梯理论，你们的团队目前处于哪个阶段呢？如果还没有进入成熟阶段，那就让我们一起努力吧，在接下来的任务中，更加注重相互配合和协作，及时发现并改正团队配合过程中出现的问题，早日让我们的团队进入成熟阶段！

4.16.5　任务扩展(Extend)

使用 Dialog 和 Toast 实现告警通知。

课后练习：在应用程序中再定义两个 Activity：LightActivity 和 PM25Activity，下拉状态栏将其展开后，点击光照告警通知，则跳转到 LightActivity；点击 PM25 告警通知，则跳转到 PM25Activity。

子任务 17　实现告警通知消除

- **子任务目标：**

  ➢ 应用 Notification 控件实现告警通知的消除

- **课时分配：**

  ➢ 4 课时

4.17.1　任务构思(Conceive)

在智能农业项目中，当智能农业沙盘中的传感器检测到的环境指标不在设置的阀值

范围内时，客户端的 App 要产生告警通知，如果想要消除产生的告警通知，有两种实现方式：

(1)下拉任务栏手动消除产生的告警通知；

(2)在 App 中定义一个 Button 手动消除产生的告警通知。

第一种方式的实现方式是系统提供的，不需要编写代码即可实现。为了学习 Notification 的相关知识点，我们将采用第二种实现方式。

4.17.2 任务设计(Design)

如果想要消除某个 Notification 通知，那么只需要调用 cancel()或 cancelAll()方法即可。

4.17.3 任务实现(Implement)

智能农业实战项目的该子任务包含的角色为开发工程师。角色扮演的过程如表 4-17-1所示。

表 4-17-1　　角色扮演过程

步骤	角色	工作	输出
1	开发工程师	编码实现子任务的业务功能	实现子任务业务功能的 Android 源代码
2	开发工程师	两两交叉代码评审	代码评审纪要
3	开发工程师	根据代码评审纪要重构代码	重构后的 Android 源代码

4.17.4 任务运作(Operate)

所有团队成员召开计划会议，并将会议纪要记录在表 4-17-2 中。

表 4-17-2　　会议纪要记录表

5W2H	结论
WHAT	
HOW	
WHY	
WHEN	
WHERE	
WHO	
HOW MUCH	

在计划会议结束之前，所有团队成员把子任务进一步细分为更小的任务，把这些任务写在便签上，并把便签贴在看板上的“ToDo”列。

所有团队成员每天早上召开 15 分钟的站立会议，每人汇报三个问题，并更新看板上的“ToDo”、“Doing”和“Done”三列。

所有团队成员相互协作，编码实现子任务的业务功能，详细的示例代码请见 Android 项目：CancelNotification，参考代码如下：

(1)创建布局文件：activity_main. xml

```
<RelativeLayout xmlns:android= "http://schemas.android.com/apk/res/android"
    xmlns:tools= "http://schemas.android.com/tools"
    android:layout_width= "match_parent"
    android:layout_height= "match_parent"
    android:paddingBottom= "@dimen/activity_vertical_margin"
    android:paddingLeft= "@dimen/activity_horizontal_margin"
    android:paddingRight= "@dimen/activity_horizontal_margin"
    android:paddingTop= "@dimen/activity_vertical_margin"
    tools:context= ".MainActivity">

    <Button
        android:id= "@+id/startBtn1"
        android:layout_width= "wrap_content"
        android:layout_height= "wrap_content"
        android:layout_alignLeft= "@+id/cancelBtn1"
        android:layout_alignParentTop= "true"
        android:layout_marginTop= "38dp"
        android:text= "产生光照告警通知"/>

    <Button
        android:id= "@+id/cancelBtn1"
        android:layout_width= "wrap_content"
        android:layout_height= "wrap_content"
        android:layout_below= "@+id/startBtn1"
        android:layout_centerHorizontal= "true"
        android:layout_marginTop= "27dp"
        android:text= "取消光照告警通知"/>

    <Button
        android:id= "@+id/cancelBtn2"
        android:layout_width= "wrap_content"
        android:layout_height= "wrap_content"
        android:layout_alignParentBottom= "true"
        android:layout_centerHorizontal= "true"
        android:layout_marginBottom= "106dp"
        android:text= "取消 PM2.5 告警通知"/>

    <Button
```

```
        android:id="@+id/startBtn2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignLeft="@+id/cancelBtn2"
        android:layout_centerVertical="true"
        android:text="生成 PM2.5 告警通知"/>

</RelativeLayout>
```

