User Manual

AWFlow+AWTK 低代码应用开发指南

用户手册

V1.0.1 Date:2025/10/31

类别	内容
关键词	AWStudio、AWTK、AWFlow、低代码开发
摘要	本文介绍了使用 AWStudio 开发 AWFlow+AWTK 低代码应用的过程，只需编写少量脚本代码即可完成应用开发。

ZLG 致远电子

修订历史

版本	日期	原因
1.0.1	2022/10/08	• 根据 AWStudio1.1.34 完善文档。 • 调整文档结构。
1.0.0	2022/06/14	• first implement

目 录

文档导读

本文介绍了使用 AWStudio 开发 AWFlow+AWTK 低代码应用的过程，借助 AWFlow Designer + AWTK Designer MVVM 的开发模式可以使用拖拽、点击和连线的方式完成界面设计与业务逻辑，只需编写少量脚本代码即可完成应用开发。

1. 操作流程

本文介绍了如何使用 AWStudio 开发低代码应用的流程，借助 AWFlow + AWTK MVVM 的开发模式，通过使用拖拽、点击和连线的方式完成界面设计与业务逻辑配置，只需编写少量脚本代码，即可完成一个带业务逻辑的 GUI 应用。

整体的流程是使用 AWFlow Designer 编辑流图配置业务逻辑，使用 AWTK Designer 设计应用界面，再通过配置 MVVM 绑定这两者。

1.1 基本流程

如下图所示，AWStudio 开发 AWFlow+AWTK 应用程序主要包括以下步骤：

- 使用 AWStudio 新建工作区；
- 使用 AWStudio 新建”AWFlow Application”项目（下文简称为 AWFlow 项目）到工作区，双击打开该项目会进入 AWFlow Designer，编辑流图，调试流图；
- 使用 AWStudio 新建”AWTK MVVM JS Application”项目（下文简称为 MVVM JS 项目）到工作区，双击打开该项目会进入 AWTK Designer，导入流图，设计好界面并设置流图与界面控件的 MVVM 绑定，可以在 PC 上模拟运行；
- 使用 AWStudio 将 MVVM JS 项目和 AWFlow 流图拖拽到到目标设备上，点击下载即可将项目资源下载到设备中。

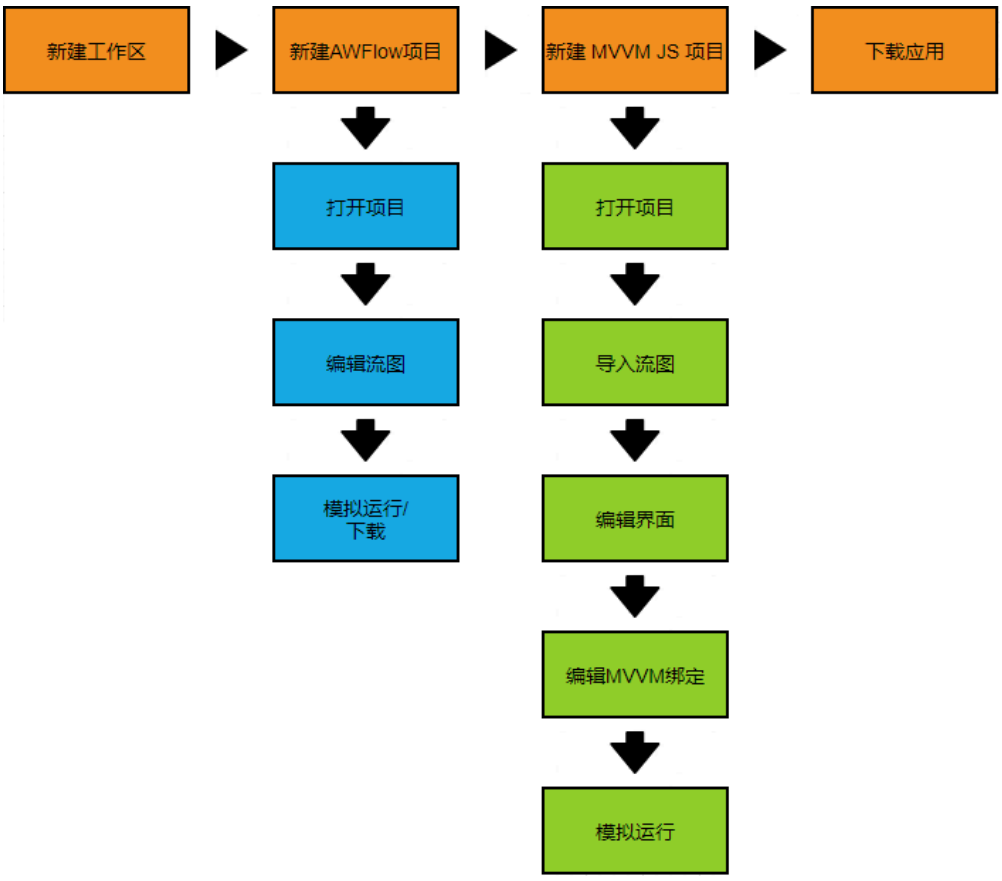

图 1.1 基本流程

1.2 新建工作区

AWStudio 以工作区的形式来管理各种项目，因此，使用 AWStudio 的第一步先新建或打开一个工作区。

AWStudio 在启动之后，会自动弹出”新建工作区”页面，如下图所示。

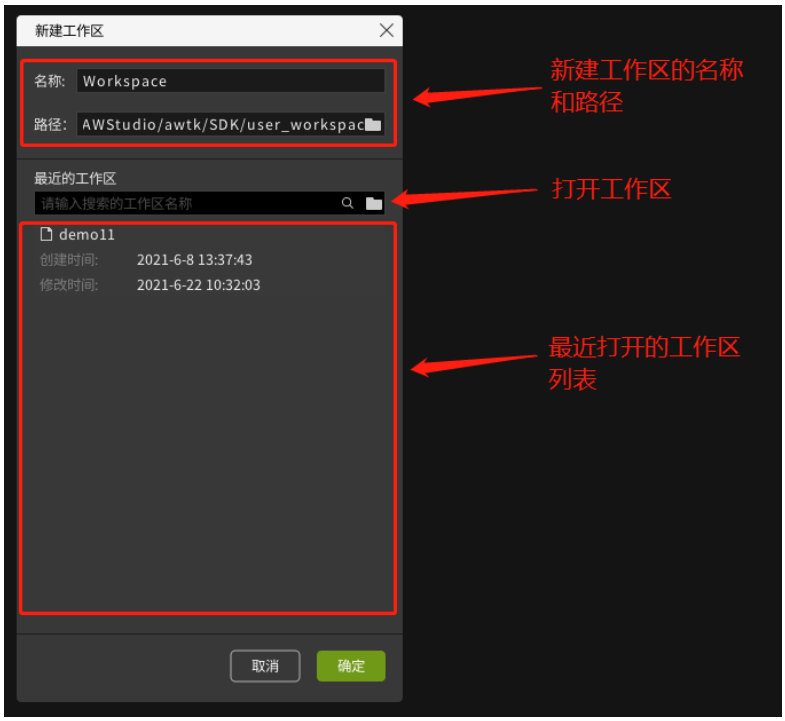

图 1.2 新建工作区页面

设置好项目名称和项目路径后，点击”确定”按钮即可新建一个工作区。打开新建的工作区后，需创建或导入项目。

1.3 新建项目

点击如下图所示的”+”按钮或”新建项目”按钮可以打开”新建项目”页面。

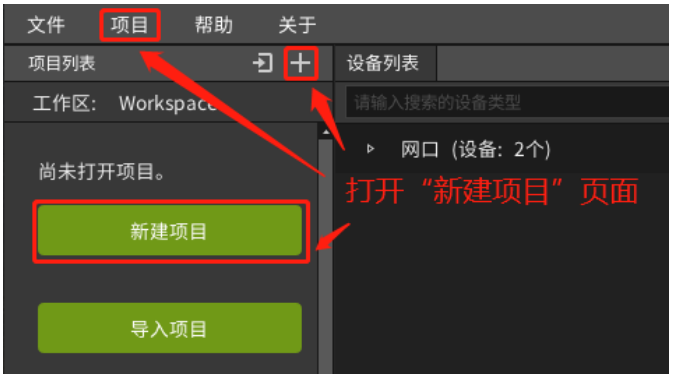

图 1.3 打开新建项目页面

1.4 打开项目

在 AWStudio 打开项目的方式有以下两种：

- 双击”项目列表”里的项目；
- 用右键点击”项目列表”里的项目，会弹出菜单，选择”打开”；

打开项目后，对应的项目编辑器会被运行，下图的标志会变为绿色的圆点，表示项目已被打开。

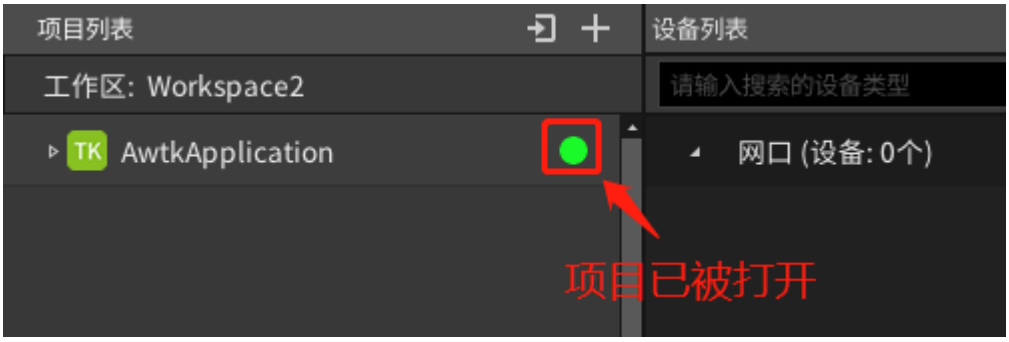

图 1.4 项目已被打开

1.5 编辑 AWFlow 项目

AWFlow 项目可以配置业务逻辑。通过编辑 AWFlow 流图，也就是加入需要的功能节点，并连线配置即可。

如下图所示，打开”新建项目”界面，选择”AWFlow Application”，新建一个 AWFlow 项目，双击工作区中的 AWFlow 项目，进入 AWFlow Designer 后可以按下列步骤进行编辑。

注: 本节只简单介绍编辑 AWFlow 项目的基本方法，更多操作请看《AWFlow Designer 用户手册》。

图 1.5 新建并打开 AWFlow 项目

1.5.1 新建 AWFlow 流图

点击 AWFlow 主界面左侧流图列表上的”+”按钮，会弹出”添加流图”页面，如下图所示。

图 1.6 添加流图页面

“添加流图”界面包括以下设置：

- 流图类型。流图类型包括以下两种：
  1. 流图。流图是 AWFlow 项目运行的基本单位，可以使用 runFlow 直接运行，也可以导入到 MVVM JS 项目，与界面绑定运行；
  2. 子流图。子流图是一种可以被嵌入其他流图里的可复用流图，它的编辑方法与流图一样，在嵌入其它流图时，可以像普通节点一样拖入使用。当编辑的流图的业务流程比较复杂的时候，子流图的灵活使用将大大优化开发效率。
- 流图名称。流图名称分为以下两种：
  1. 流图的名称用作 AWFlow 项目里的流图标识，当项目有多个流图时，可以通过设置不同的名字加以区分；
  2. 子流图的名字用作区分不同子流图的标识，也可以看作子流图节点的类型名称；
- 目标板型号。目标板型号指流图运行时的硬件平台。

在”添加流图”页面里设置”流图名称”,“流图类型”和”目标板型号”，点击”确定”按钮，可以添加并自动打开流图，如下图所示。

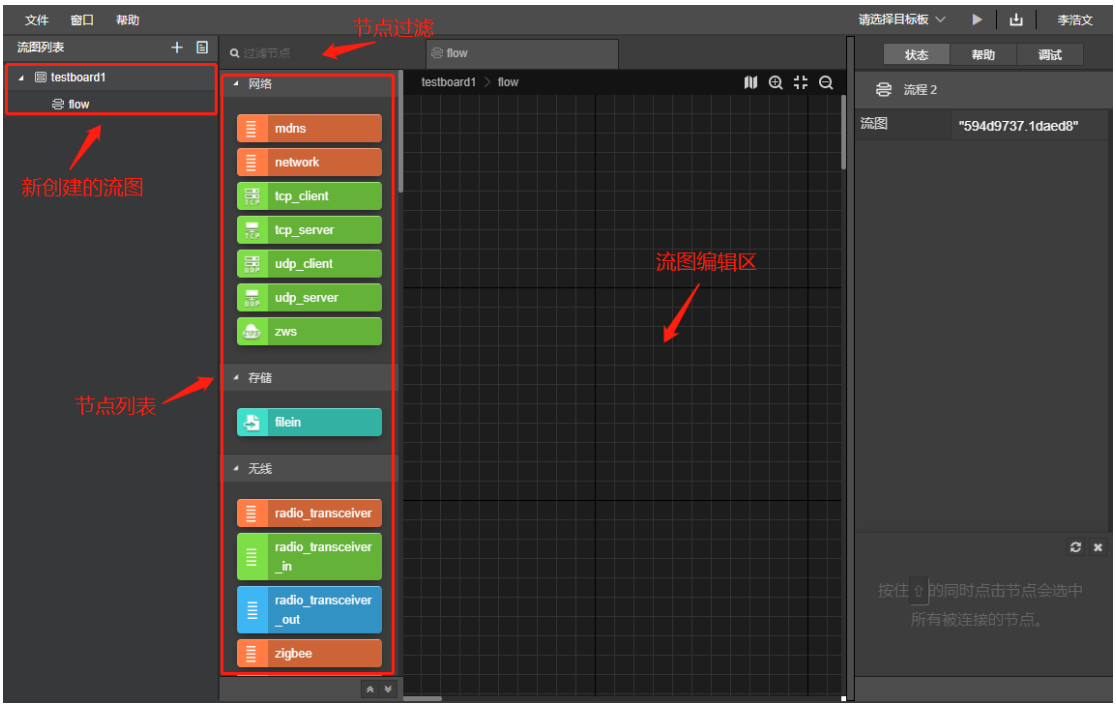

图 1.7　新建流图后的 AWFlow Designer 主界面

1.5.2　编辑 AWFlow 流图

打开流图后，可以按如下步骤进行编辑：

- 从节点列表中选中节点，拖动到编辑区；
- 双击节点，设置节点的属性；
- 点击节点输出端上的白色圆点，可以拖出一条白线，将它连接到另一节点的输入端，如下图所示。