(2)创建 Activity 类:MainActivity.java

```
public class MainActivity extends Activity {

    private boolean pm=false ;
    private bool eanpmcancle=false;
    private boolean light=false;
    private booleanl ightcancle=false;
    private static final int Light_ID =1;
    private static finalint PM_ID =2;

    NotificationManager mNotificationManager;
    Notification mNotification;
    private Button LightBtn;
    private Button LightcancelBtn;
    private Button PMBtn;
    private Button PMcancelBtn;
    private Resources mResources;

    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main );

        LightBtn=(Button) findViewById(R.id.startBtn1);
        LightcancelBtn=(Button) findViewById(R.id.cancelBtn1);

        PMBtn=(Button) findViewById(R.id.startBtn2);
        PMcancelBtn=(Button) findViewById(R.id.cancelBtn2);

        mResources=getResources();

        LightBtn.setOnClickListener(new OnClickListener() {
```

```
        @Override
        public void onClick(View v) {
            light=true;
            create_notification();
        }
    });

    LightcancelBtn.setOnClickListener(new OnClickListener() {

        @Override
        public void onClick(View v) {
            light=false;
            lightcancle=true;
            create_notification();
        }
    });

    PMBtn.setOnClickListener(new OnClickListener() {

        @Override
        public void onClick(View v) {
            pm=true;
            create_notification();
        }
    });

    PMcancelBtn.setOnClickListener(new OnClickListener() {

        @Override
        public void onClick(View v) {
            pm=false;
            pmcancle=true;
            create_notification();
        }
    });
}

private void create_notification() {
    String ns=Context.NOTIFICATION_SERVICE;
    mNotificationManager=(NotificationManager) this.getSystemService(ns);
```

```
        int icon=R.drawable.g;
        CharSequence tickerText="lenovo";
        long when=System.currentTimeMillis();
        mNotification=new Notification(icon, tickerText, when);
        mNotification.defaults=Notification.DEFAULT_ALL;
        mNotification.flags|=Notification.FLAG_NO_CLEAR;
        mNotification.flags|=Notification.FLAG_SHOW_LIGHTS;
        Context context=this;
        CharSequence contentTitlelight=mResources.getString(R.string.content_title_light);
        CharSequence contentTitlepm=mResources.getString(R.string.content_title_pm);
        CharSequence contentText=mResources.getString(R.string.content_need);
        Intent notificationIntent=new Intent(this, MainActivity.class);
        PendingIntent contentIntent=PendingIntent.getActivity(context, 0,
                notificationIntent, 0);
        if (light) {
            mNotification
                    .setLatestEventInfo(context, contentTitlelight, contentText,
                contentIntent);
            mNotificationManager.notify(Light_ID, mNotification);
            light=false;
        }
        if (light==false&&lightcancle==true) {
            mNotificationManager.cancel(Light_ID);
        }
        if (pm) {
            mNotification.setLatestEventInfo(context, contentTitlepm, contentText,
        contentIntent);
            mNotificationManager.notify(PM_ID, mNotification);
            pm=false;
        }
        if (pm==false&&pmcancle==true) {
            mNotificationManager.cancel(PM_ID);
        }
    }
}
```