注: AWFlow Designer 连接节点只能由输出端连接到输入端，而输出端连接输出端或者输入端连接输入端都是不合法的。

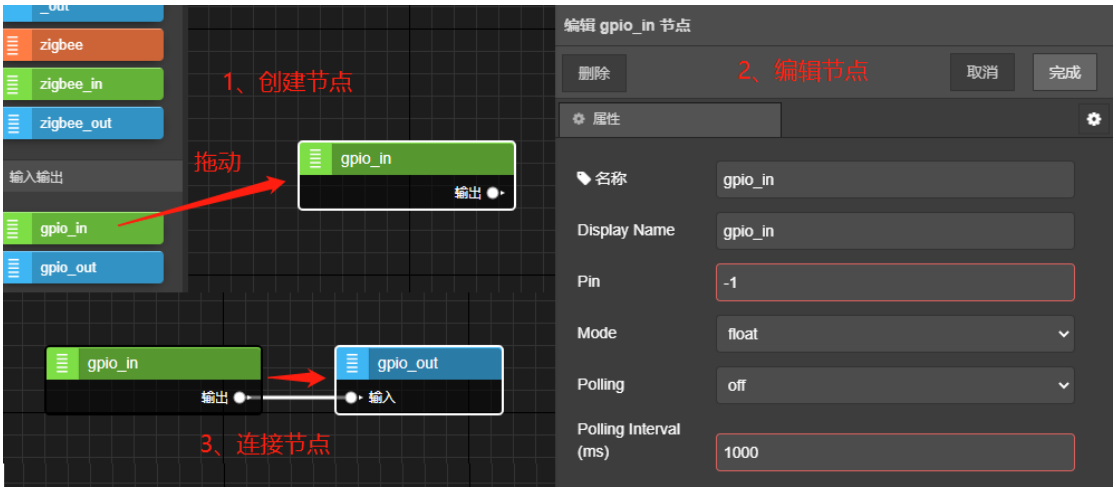

图 1.8　编辑流图的步骤

1.6 编辑 AWTK MVVM 项目

AWTK MVVM 项目可以实现应用程序的界面。设计好界面后，导入上文中提到的流图，并通过 MVVM 绑定设置将界面控件的属性、事件与流图节点的属性、命令绑定即可。

如下图所示，打开”新建项目”界面，选择”AWTK MVVM JS Application”，新建一个 MVVM 项目，双击工作区中的 MVVM 项目，进入 AWTK Designer 后可以按下列步骤进行编辑。

注：本节只简单介绍编辑 AWTK MVVM JS 项目的基本方法，更多操作请看《AWTK Designer MVVM 使用指南》。

图 1.9 打开 MVVM JS 项目

1.6.1 编辑界面

新建或打开一个窗体，开始编辑界面，编辑界面的步骤如下：

- 从控件列表中选中控件，拖动到编辑区；
- 选中控件节点，设置控件的属性，如下图所示；

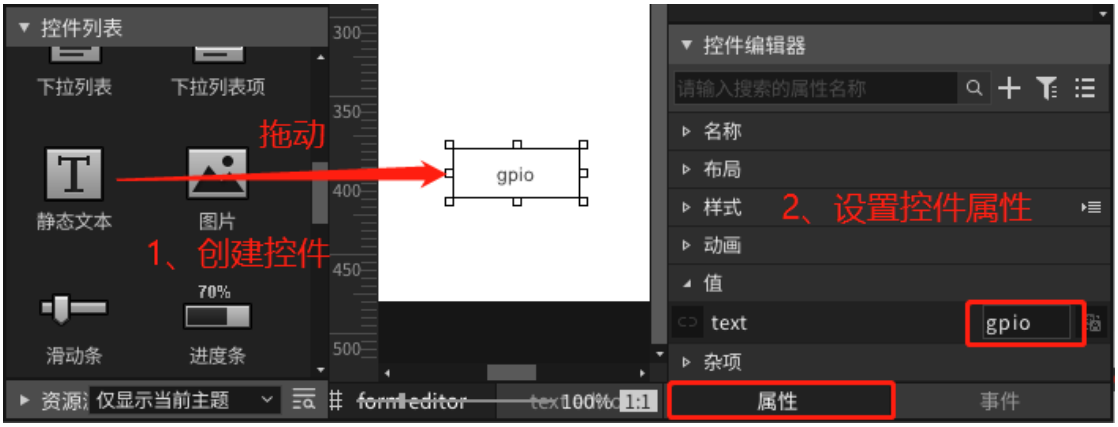

图 1.10 编辑 AWTK 界面

按照以上步骤完成界面的编辑，如下图所示。

图 1.11　MVVM JS 项目界面示例

1.6.2　新建视图模型

点击 AWTK Designer 底部的“新建模型”按钮，进入”新建视图模型”页面，可以按需更改名字或者路径，一般按照默认即可，点击”确定”按钮完成视图模型的创建，如下图所示。

图 1.12　新建视图模型

1.6.3 导入流图

点击全局对象管理器窗口上的”导入流图”按钮，进入”选择导入的流图”页面；再点击下方”打开 Flow 项目”，选择要导入的流图所在的 AWFlow 项目，在流图列表中选择要导入的流图，点击”导入”按钮可以导入流图，如下图所示。

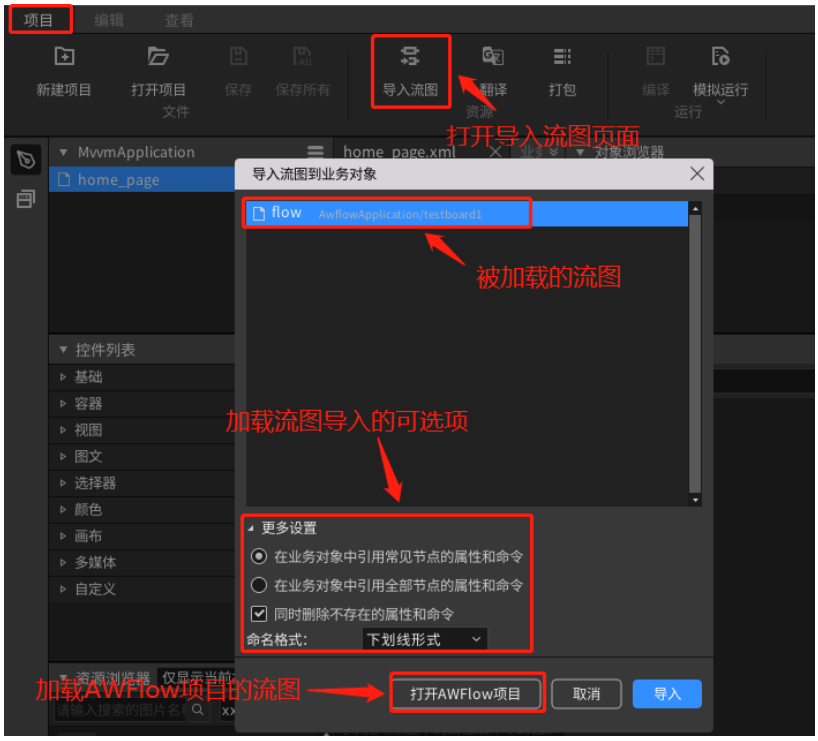

图 1.13 导入 AWFlow 流图

导入流图后，会有一个以流图名称命名的对象，比如此处的”flow”的对象，点击左侧的下拉按钮展开可以看到流图中可用于 MVVM 绑定的属性和命令，如下图所示。

图 1.14 导入 AWFlow 流图后的全局对象管理器

1.6.4 编辑 MVVM 绑定

1. 数据绑定

选中控件，在控件编辑器的属性页面上找到要绑定的属性，点击属性左侧的按钮，可以选择设置属性绑定。例如，选中一个”button”控件，将控件的”text”属性与流图里的”hold”节点的”value”属性绑定，设置如下图所示，程序运行时，”button”控件的”text”属性会被设置为”hold”节点的”value”属性的值，并随之变化而变化。

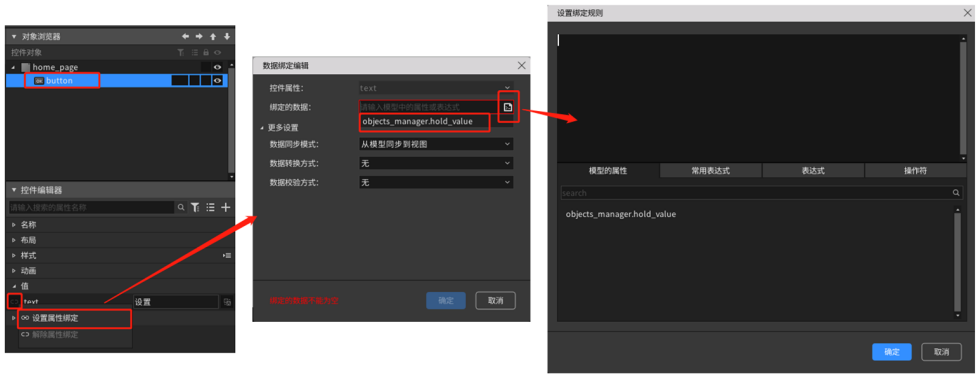

图 1.15　数据绑定

2. 命令绑定

选中控件，点击控件编辑器的事件页面上添加按钮，可以选择要添加的事件，之后设置事件动作为指定的 ViewModel 命令。例如，选中一个”button”控件，将控件的”click”（点击）事件与某个命令绑定，如下图所示，程序运行时，当”button”控件触发”click”事件时，程序执行设置的 fscript 脚本，在控制台打印”hello world”。