确保以上每一步都编译通过：在 IDE 集成开发环境中，没有出现红叉的错误提示。应用程序的运行效果如图 4-17-1 所示。

图 4-17-1　应用程序运行效果图

开发工程师之间两两交叉代码评审，并根据评审纪要重构代码。

所有团队成员召开评审会，并填写表 4-17-3。

表 4-17-3　　评审建议表

序号	建议描述	是否接纳	原因
1		□接纳　□部分接纳　□未接纳	
2		□接纳　□部分接纳　□未接纳	
3		□接纳　□部分接纳　□未接纳	
4		□接纳　□部分接纳　□未接纳	
5		□接纳　□部分接纳　□未接纳	
6		□接纳　□部分接纳　□未接纳	
7		□接纳　□部分接纳　□未接纳	
8		□接纳　□部分接纳　□未接纳	
9		□接纳　□部分接纳　□未接纳	
10		□接纳　□部分接纳　□未接纳	

所有团队成员召开复盘会，并填写表 4-17-4。

表 4-17-4　　复盘会总结表

回顾目标		评估结果
当初的目的是什么（期望的结果）		Highlights (与原来目标比)
要达成的目标 & 里程碑		Lowlights (与原来目标比)
	Goal 1 / 2 Result	
	Insight 3 / 4 Analysis	
总结规律		分析原因
经验 & 规律（不要轻易下结论）		成功关键因素（主观/客观）
行动计划		
新举措：		失败根本原因（主观/客观）
叫停：		
继续：		

同学们，根据塔克曼阶梯理论，你们的团队目前处于哪个阶段呢？如果还没有进入成熟阶段，那就让我们一起努力吧，在接下来的任务中，更加注重相互配合和协作，及时发现并改正团队配合过程中出现的问题，早日让我们的团队进入成熟阶段！

4.17.5　任务扩展(Extend)

课后练习：当产生告警通知后，直接调用 cancelAll()方法消除所有的告警通知。

子任务 18　实现系统设置布局

- **子任务目标：**
  - 应用相关布局和控件实现系统设置布局
- **课时分配：**
  - 4 课时

4.18.1　任务构思(Conceive)

在智能农业项目中，系统设置主要包括对语言、二氧化碳、光照、空气、土壤等参数进行设置。系统提供了 5 种常见布局和若干常见控件，它们之间相互组合可实现各种想要的布局。

4.18.2　任务设计(Design)

所有的布局和控件都定义在 res 文件夹的 layout 子文件夹中。复杂的系统设置布局可通过 PreferencesActivity 和 PreferencesFragment 两个类实现，简单的系统设置布局仅通过 Dialog 和 Menu 即可实现。

4.18.3　任务实现(Implement)

智能农业实战项目的该子任务包含的角色为开发工程师。角色扮演的过程如表4-18-1所示。

表 4-18-1　　角色扮演过程

步骤	角色	工作	输出
1	开发工程师	编码实现子任务的业务功能	实现子任务业务功能的 Android 源代码
2	开发工程师	两两交叉代码评审	代码评审纪要
3	开发工程师	根据代码评审纪要重构代码	重构后的 Android 源代码

4.18.4　任务运作(Operate)

所有团队成员召开计划会议，并将会议纪要记录在表 4-18-2 中。

表 4-18-2　　会议纪要记录表

5W2H	结论
WHAT	
HOW	
WHY	
WHEN	
WHERE	
WHO	
HOW MUCH	

在计划会议结束之前，所有团队成员把子任务进一步细分为更小的任务，把这些任务写在便签上，并把便签贴在看板上的“ToDo”列。

所有团队成员每天早上召开 15 分钟的站立会议，每人汇报三个问题，并更新看板上的“ToDo”、“Doing”和“Done”三列。

所有团队成员相互协作，编码实现子任务的业务功能，详细的示例代码请见 Android 项目：CreateSystemSettingsLayout，参考代码如下：

(1)创建布局文件：setting_fragment. xml

```
<? xml version= "1.0"encoding= "utf-8"? >
```