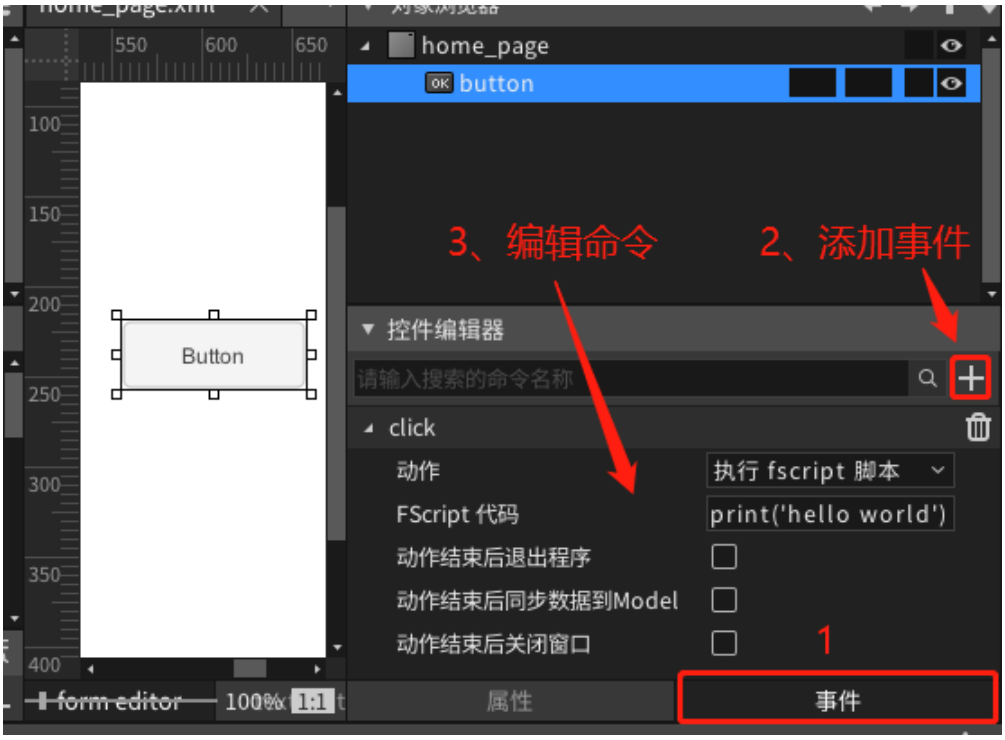

图 1.16　命令绑定

1.6.5 模拟运行

点击 AWTK Designer 工具栏上的打包按钮进行打包，之后点击模拟运行按钮，可以在 PC 上模拟运行应用，如下图所示。

图 1.17 模拟运行

1.7 下载应用

1. 连接设备

将运行有 runFlow 的开发板连接到电脑的网口，连接后可以在 Studio 的设备列表中找到该设备，如下图所示。

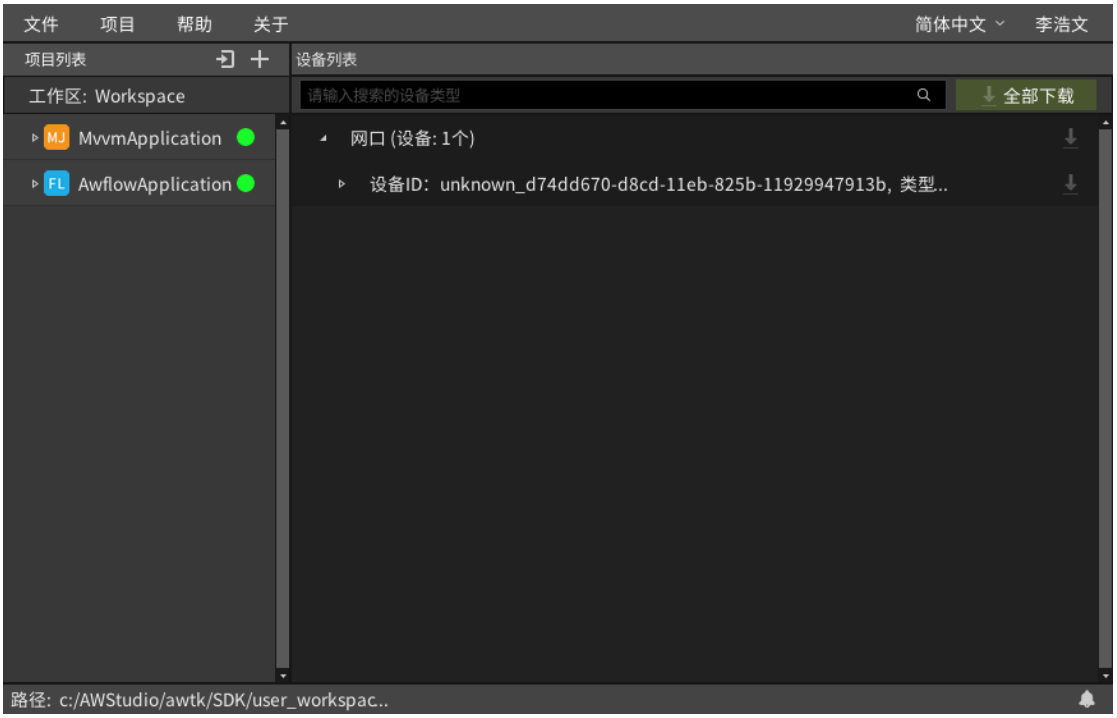

图 1.18 连接设备

2. 下载

将要 MVVM JS 项目和流图拖到设备上，点击设备右侧的”↓”按钮，可以将应用运行时的资源下载到设备，并重启设备，如下图所示。

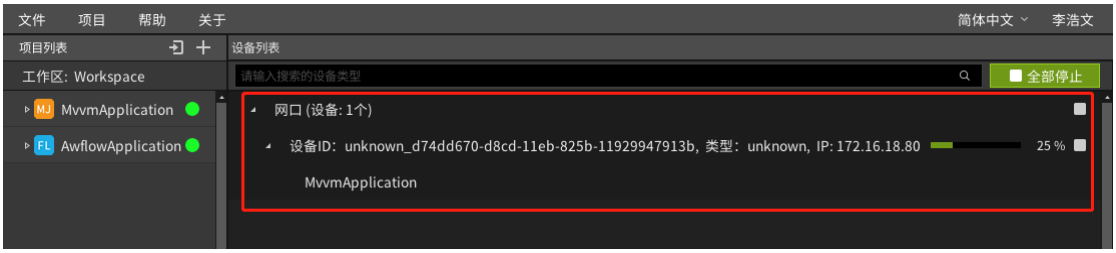

图 1.19　下载 MVVM 项目和流图

2. 基础功能案例

根据上文中的说明，我们知道使用 AWStudio 开发 AWFlow+AWTK 低代码应用的基本操作如下：

- 使用 AWFlow Designer 编辑流图实现程序业务逻辑;
- 使用 AWTK Designer 编辑 UI 文件实现程序界面设计。

本章将具体介绍在完成以上步骤后，如何将程序的业务逻辑和界面关联起来，即怎么实现二者之间的数据交互，并且还将介绍致远电子推出的物联网 IoT 云平台 ZWS 云[1]，演示如何实现设备和云端之间的数据交互。

2.1 流图数据传送到界面上显示

此处我们假设需要实现一个空调控制器，需要将当前的温度值显示到 GUI 界面上。

首先使用 AWStudio 新建一个工作区，并在工作区中分别新建一个 AWFlow 项目和一个 AWTK MVVM-JS 项目。其中 AWFlow 流图负责实现采集温度数据逻辑，再将数据传送到 AWTK GUI 界面上显示。

将 AWFlow 流图中的数据显示到界面上需要借助流图中的**锁存器**（即”hold”节点），AWTK GUI 可以读取该节点的数据并显示出来（通过 AWTK MVVM 机制将数据与控件进行绑定）。

2.1.1 定时随机生成一个温度值

为了方便演示，这里定时随机生成一个温度值，用于后续显示到界面上。首先在工作区中双击打开 AWFlow 项目，进入 AWFlow Desigenr 后，新建一张流图。

步骤一：将”节点列表”中”timer”类型的节点拖到流图编辑区，并将”定时周期”属性设置为 1000 ms，每隔一秒触发一次，如下图所示：

图 2.1 添加定时器

步骤二：将”节点列表”中的”random_number”类型的节点拖到流图编辑区，其输入口与上文中创建的”timer”节点相连，设置如下：

- 将生成的随机数名称（’‘主题’’属性）设置为”temperature”;
- 将随机数的范围限制在 0 到 40 之前;

[1] https://www.zlgcloud.com

图 2.2　生成随机温度值

2.1.2　使用锁存器保存温度值

上一小节中我们定时生成了一个随机数作为温度值，接下来使用锁存器（”hold”节点）将其保存起来，步骤如下：

步骤一：将”节点列表”中”hold”类型的节点拖到流图编辑区，设置如下：

- 设置”名称”属性为”temperature”。
- 设置”显示名称”属性为”temperature”。
- 设置需要保存的数据”Key”为上一个节点（random_number 节点）传输过来的”payload”。

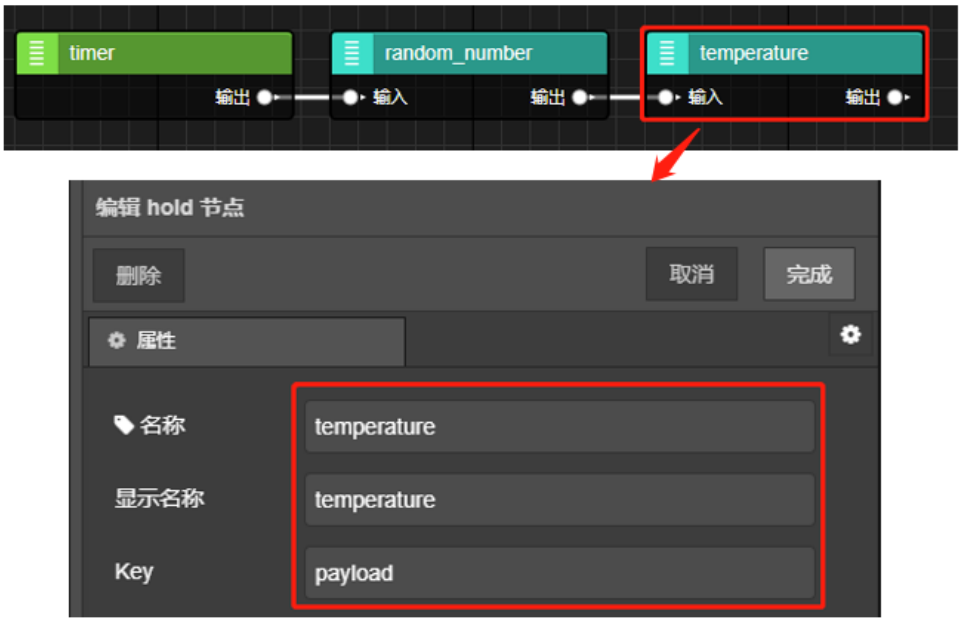

图 2.3　使用锁存器保存温度值

2.1.3 将温度值绑定到界面上显示

步骤一：在工作区中双击打开 AWTK MVVM-JS 项目，进入 AWTK Desigenr 后，默认打开主页 home_page，为其创建一个 ViewModel，如下图所示：

图 2.4 新建 ViewModel

步骤二：参考本文第一章的内容导入我们上一小节中编辑好的流图，如下图所示：

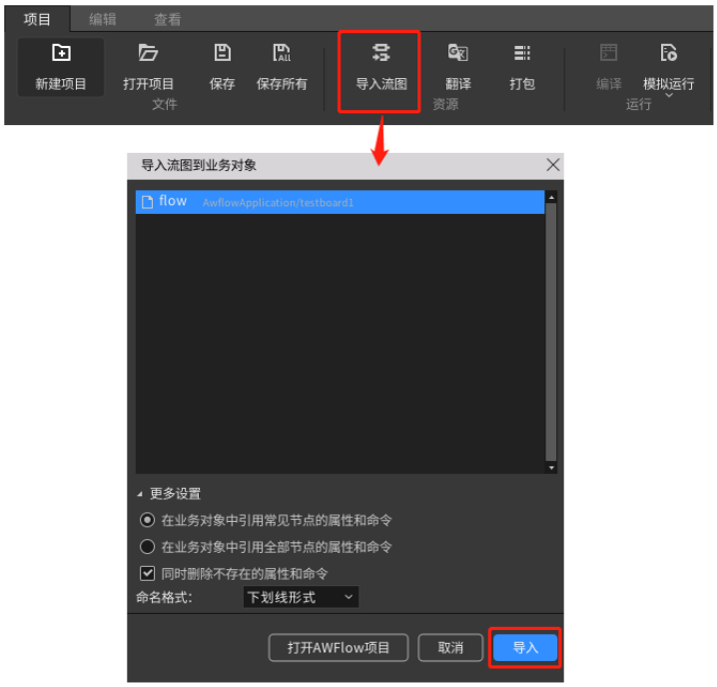

图 2.5　导入流图

步骤三：在主页 home_page 中创建一个 label 控件，如下图所示：

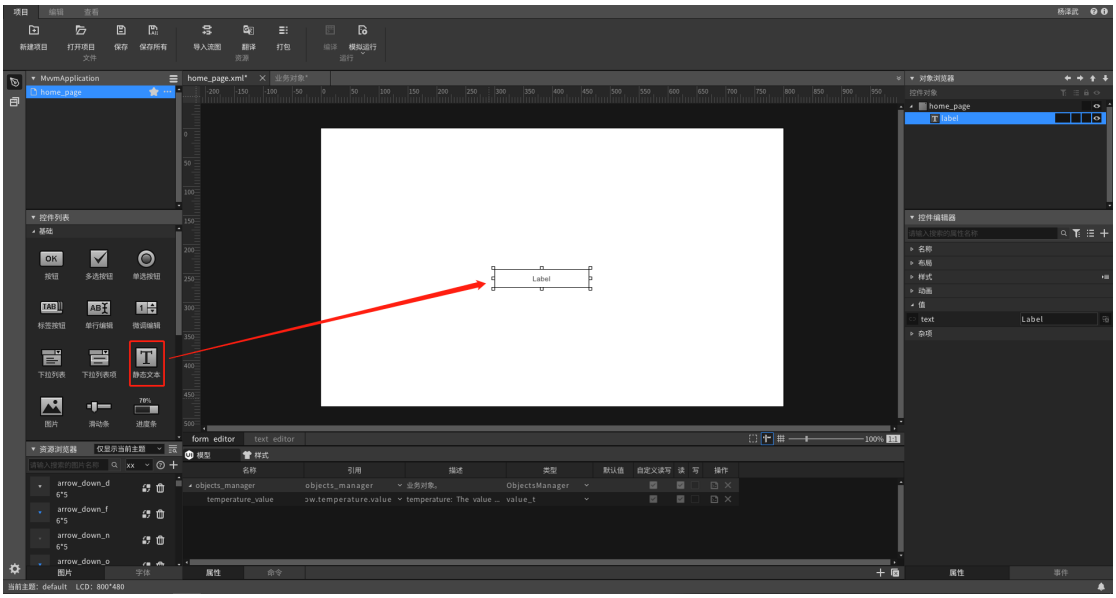

图 2.6　创建 label 控件

步骤四：将 label 控件的 text 属性与锁存器（”hold”节点）中保存的”temperature”温度值绑定起来，操作如下图所示，只需在”绑定的数据“一栏选择对应 flow 的节点：

```
/* 此处为温度值的访问路径：全局模型对象.hole节点.value值 */
/* > 备注：hole节点的名称为temperature */
objects_manager.temperature.value
```

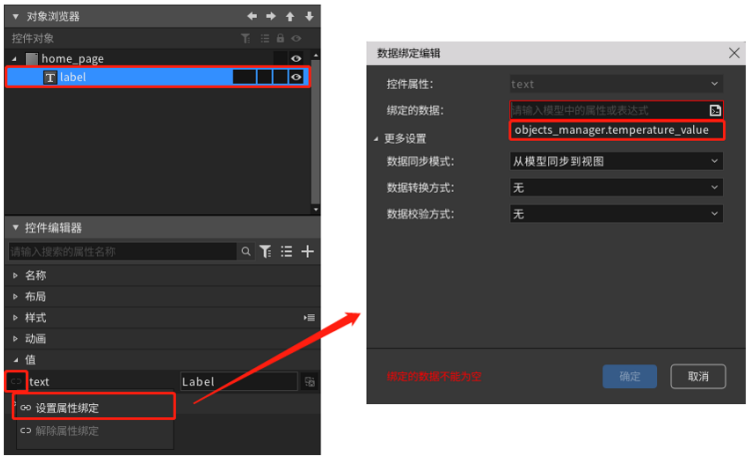

图 2.7　绑定温度值

2.1.4　运行效果

完成以上操作后，就把流图中的温度值绑定到界面的 label 控件上了，当温度值更新时会刷新到界面上。

我们可以通过 AWTK Designer 中的”模拟运行”来查看效果，如下图所示：

图 2.8　点击”模拟运行”

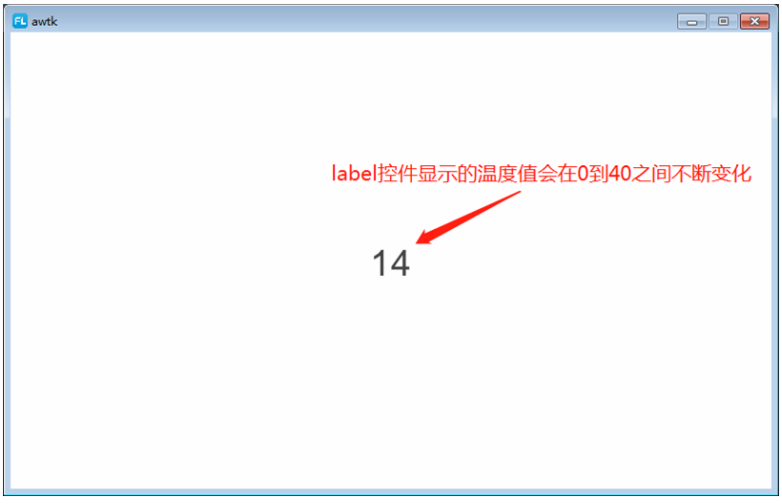

图 2.9　将流图数据显示到界面上

注: 这里为了更好得展示效果，调大了 label 控件显示文本的字号。

2.2 界面数据传送到流图

此处依旧以实现一个空调控制器为例，假设点击界面上的按钮时，将界面上编辑框中输入的温度值传送到流图。在流图中可以对温度值做进一步处理，比如控制开关启动压缩机等。

该功能需要借助 AWFlow 中的**界面事件接收器**（即”ui_input”节点），它主要用于接收 AWTK GUI 界面中的事件输入，进而触发后续的流图逻辑运行。

2.2.1 添加界面事件接收器

打开 AWFlow Designer，将”节点列表”中”ui_input”类型的节点拖到流图编辑区，然后将节点的”名称”属性和”显示名称“属性都设置为”set_temperature”，如下图所示：

图 2.10 添加界面事件接收器

2.2.2 在界面中绑定事件并发送数据

步骤一：打开 AWTK Designer，在主页 home_page 中创建一个 edit 控件和一个 button 控件，并调整它们的字体大小，方便后续演示，如下图所示：

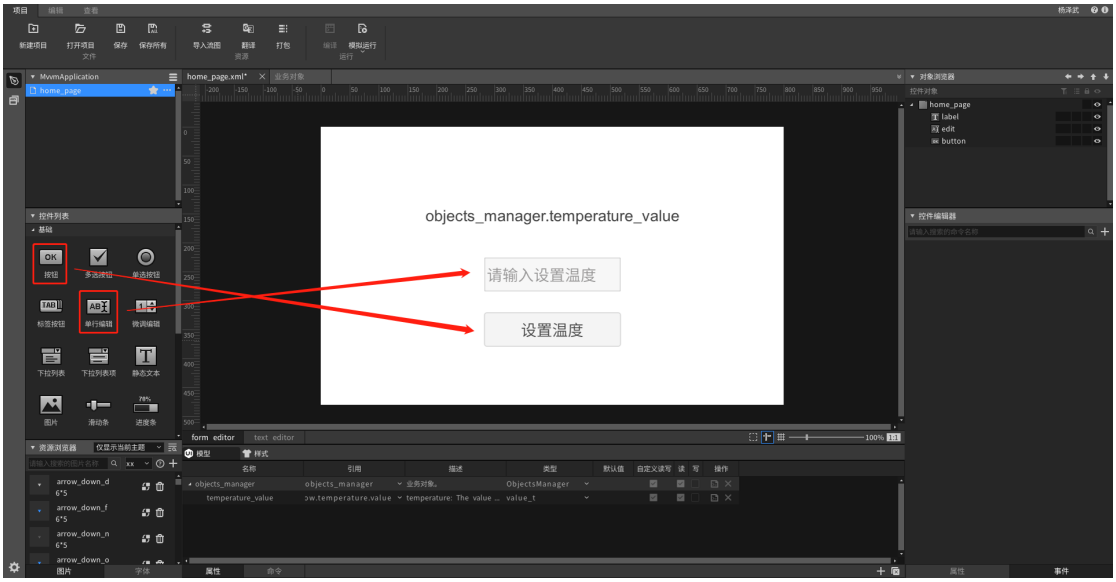

图 2.11 创建 edit 控件和 button 控件

步骤二：选中 button 控件，在控件编辑器中切换到”事件”页面，点击右上角的”+”按钮，为其添加一个点击（click）事件，事件设置如下：

- 设置”动作”为”执行视图模型中的命令”。
- 设置”命令的名称”为”objects_manager.set_temperature.run”，即触发上一小节在流图中添加“ui_input”节点（名称为”set_temperature”）。
- 设置”命令的参数”为 edit 控件中的文本，操作如下图所示：

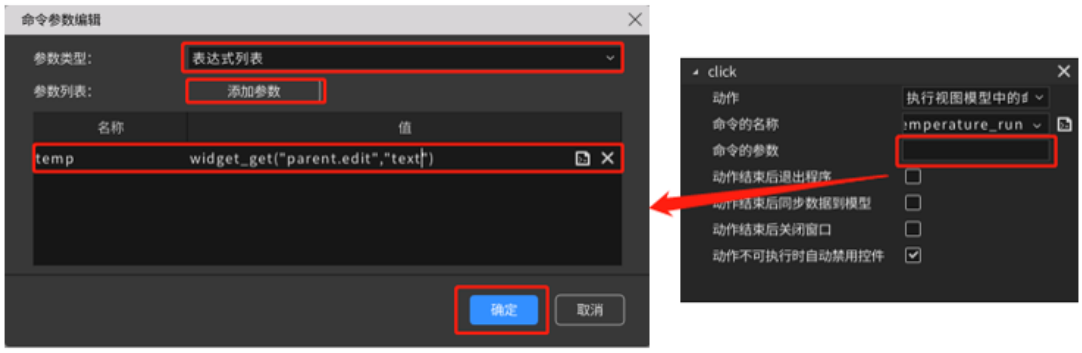

图 2.12　设置命令的参数

点击”命令的参数”输入框，打开”命令参数编辑”页面，更改’‘参数类’‘为’‘表达式列表’’，点击“添加参数”，在“名称”与“值”一栏分别填写对应名称与表达式，点击“确定”按钮完成编辑。

注:

1. AWTK-MVVM 命令传参的相关介绍详见：awtk-mvvm/docs/11.command_binding.md[2];
2. FScript 脚本的介绍与具体用法详见：awtk/docs/fscript.md[3]。

2.2.3　处理流图收到的温度数据

完成以上步骤后，模拟运行时点击程序中的按钮应该就会触发流图中对应名称的”ui_input”节点，根据 AWTK-MVVM 的命令传参规定[4]，命令的入参为字符串类型的，即”ui_input”节点收到的参数形式如下：

```
/* 此处在edit控件中输入33，再点击button控件，uiinput节点收到的参数如下 */
string?temp=33
```

注: 如果传入多个参数，参数之间采用”&”符号相连，比如：string?args1=value1&args2=value2。

该数据不能直接使用，因此，我们需要使用”fscript”节点对”ui_input”节点收到的数据进行处理，步骤如下，操作如下图所示：

[2] https://github.com/zlgopen/awtk-mvvm/blob/master/docs/11.command_binding.md
[3] https://github.com/zlgopen/awtk/blob/master/docs/fscript.md
[4] https://github.com/zlgopen/awtk-mvvm/blob/master/docs/11.command_binding.md

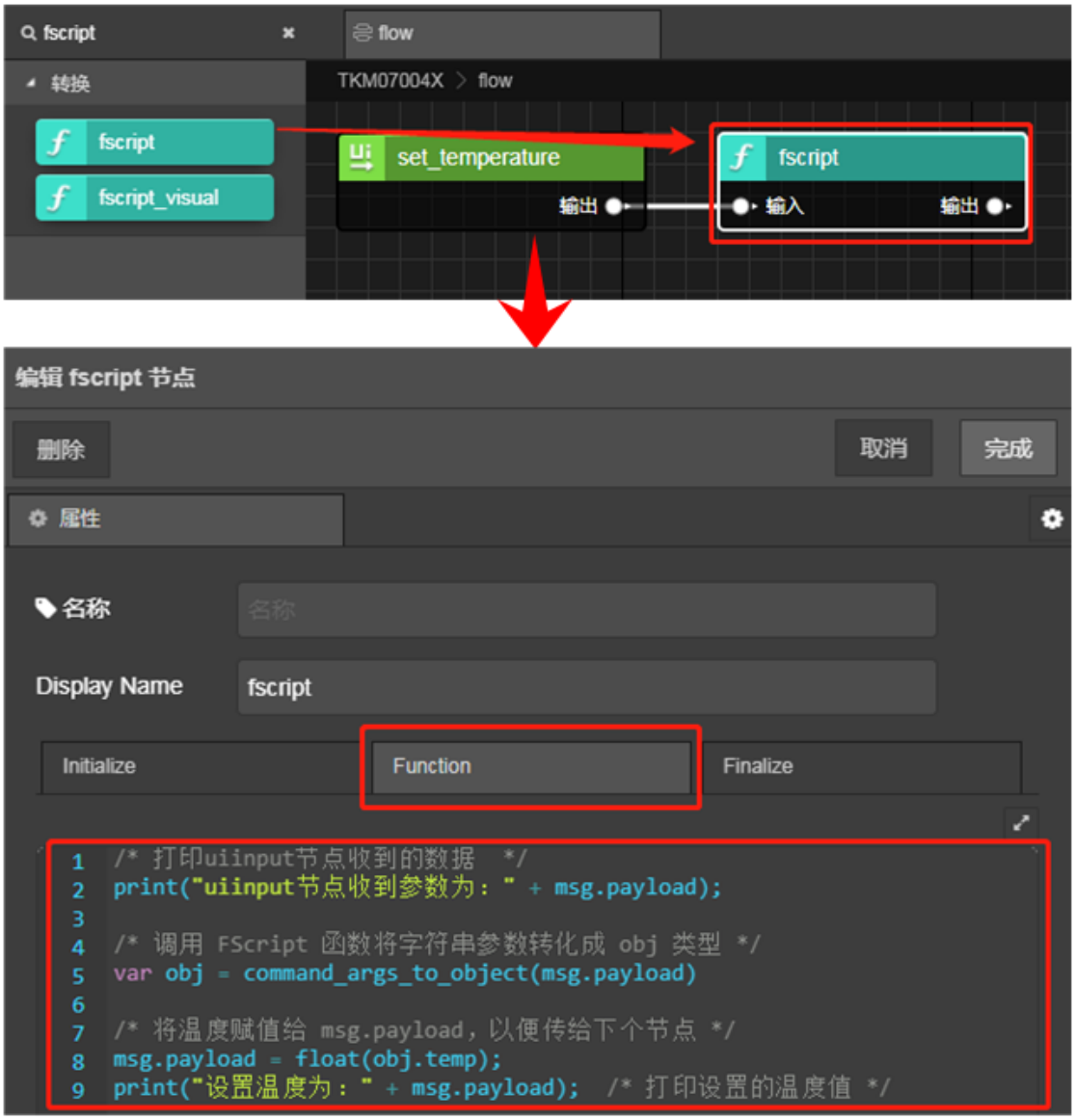

图 2.13　处理流图收到的温度数据

步骤一：打开 AWFlow Designer，将”节点列表”中”fscript”类型的节点拖到流图编辑区，然后将其与”ui_input”节点相连。

步骤二：编辑”fscript”，切换到”Function”页面，添加数据处理脚本，代码如下：

```
/* 打印uiinput节点收到的数据 */
print("uiinput节点收到参数为：" + msg.payload);

/* 调用 FScript 函数将字符串参数转化成 obj 类型 */
var obj = command_args_to_object(msg.payload)

/* 将温度赋值给 msg.payload，以便传给下个节点 */
msg.payload = float(obj.temp);
print("设置温度为：" + msg.payload); /* 打印设置的温度值 */
```