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:background="#ffffff"
    android:gravity="center"
    android:orientation="vertical">
    <LinearLayout
        android:layout_width="300dip"
        android:layout_height="wrap_content"
        android:gravity="top|center_horizontal"
        android:orientation="vertical"
        android:paddingBottom="25dip"
        android:paddingLeft="15dip"
        android:paddingRight="15dip"
        android:paddingTop="20dip">
        <LinearLayout
            android:id="@+id/language_layout"
            android:layout_width="match_parent"
            android:layout_height="42dip"
            android:layout_marginLeft="5dip"
            android:layout_marginRight="5dip"
            android:layout_marginTop="3dip"
            android:orientation="horizontal">
            <TextView
                style="@style/setting_item_label_style"
                android:layout_width="0dip"
                android:layout_height="wrap_content"
                android:layout_weight="60"
                android:text="@string/language_seting"/>

            <View
                style="@style/setting_item_image_style"
                android:layout_width="33dip"
                android:layout_height="33dip"/>
        </LinearLayout>
        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="42dip"
            android:layout_marginLeft="5dip"
            android:layout_marginRight="5dip"
            android:layout_marginTop="3dip"
```

```
            android:orientation="horizontal">
            <TextView
                style="@style/setting_item_label_style"
                android:layout_width="0dip"
                android:layout_height="wrap_content"
                android:layout_weight="60"
                android:text="@string/auto_control"/>
            <CheckBox
                android:id="@+id/autoControlCheckBox"
                android:layout_width="50dip"
                android:layout_height="30dip"
                android:layout_gravity="center_vertical"
                android:layout_marginRight="10dip"
                android:background="@drawable/slide_check_box_btn"
                android:button="@null"/>
        </LinearLayout>
    </LinearLayout>

</LinearLayout>
```

(2)创建选择器文件:slide_check_box_btn.xml

```
<? xml version="1.0"encoding="utf-8"? >
<selector xmlns:android="http://schemas.android.com/apk/res/android">
    <item android:drawable="@drawable/slide_check_box_on" android:state_checked="true"></item>
    <item android:drawable="@drawable/slide_check_box_on"android:state_selected="true"></item>
    <item android:drawable="@drawable/slide_check_box_on"android:state_pressed="true"></item>
    <item android:drawable="@drawable/slide_check_box_off"></item>
</selector>
```

(3)添加布局文件:setting_detail_dialog.xml

```
<LinearLayout
    android:layout_width="0dip"
    android:layout_height="wrap_content"
    android:layout_weight="1"
    android:orientation="vertical">
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:gravity="center"
        android:orientation="horizontal">
```

```
        <TextView
            android:id="@+id/major_left_text_view"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="@string/air_tmper_current_value"
            android:textColor="@color/white"
            android:textSize="13sp"/>

        <TextView
            android:id="@+id/major_value_text_view"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_marginLeft="5dip"
            android:text="50"
            android:textColor="@color/white"
            android:textSize="15sp"/>
    </LinearLayout>
    <LinearLayout
        android:id="@+id/major_min_layout"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_margin="10dip"
        android:gravity="center"
        android:orientation="horizontal">
        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="@string/min_value"
            android:textColor="@color/white"
            android:textSize="13sp"/>
        <EditText
            android:id="@+id/major_min_value_edit_text"
            android:layout_width="50dip"
            android:layout_height="wrap_content"
            android:layout_marginLeft="10dip"
            android:inputType="numberSigned"
            android:singleLine="true"
            android:textColor="@color/white"
            android:textSize="15sp"/>
    </LinearLayout>
</LinearLayout>
```

确保以上每一步都编译通过：在 IDE 集成开发环境中，没有出现红叉的错误提示。

应用程序的运行效果如图 4-18-1 所示。

点击“空气温湿度阀值设置”左箭头，运行效果如图 4-18-2 所示。

图 4-18-1　应用程序运行效果图

图 4-18-2　“空气温湿度阀值设置”界面

智能农业实战项目完整源代码中的核心代码片段如下：

```
//设置自动控制模式
if(isSetAutoControl){
    jsonObj.put("controlAuto", controlAuto);
}
//设置 CO2 浓度最小值和最大值
if(isSetCo2){
    jsonObj.put("minCo2", minCo2);
    jsonObj.put("maxCo2", maxCo2);
}
//设置灯光强度最小值和最大值
if(isSetLight){
    jsonObj.put("minLight", minLight);
    jsonObj.put("maxLight", maxLight);
}
if(isSetAir){
    //设置空气温度最小值和最大值
    jsonObj.put("minAirTemperature", minAirTemperature);
    jsonObj.put("maxAirTemperature", maxAirTemperature);
    //设置空气湿度最小值和最大值
    jsonObj.put("minAirHumidity", minAirHumidity);
    jsonObj.put("maxAirHumidity", maxAirHumidity);
}
if(isSetSoil){
```

```
        //设置土壤温度最小值和最大值
        jsonObj.put("minSoilTemperature", minSoilTemperature);
        jsonObj.put("maxSoilTemperature", maxSoilTemperature);
        //设置土壤湿度最小值和最大值
        jsonObj.put("minSoilHumidity", minSoilHumidity);
        jsonObj.put("maxSoilHumidity", maxSoilHumidity);
    }
    return jsonObj.toString();
    //处理设置告警范围请求
    ServerApp app=(ServerApp)context;
    JSONObject jsonRequest=new JSONObject(param);
    //空气温度最小值
    if(jsonRequest.has("minAirTemperature")){
        app.setMinAirTemperature(jsonRequest.getInt("minAirTemperature"));
    }
    //空气温度最大值
    if(jsonRequest.has("maxAirTemperature")){
        app.setMaxAirTemperature(jsonRequest.getInt("maxAirTemperature"));
    }
    //空气温度最大值
    if(jsonRequest.has("minAirHumidity")){
        app.setMinAirHumidity(jsonRequest.getInt("minAirHumidity"));
    }
    //空气湿度最大值
    if(jsonRequest.has("maxAirHumidity")){
        app.setMaxAirHumidity(jsonRequest.getInt("maxAirHumidity"));
    }
    //土壤温度最小值
    if(jsonRequest.has("minSoilTemperature")){
        app.setMinEarthTemperature(jsonRequest.getInt("minSoilTemperature"));
    }
    //土壤温度最大值
    if(jsonRequest.has("maxSoilTemperature")){
        app.setMaxEarthTemperature(jsonRequest.getInt("maxSoilTemperature"));
    }
    //土壤湿度最小值
    if(jsonRequest.has("minSoilHumidity")){
        app.setMinEarthHumidity(jsonRequest.getInt("minSoilHumidity"));
    }
    //土壤湿度最大值
    if(jsonRequest.has("maxSoilHumidity")){
        app.setMaxEarthHumidity(jsonRequest.getInt("maxSoilHumidity"));
    }
```