注: FScript 脚本中字符串处理的相关函数请参考：awtk/docs/fscript_str.md[5]。

[5] http://gitlab/zlgopen/awtk/blob/master/docs/fscript_str.md

2.2.4 运行效果

在 AWTK Designer 中点击”模拟运行”启动程序，然后在 edit 控件中输入温度 33，点击”设置温度”按钮，可以看到控制台中的输出，如下图所示：

图 2.14 将界面数据保存到流图中

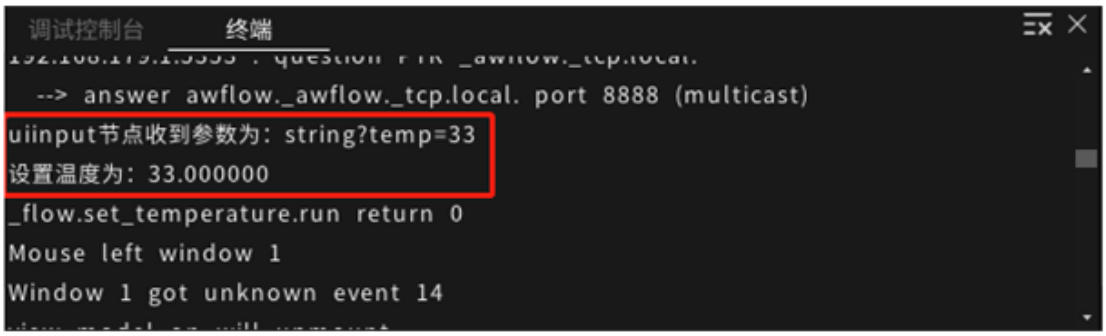

图 2.15 控制台输出

2.3 流图数据控制界面动画启停

此处简单地以一个按钮控制图片控件位移动画为例，实现的效果为：点击按钮时，将流图中控制动画的数据取反，界面根据流图中的数据开启或暂停动画，并在界面中显示数据的信息。

在 MVVM 中实现对动画的控制，主要是通过指定控件（widget_t）中的“exec”来执行特定的函数，关于“exec”具体的使用方法可以参考 《AWTK Designer MVVM 使用指南》[6]中 MVVM 控制动画启停章节。

[6] https://awtk.zlg.cn/pro/docs/awstudio_docs/AWTK_Designer_MVVM_Use_Guide/

2.3.1 添加界面事件接收器

打开 AWFlow Designer，将”节点列表”中”ui_input”类型的节点拖到流图编辑区，然后将节点的”名称”属性和”显示名称“属性都设置为”button_click”，如下图所示：

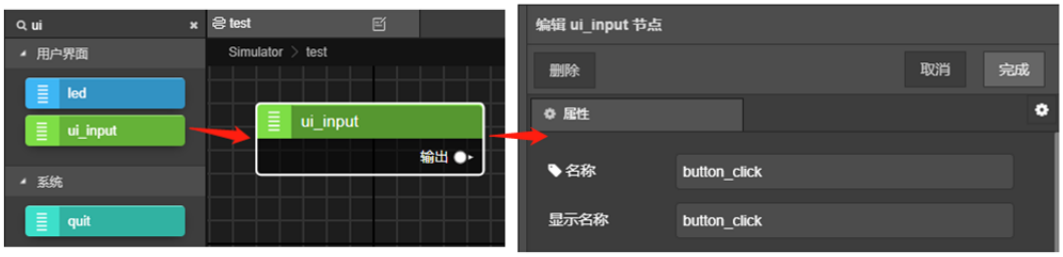

图 2.16 添加界面事件接收器

2.3.2 改变控制动画启停的数据

本案例中，我们通过改变一个布尔类型的变量值来控制动画的启停。因此，需要使用“fscript”节点对数据进行更改，操作如下图所示：

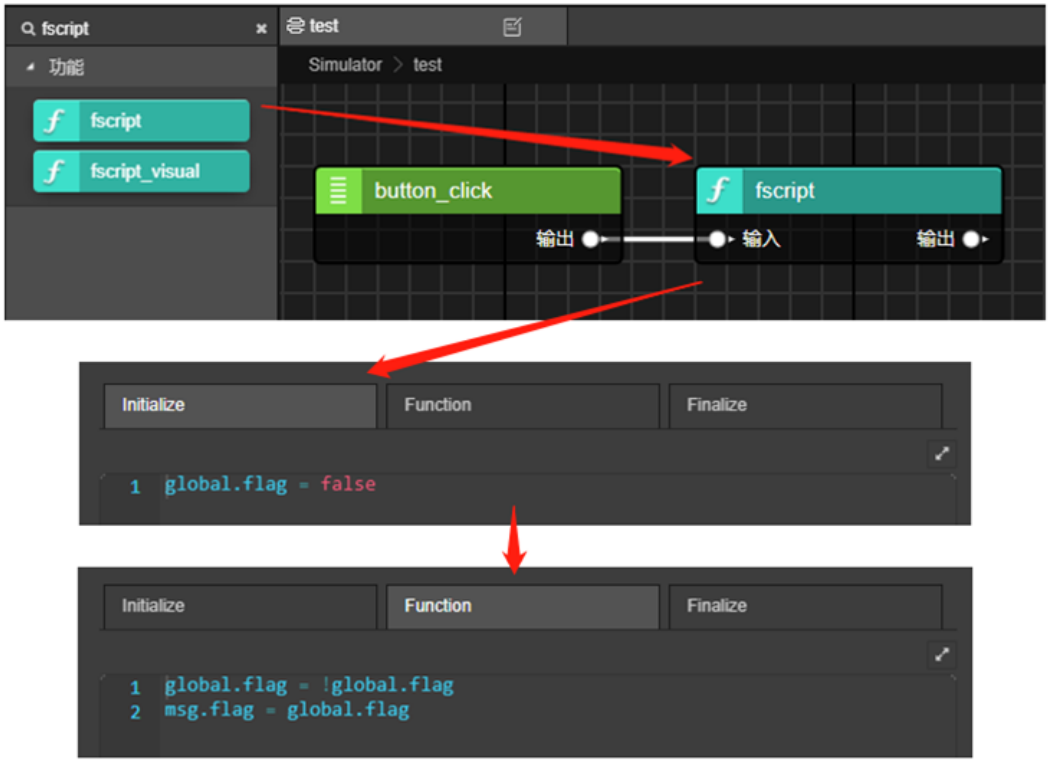

图 2.17 通过 fscript 节点修改数据

步骤一：将”节点列表”中”fscript”类型的节点拖到流图编辑区，然后将其与”ui_input”节点相连。

步骤二：编辑”fscript”，切换到”Initialize”页面，进行数据初始化，代码如下：

```
/* 初始化 global.flag 为 false */
global.flag = false
```

步骤三：切换到”Function”页面，添加数据处理脚本，代码如下：

```
/* 对 global.flag 进行取反 */
global.flag = !global.flag
/* 将 global.flag 的值赋给 msg.flag */
msg.flag = global.flag
```

2.3.3 使用锁存器保存数据

上一节中，我们将 global.flag 的数据保存到了 msg.flag 中，接下来使用锁存器（” hold”节点）将该数据保存起来，操作步骤如下：

将”节点列表”中”hold”类型的节点拖到流图编辑区，设置如下：

- 设置”名称”属性为”animate_flag”。
- 设置”显示名称”属性为”animate_flag”。
- 设置需要保存的数据”Key”为上一个节点（fscript 节点）传输过来的”flag”。

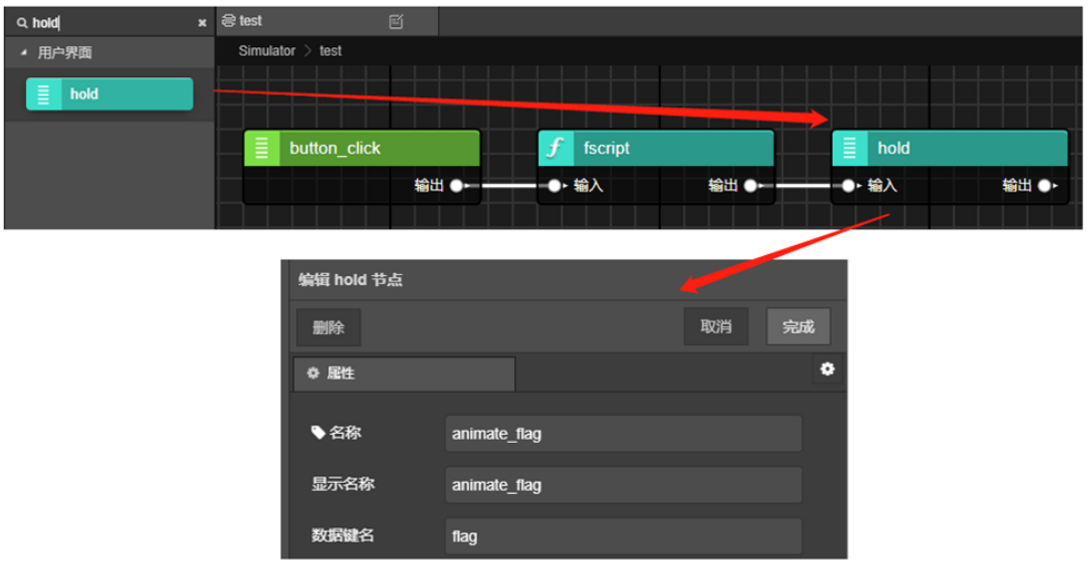

图 2.18 保存数据

2.3.4 界面设计

在本案例中，我们需要使用到的控件分别有:button、image 和 label，它们的作用分别是，控制动画启停，展示动画和显示控制动画的变量。将它们拖拽到界面编辑器上并进行布局，如下图所示：

图 2.19　界面设计

2.3.5　为 image 控件创建动画与数据绑定

首先为 image 控件创建控件动画，操作步骤如下所示：

步骤一：选中 image 控件并在控件编辑器中的”动画”一栏点击”+”按钮；

步骤二：设置动画的参数，完成后点击确定创建动画。

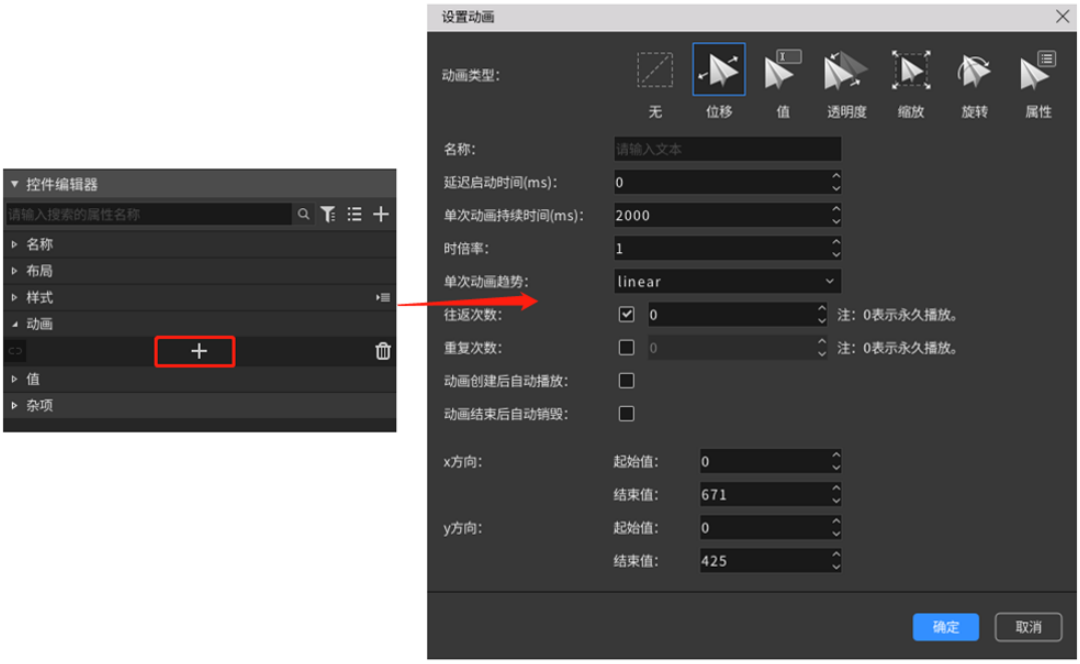

图 2.20　创建控件动画

创建完动画后，我们需要为 image 控件添加一个“exec”属性并对其进行属性绑定来控制动画的启停，操作步骤如下：

步骤一：为 image 控件添加“exec”属性；

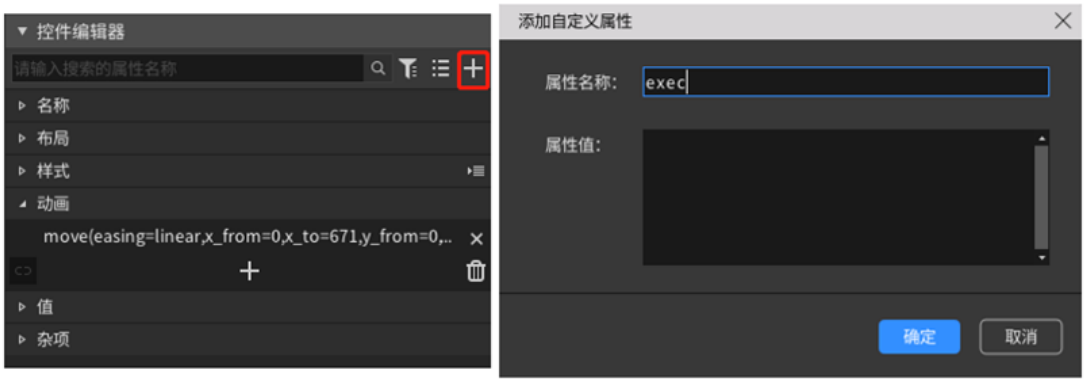

图 2.21　添加”exec”属性

步骤二：为“exec”属性设置属性绑定，绑定规则为：

```
objects_manager.hold_value ? 'start_animator' : 'pause_animator'
```

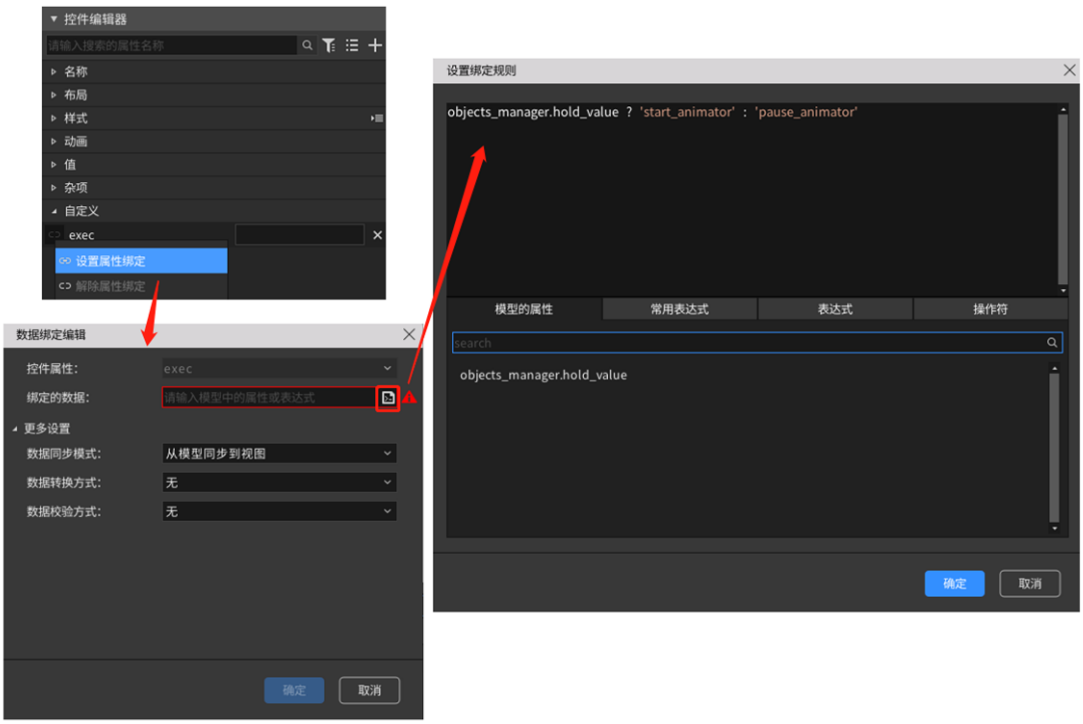

图 2.22　设置属性绑定

2.3.6 为 button 控件绑定命令

为 button 控件的“click”事件绑定流图中的“button_click”命令，操作步骤如下：

步骤一：选中 button 控件，在控件编辑器中添加“click”事件；

步骤二：为“click”事件绑定流图中的“button_click”命令，如下图所示

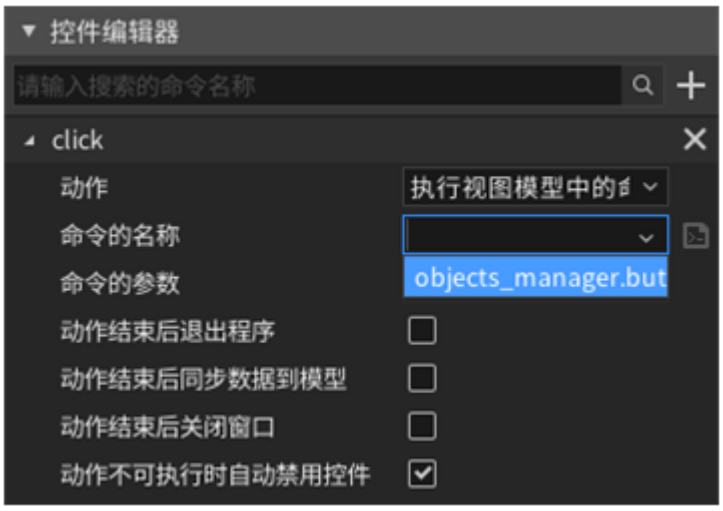

图 2.23 事件设置

2.3.7 为 label 控件绑定数据

为 label 控件绑定流图上保存的数据，可以帮助我们观察数据与动画的关联性，操作如下图所示

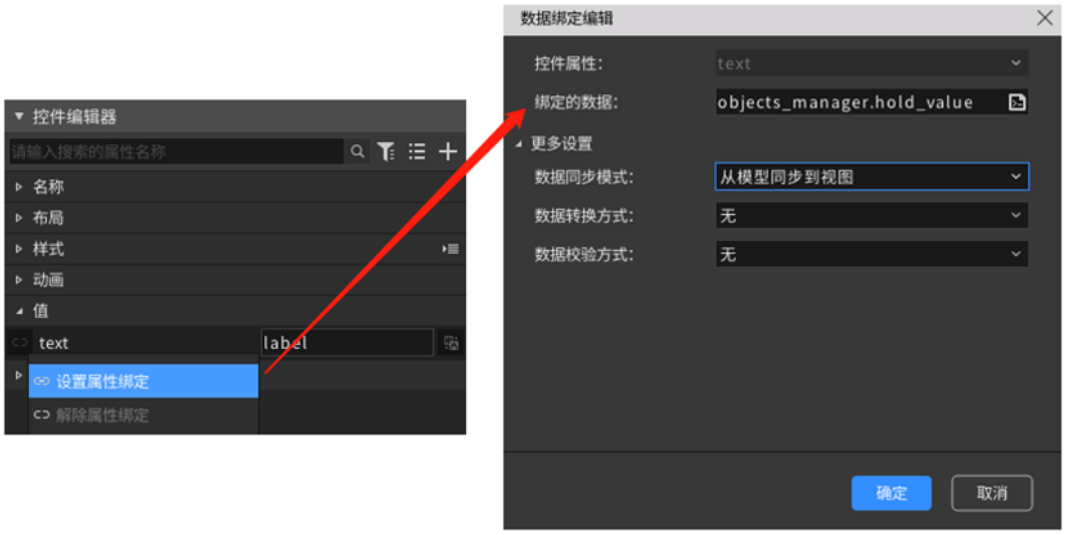

图 2.24 label 绑定数据

2.3.8 运行效果

在 AWTK Designer 中点击”模拟运行”按钮启动程序，通过点击 button 可以实现控制 image 控件动画的启停，label 显示的数据也随着动画的启停在 0 与 1 之间变化。

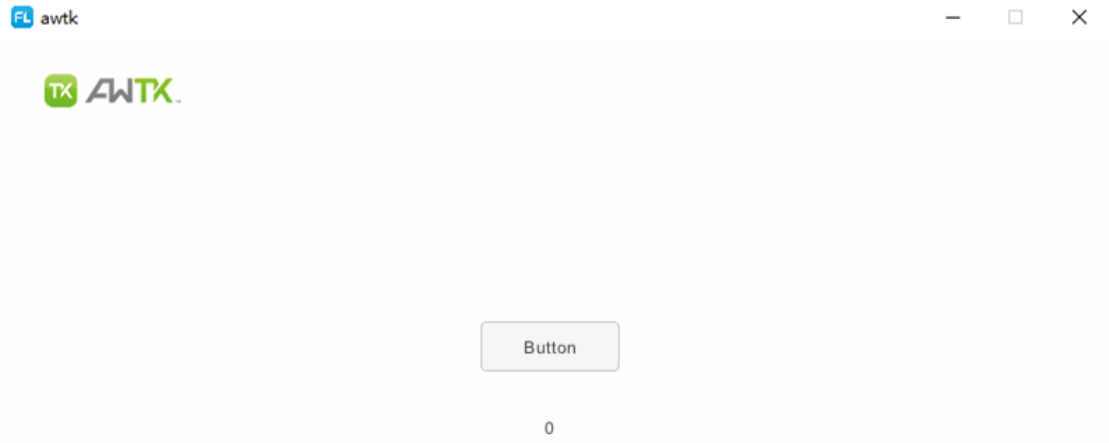

图 2.25 停止动画

图 2.26 开启动画

注: 本案例中，对数据的修改是在流图中进行的，如果在界面中使用 fscript 代码修改数据，在数据修改完成后需要调用 notify_view_props_changed() 触发数据修改事件，否则数据不会同步

3. 高级功能案例

本章将介绍 AWStudio 开发的 AWFlow+AWTK 应用程序的高级案例，以此演示一些特有功能点，目前包括以下案例：

- 基于 Modbus 的温湿度采集设备。
- 基于 ZWS 云的数据传输。

3.1 基于 Modbus 的温湿度采集设备

3.1.1 案例介绍

本案例基于温湿度变送器（传感器）实现了温湿度采集应用。该传感器采用 RS485 接口与设备连接，并使用 ModbusRTU 协议进行数据通信。此处我们主要讲解在 AWStudio 中如何基于 ModbusRTU 协议通信，定时采集温湿度数据，帮助读者了解 AWFlow Designer 中 Modbus 相关节点的用法。

3.1.2 硬件设备

本案例中用到的硬件设备主要有 **显控一体机**和 **温湿度变送器**，它们的介绍详见下文：

1. 显控一体机

显控一体机主要用于运行 AWFlow+AWTK 低代码应用程序。在本案例中，它通过 RS485 接口与温度传感器连接，使用 ModbusRTU 协议进行通信，定时采集温湿度数据，设备如下图所示：

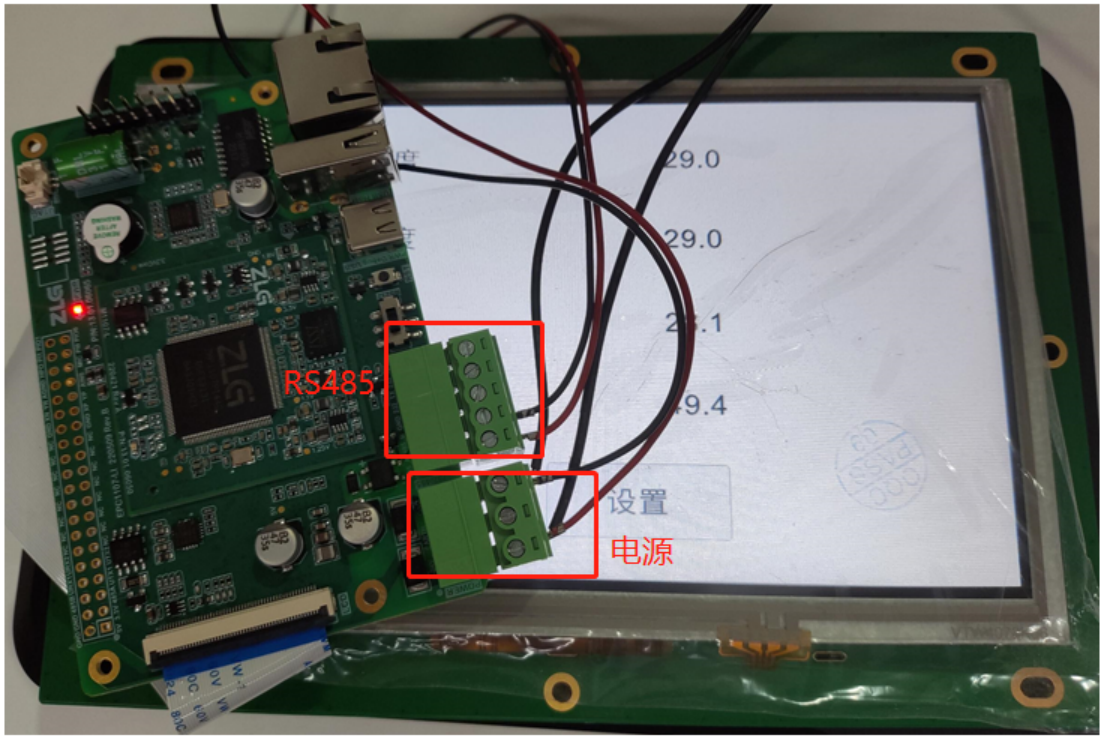

图 3.1 显控一体机

2. 温湿度变送器

温湿度变送器（传感器）主要用于测量外部温度并通过 RS485 接口与显控一体机连接，使用 ModbusRTU 协议进行通信，显控一体机会定时从传感器上采集温湿度数据，传感器如下图所示：