```
//灯光强度最小值
if(jsonRequest.has("minLight")){
    app.setMinLight(jsonRequest.getInt("minLight"));
}
//灯光强度最大值
if(jsonRequest.has("maxLight")){
    app.setMaxLight(jsonRequest.getInt("maxLight"));
}
//CO2 浓度最小值
if(jsonRequest.has("minCo2")){
    app.setMinCo2(jsonRequest.getInt("minCo2"));
}
//CO2 浓度最大值
if(jsonRequest.has("maxCo2")){
    app.setMaxCo2(jsonRequest.getInt("maxCo2"));
}
//警告控制,0 表示手动控制,1 表示自动控制
if(jsonRequest.has("controlAuto")){
    app.setControlAuto(jsonRequest.getInt("controlAuto"));
}
//返回结果
jsonResponse.put("result", "ok");
return jsonResponse.toString();
```

开发工程师之间两两交叉代码评审，并根据评审纪要重构代码。

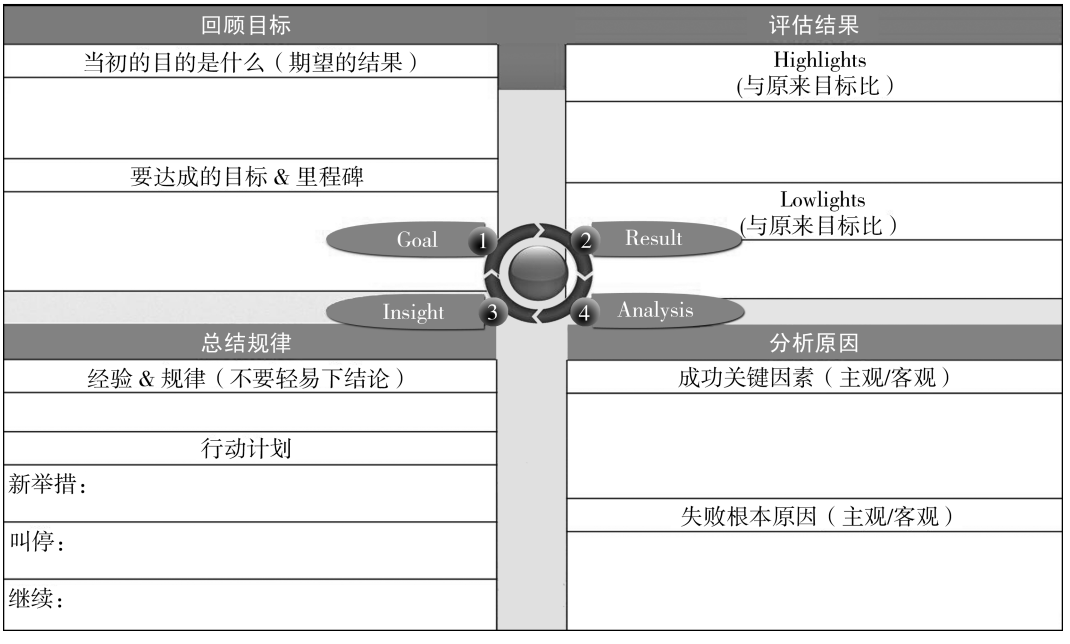
所有团队成员召开评审会，并填写表 4-18-3。

表 4-18-3　　评审建议表

序号	建议描述	是否接纳	原因
1		□接纳　□部分接纳　□未接纳	
2		□接纳　□部分接纳　□未接纳	
3		□接纳　□部分接纳　□未接纳	
4		□接纳　□部分接纳　□未接纳	
5		□接纳　□部分接纳　□未接纳	
6		□接纳　□部分接纳　□未接纳	
7		□接纳　□部分接纳　□未接纳	
8		□接纳　□部分接纳　□未接纳	
9		□接纳　□部分接纳　□未接纳	
10		□接纳　□部分接纳　□未接纳	

所有团队成员召开复盘会，并填写表 4-18-4。

表 4-18-4　　复盘会总结表



同学们，根据塔克曼阶梯理论，你们的团队目前处于哪个阶段呢？如果还没有进入成熟阶段，那就让我们一起努力吧，在接下来的任务中，更加注重相互配合和协作，及时发现并改正团队配合过程中出现的问题，早日让我们的团队进入成熟阶段！

4.18.5　任务扩展(Extend)

课后练习：使用 PreferencesActivity 或 PreferencesFragment 实现系统设置布局。

子任务 19　实现系统配置项设置

- **子任务目标：**
  - 应用系统配置项各个参数的设置
- **课时分配：**
  - 4 课时

4.19.1　任务构思(Conceive)

在智能农业项目中，要对各个环境指标的阀值进行设置，从而将各环境指标的当前值与设置的阀值进行比较，当智能农业沙盘中的传感器检测到的环境指标不在设置的阀值范围内时，客户端的 App 要产生告警通知，并且自动打开/关闭沙盘上的受控设备。

4.19.2　任务设计(Design)

PreferencesActivity 和 PreferencesFragment 两个类是实现系统设置最常用的两个类,可以对多种类型的系统参数进行设置,因此直接使用系统提供的这两个类即可。

4.19.3　任务实现(Implement)

智能农业实战项目的该子任务包含的角色为开发工程师。角色扮演的过程如表4-19-1所示。

表 4-19-1　　角色扮演过程

步骤	角色	工作	输出
1	开发工程师	编码实现子任务的业务功能	实现子任务业务功能的 Android 源代码
2	开发工程师	两两交叉代码评审	代码评审纪要
3	开发工程师	根据代码评审纪要重构代码	重构后的 Android 源代码

4.19.4　任务运作(Operate)

所有团队成员召开计划会议,并将会议纪要记录在表 4-19-2 中。

表 4-19-2　　会议纪要记录表

5W2H	结论
WHAT	
HOW	
WHY	
WHEN	
WHERE	
WHO	
HOW MUCH	

在计划会议结束之前,所有团队成员把子任务进一步细分为更小的任务,把这些任务写在便签上,并把便签贴在看板上的“ToDo”列。

所有团队成员每天早上召开 15 分钟的站立会议,每人汇报三个问题,并更新看板上的“ToDo”、“Doing”和“Done”三列。

所有团队成员相互协作,编码实现子任务的业务功能,详细的示例代码请见 Android 项目:ImplementSystemPreference,参考代码如下:

(1)创建 XML 文件:Preference. xml