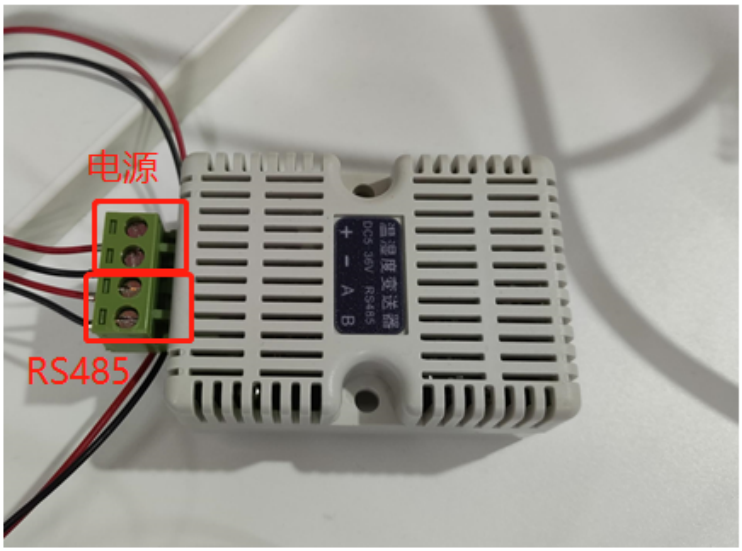

图 3.2　温湿度变送器

3.1.3　基于 Modbus 实现数据采集

为了方便演示，这里定时每一秒采集一次温湿度变送器的数据并将其显示到界面上，首先在工作区中双击打开 AWFlow 项目，进入 AWFlow Designer 后，新建一张流图。

步骤一：将”节点列表”中”th_transmitter_in”类型的节点拖拽到流图编辑区，设置如下：

- 设置”从机 ID”属性为 1，此处为温湿度变送器的 ID，可根据实际情况填写。
- 设置”温度寄存器地址”属性为 0 。
- 设置”湿度寄存器地址”属性为 1 。此处两个寄存器地址由温湿度变送器决定。
- 设置”输出周期”属性为 1000，每 1 秒输出一次读取数据。
- 点击”modbus 配置参数”属性右边的编辑按钮，进入”modbus_master_rtu”节点的编辑。

注: 此处使用的节点为该温湿度变送器专用，如果为其他设备，请使用”modbus_master_in”节点与”modbus_data_in_convert”节点搭配实现 modbus 通信。

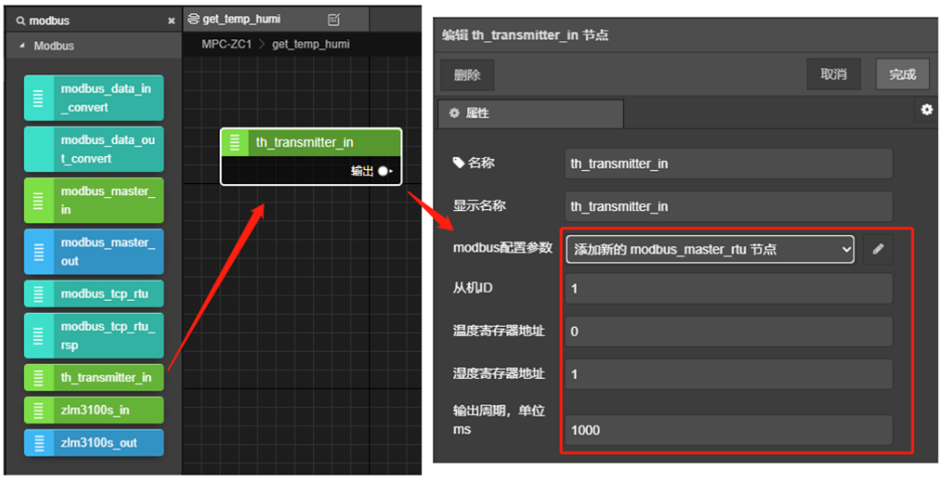

图 3.3　配置 th_transmitter_in

- 设置通信串口”端口”属性为”/dev/ttyZMP1”，这里填写的是设备与温湿度变送器连接后的串口驱动名称，读者请根据实际情况填写。
- 设置”波特率”为 9600，由温湿度变送器的通信协议规则决定。
- 设置”数据位”属性为”8bit”，由温湿度变送器的通信协议规则决定。
- 设置”校验位”属性为”none”，由温湿度变送器的通信协议规则决定。
- 设置”停止位”属性为”1bit”，由温湿度变送器的通信协议规则决定。

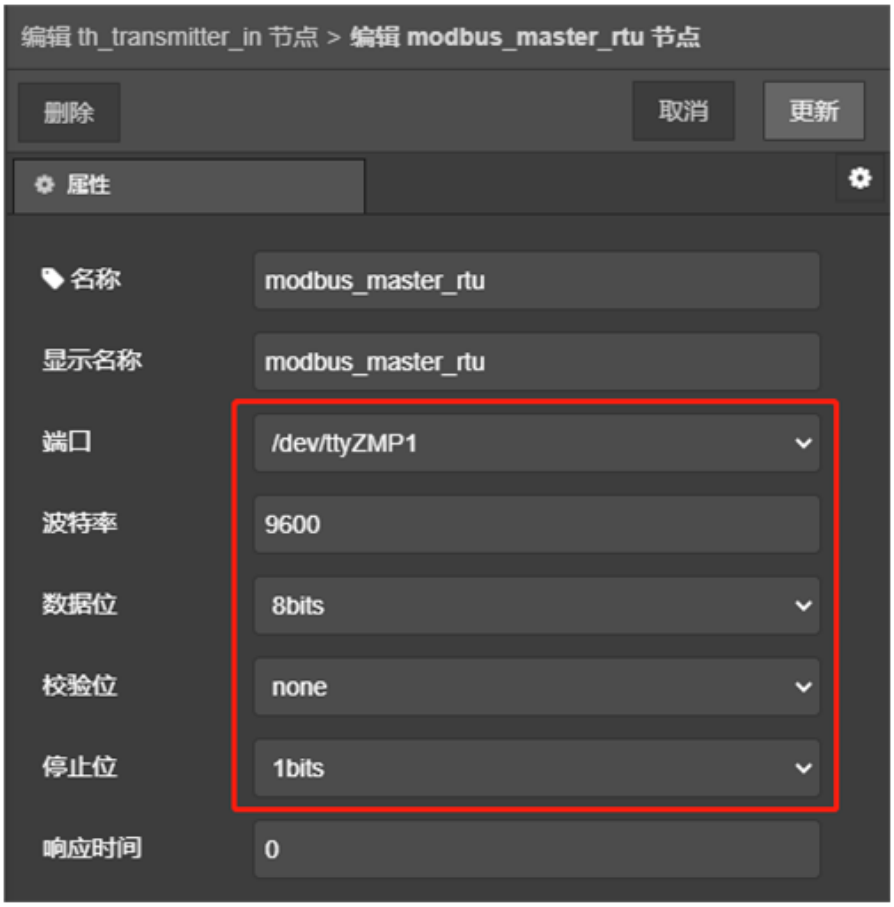

图 3.4　配置 modbus 参数

步骤二：th_transmitter_in 节点会直接将处理好的温湿度数据传递到下个节点，接下来只需参考上一章的内容，使用**锁存器**（即”hold”节点）将采集到的温湿度值显示到界面上即可。

注: th_transmitter_in 节点传递出来的温湿度数据分别为”msg.temperature”和”msg.humidity”

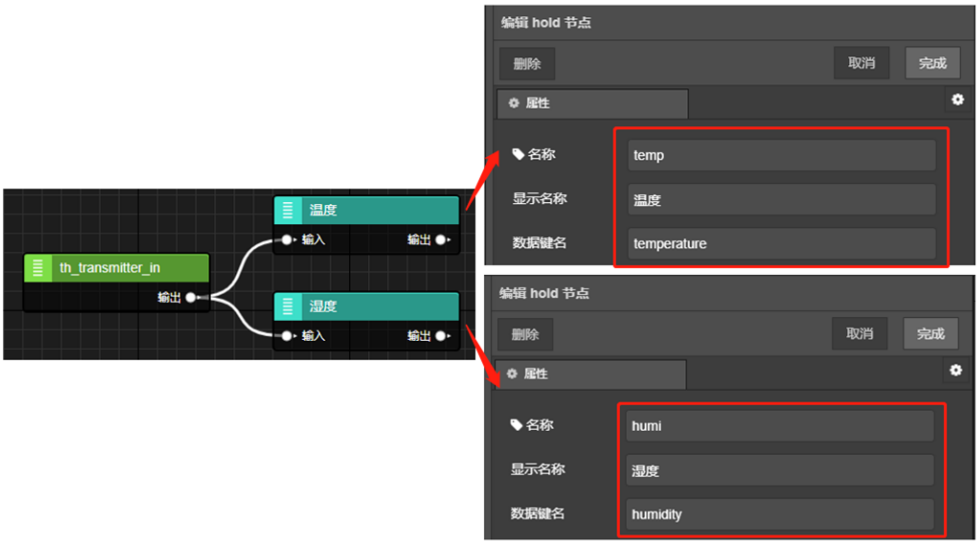

图 3.5　使用锁存器保存温湿度

3.1.4　运行结果

FlowModbus-Demo 温湿度采集设备运行的效果如下图所示：

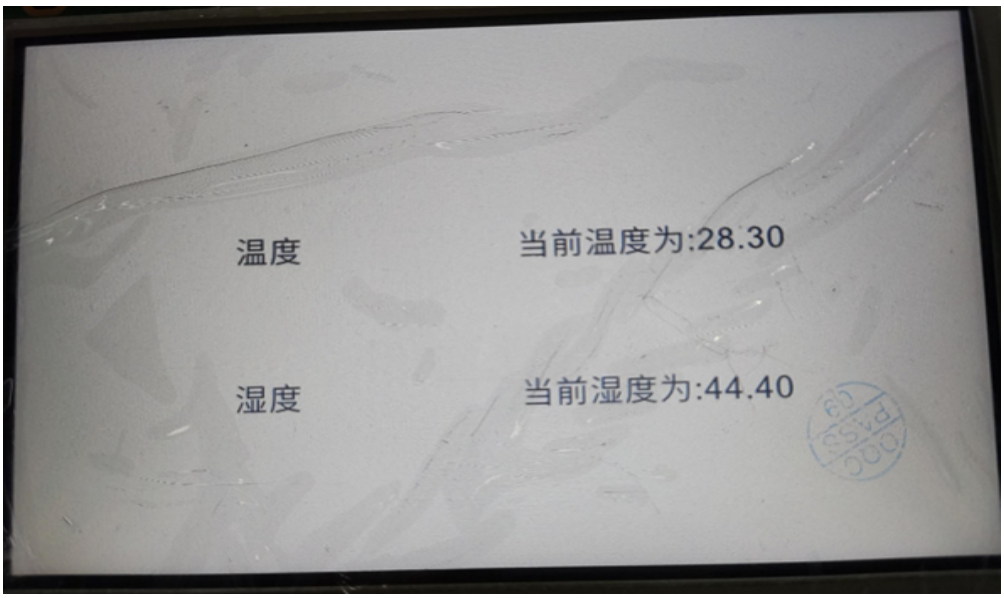

图 3.6　运行效果

3.2 基于 ZWS 云的数据传输

3.2.1 案例介绍

本案例是通过 AWFlow Designer 软件模拟与 ZWS 云进行数据的收发，此处我们主要讲解如何使用 ZWS 云管理设备，并通过 AWFlow Designer 实现与 ZWS 云进行数据交流，帮助读者了解 AWFlow Designer 中 ZWS 云相关节点的用法。

3.2.2 流图数据上传到云端（ZWS 云）

将设备采集到的数据上传到云端是物联网应用场景中常见的需求，下面我们以 ZWS 云[7] 为例，介绍如何把 AWFlow 流图上的温度数据上传到云端。

1. 在 ZWS 云上添加设备

为方便后续演示将上文 2.1 章节中随机生成的温度值上传至 ZWS 云，我们需要登录 ZWS 云并在上面添加一台云设备，步骤如下：

步骤一：登录 ZWS 云[8] 后，在左侧的菜单栏中选择【设备管理】->【设备类型】，在”设备类型”页面点击”添加类型”。

步骤二：在弹出的”添加设备类型”对话框中，将”类型模板”设置为”invert”；将”类型名称”设置为”temp_iot”，点击”确认”按钮。

步骤三：添加成功后，可以在设备类型列表中看见”temp_iot”类型，如下图所示：

图 3.7 添加设备类型

步骤四：点击”temp_iot”类型”操作”列中的编辑按钮，进入设备类型的详情页面，选择”数据点配置”页面，可以看到设备类型本身自带的默认数据点，我们后续将使用自带的”逆变器温度 (temperature)”演示数据上传。

如果需要上传其他数据，可以自行添加数据点，比如”室内温度 (indoor_temperature)”和”设定温度 (setting_temperature)”等，如下图所示：

[7] https://www.zlgcloud.com
[8] https://www.zlgcloud.com

图 3.8　配置设备类型的数据点

步骤五：在页面的左侧的主菜单中，切换到【设备管理】->【设备列表】，点击”设备列表”页面中的”添加设备”按钮，然后在”添加设备”页面中，设置以下信息：

- 将”设备类型”设置为”temp_iot”；
- 将”设备名称”设置为”温度采集终端”；
- 将”设备 ID”设置为”temp_iot_0”；
- 选择系统生成秘钥。

步骤六：点击下方的”保存”按钮，设备添加完毕。此时返回”设备列表”页面即可看见刚才添加的设备并显示离线状态，其中”设备秘钥”会在后续上传数据时用到，如下图所示：

图 3.9　成功添加设备

完成以上 ZWS 云上的准备工作后，我们便可以开始编辑流图了，步骤详见下文。

2. 将数据上传到 ZWS 云

将数据上传到 ZWS 云需要用到 AWFlow 中的 zws_iot 节点和 zws_iot_data_out 节点，它们分别用于配置 ZWS 云以及上传指定数据。