```
<? xml version= "1.0"encoding= "UTF-8"? >
<PreferenceScreen xmlns:android= "http://schemas.android.com/apk/res/android"
```

```
    android:title="Settings">
    <PreferenceCategory android:title="空气温度阀值设置">
        <EditTextPreference
            android:dialogTitle="请输入空气温度的最小值"
            android:key="edit_0"
            android:summary="点击输入"
            android:title="最小值"/>

        <EditTextPreference
            android:dialogTitle="请输入空气温度的最大值"
            android:key="edit_1"
            android:summary="点击输入"
            android:title="最大值"/>
    </PreferenceCategory>

    <PreferenceCategory android:title="空气湿度阀值设置">
        <EditTextPreference
            android:dialogTitle="请输入空气湿度的最小值"
            android:key="edit_2"
            android:summary="点击输入"
            android:title="最小值"/>
        <EditTextPreference
            android:dialogTitle="请输入空气湿度的最大值"
            android:key="edit_3"
            android:summary="点击输入"
            android:title="最大值"/>
    </PreferenceCategory>
</PreferenceScreen>
```

(2)创建 java 文件:Preference.java

```
public class AndroidPreferenceDemoII extends PreferenceActivity {

    Context mContext=null ;

    @SuppressWarnings("deprecation")
        @Override
            protected void onCreate(Bundle savedInstanceState) {
                super .onCreate(savedInstanceState);
                //从资源文件中添 Preferences ,选择的值将会自动保存到 SharePreferences
                addPreferencesFromResource(R.xml.preferences);

                mContext =this ;
```

```
        // EditTextPreference 组件
        EditTextPreference mEditText0=(EditTextPreference) findPreference("edit_0");
        EditTextPreference mEditText1=(EditTextPreference) findPreference("edit_1");
        EditTextPreference mEditText2=(EditTextPreference) findPreference("edit_2");
        EditTextPreference mEditText3=(EditTextPreference) findPreference("edit_3")

        //设置 dialog 按钮信息
        mEditText0.setPositiveButtonText("确定");
        mEditText0.setNegativeButtonText("取消");

        mEditText1.setPositiveButtonText("确定");
        mEditText1.setNegativeButtonText("取消");

        mEditText2.setPositiveButtonText("确定");
        mEditText2.setNegativeButtonText("取消");

        mEditText3.setPositiveButtonText("确定");
        mEditText3.setNegativeButtonText("取消");
    }
}
```

确保以上每一步都编译通过：在 IDE 集成开发环境中，没有出现红叉的错误提示。

应用程序的运行效果如图 4-19-1 所示。

点击空气温度阀值的最小值，对空气温度阀值的最小值进行设置，如图 4-19-2 所示。

图 4-19-1　应用程序运行效果图

图 4-19-2　设置空气温度阀值最小值

点击空气温度阀值的最大值，对空气温度阀值的最大值进行设置，如图4-19-3所示。

点击空气湿度阀值的最小值，对空气湿度阀值的最小值进行设置，如图 4-19-4 所示。

图 4-19-3　设置空气温度阀值最大值

图 4-19-4　设置空气湿度阀值最小值

点击空气湿度阀值的最大值，对空气湿度阀值的最大值进行设置，如图4-19-5所示。

图 4-19-5　设置空气湿度阀值最大值

开发工程师之间两两交叉代码评审，并根据评审纪要重构代码。

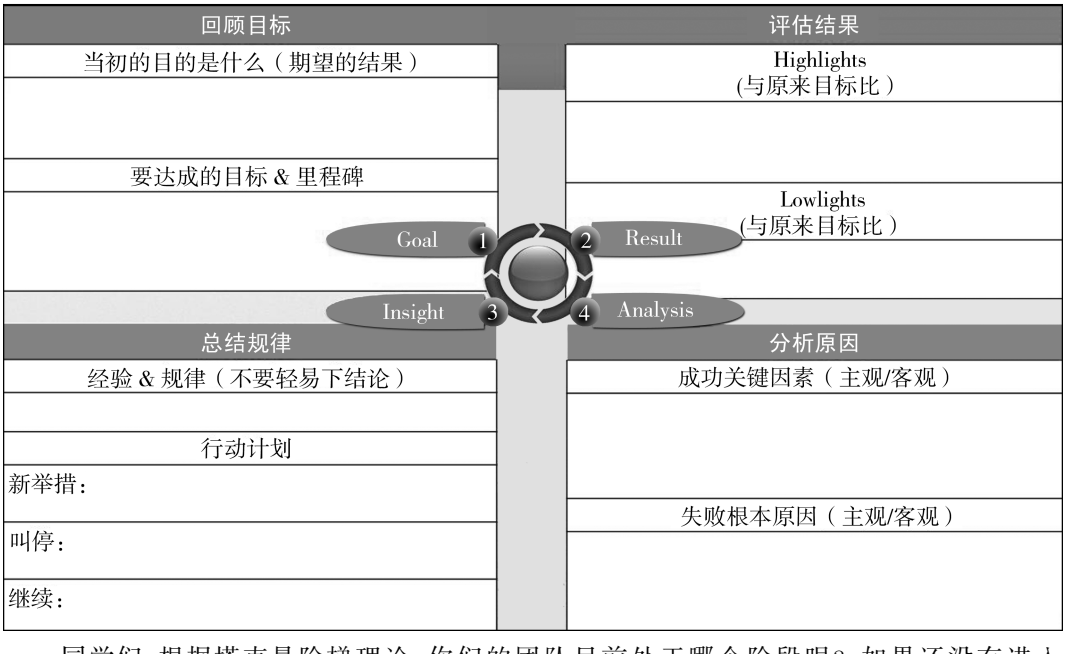
所有团队成员召开评审会，并填写表 4-19-3。

表 4-19-3　　评审建议表

序号	建议描述	是否接纳	原因
1		□接纳　□部分接纳　□未接纳	
2		□接纳　□部分接纳　□未接纳	
3		□接纳　□部分接纳　□未接纳	
4		□接纳　□部分接纳　□未接纳	
5		□接纳　□部分接纳　□未接纳	
6		□接纳　□部分接纳　□未接纳	
7		□接纳　□部分接纳　□未接纳	
8		□接纳　□部分接纳　□未接纳	
9		□接纳　□部分接纳　□未接纳	
10		□接纳　□部分接纳　□未接纳	

所有团队成员召开复盘会，并填写表 4-19-4。

表 4-19-4　　复盘会总结表



同学们，根据塔克曼阶梯理论，你们的团队目前处于哪个阶段呢？如果还没有进入成熟阶段，那就让我们一起努力吧，在接下来的任务中，更加注重相互配合和协作，及时发现并改正团队配合过程中出现的问题，早日让我们的团队进入成熟阶段！

4.19.5　任务扩展(Extend)

课后练习：在联想移动开发平台上使用 PreferencesFragment 实现系统配置项设置。