此处依旧以实现一个空调控制器为例，假设需要把上文 2.1 章节中随机生成的温度值上传到 ZWS 云上，该温度值保存在 temp_iot_0 设备的 temperature 数据点中，实现步骤如下：

步骤一：参考上文 2.1.1 章节每秒钟随机生成一个温度值，其中 random_number 节点生成的随机数的主题属性为 temperature。

步骤二：将”节点列表”中的”fscript”类型的节点拖到流图编辑区，该节点需要构建下一节点”zws_iot_data_out”所需的数据包，这里以”Key-Value”的形式上传数据，所需数据属性如下：

(1) msg.payload：需要上传的数据值，此处指随机生成的温度值，由前面的 random_number 节点提供。

(2) msg.key_name：需要上传的数据值名称，需要与 ZWS 云上的设备数据点名称一致，此处指上文在 temp_iot 设备中添加的数据点 temperature。

(3) msg.zwsPayloadType：该属性为 int 类型变量，表示上传数据的类型，此次设置为 2，表示 double 类型，更多类型详见”zws_iot_data_out”节点的帮助说明。

“fscript”节点生成上传所需数据包的代码如下：

```
/* msg.topic 为上一个节点传输过来的温度名称，即需要上传设备数据点名称 */
/* 例如：temperatuer、indoor_temperature、setting_temperature */
/* 此处将其赋值给 msg.key_name，下一节点会根据该名称将数据上传至云端指定设备数据点 */
msg.key_name = msg.topic;
/* 设置 msg.zwsPayloadType 为 2，表示上传的数据类型为 double */
msg.zwsPayloadType = 2;
```

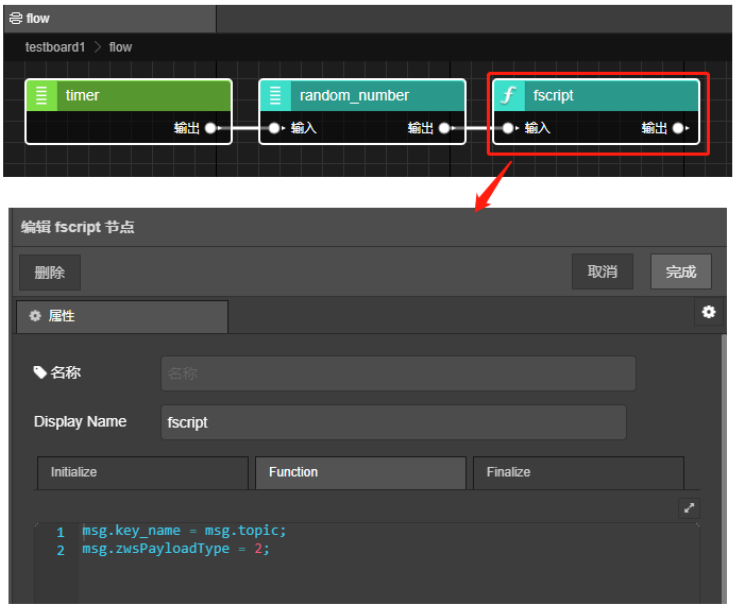

图 3.10　生成上传所需的数据包

步骤三：将”节点列表”中的”zws_iot_data_out”类型的节点拖到流图编辑区，设置如下：

- 设置”设备配置”属性，如果没有添加过节点，则需要手动添加。
- 设置”数据上报方式”属性为”Key-Value”，通过该形式上传数据，提供 payload、key_name 和 zwsPayloadType 属性，详见上一步。

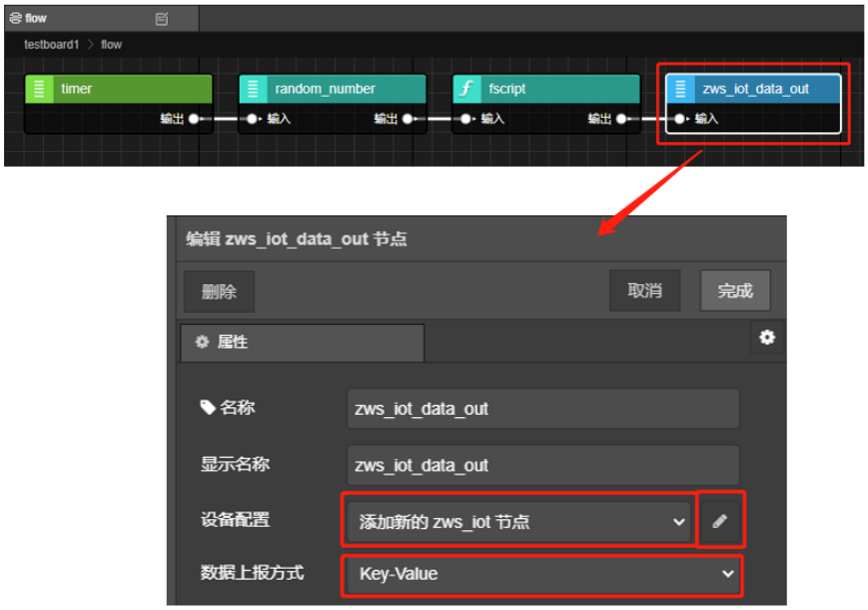

图 3.11　将数据上传至 ZWS 云

在本项目中，需要添加新的“zws_iot”节点，在“设备配置”一栏选择“添加新的 zws_iot 节点，然后点击右边的按钮，进入配置页面

- 设置”是否连接 zws 平台”属性为”connected”，即链接 ZWS 云。
- 设置”设备类型”属性为”temp_iot”，即 ZWS 云中添加的设备类型，它定义了设备的各种属性。
- 设置”设备 id”属性为”temp_iot_0”，即 ZWS 云中添加的设备，ZWS 通过 **设备类型 + 设备 ID** 识别一个唯一的设备。
- 设置”设备密钥”属性为 ZWS 云中添加设备时自动生成的秘钥，在链接 ZWS 时会对该秘钥进行校验，确保设备链接是合法的，此处的秘钥为”4d4382d0-7d92-11ec-959e-c1c503b62d13”。

以上信息均需要读者根据实际情况填写，无需跟本文一致，配置完毕后，点击添加，并将”zws_iot_data_out”节点的“设备配置”属性修改为新添加的”zws_iot”节点

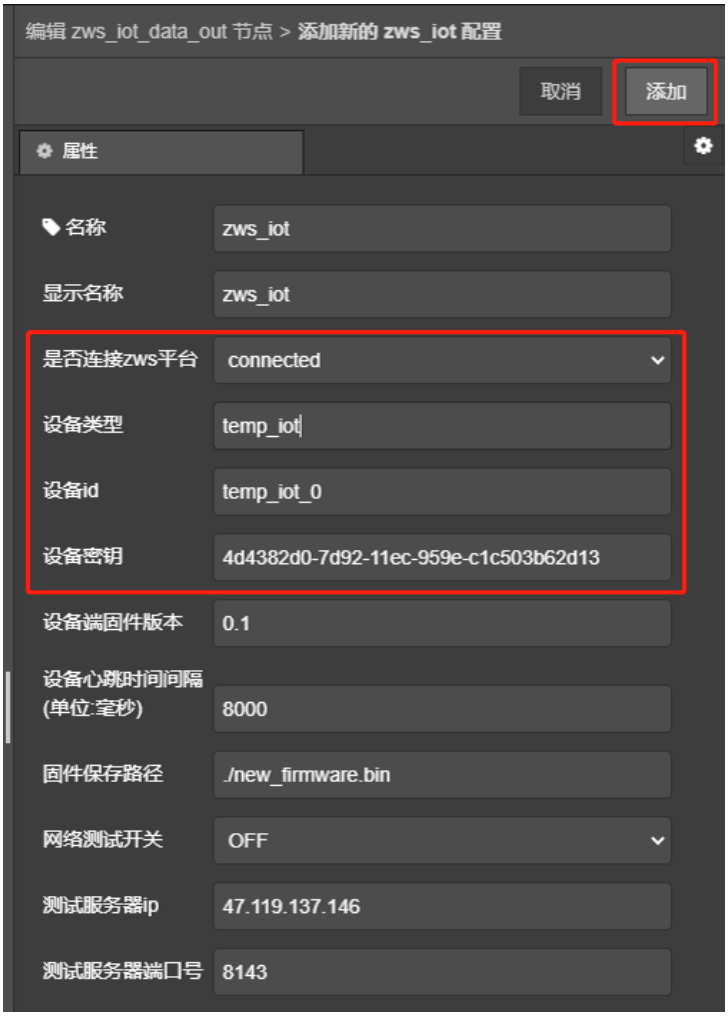

图 3.12　配置 ZWS 云

3. 运行效果

此处上传数据与 GUI 界面无关，可以直接使用 AWFlow Designer 中内置的模拟器来运行流图，如下图所示：

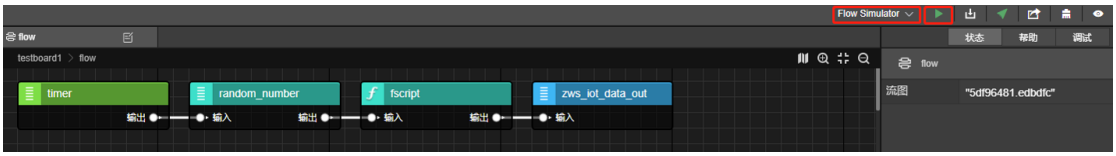

图 3.13　模拟运行流图

运行成功后，可以在 ZWS 云的设备列表中进入温度采集终端（设备 ID：temp_iot_0）详情页面，如下图所示：

图 3.14　查看设备详情

然后切换到实时数据页面，并选择”data 数据”，可以看到实时上传的 temperatuer 数据，如下图所示：

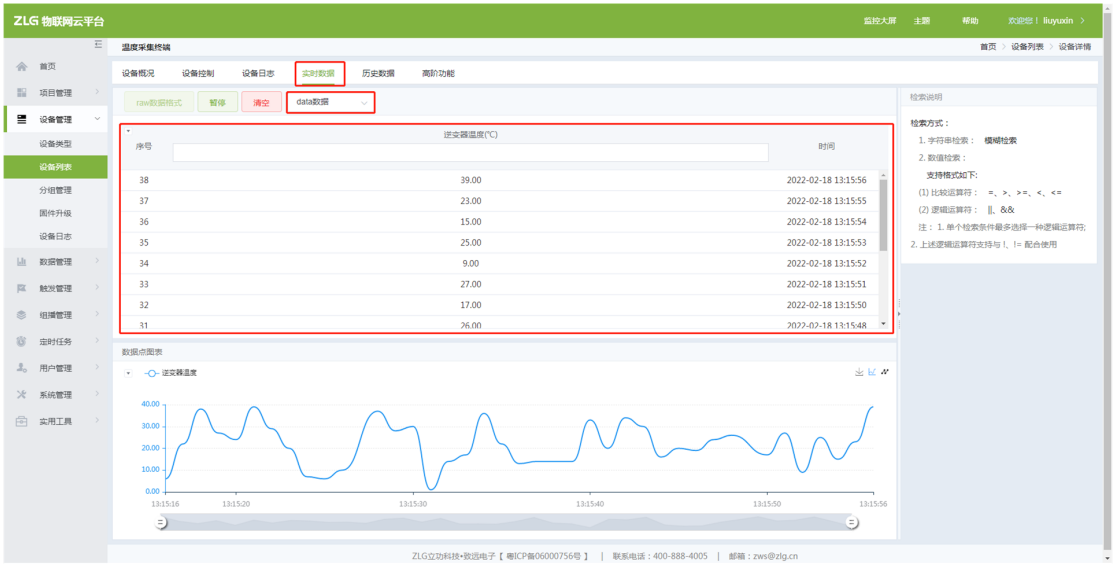

图 3.15　查看实时数据

3.2.3　云端（ZWS 云）发送命令到流图

ZWS 云[9] 除了可以保存设备上传的数据外，还可以向设备发送命令，常用于远程控制。比如一个使用 AWFlow 技术实现的空调机，我们可以在云端远程开关空调。下面以ZWS 云[10]为例，介绍如何从云端控制 AWFlow 流图运行。

1. 在 ZWS 云中发送命令

首先请参考上文 3.2.2 章节注册并登录 ZWS 云[11]，添加对应设备，此处仍然以 3.2.2 章节中添加的 temp_iot_0 设备为例，假设需要从 ZWS 云上发送温度值给设备，步骤如下：

步骤一：登录 ZWS 云[12] 后，参考上一章节的内容，在 ZWS 云的设备列表中进入温度采集终端（设备 ID：temp_iot_0）的详情页面。

[9] https://www.zlgcloud.com
[10] https://www.zlgcloud.com
[11] https://www.zlgcloud.com
[12] https://www.zlgcloud.com

步骤二：在设备详情页面中切换到”设备控制”，设置如下：

- 设置”设备命令”为”透传命令（raw）”。
- 设置”编码格式”为”ascii”。

完成以上设置后，可以在”数据”编辑框中输入想要发送的数据，例如此处发送温度值”33”，然后点击确定按钮即可发送，如下图所示：

图 3.16　在 ZWS 云中发送命令

2. 接收 ZWS 云发送的命令

在流图中接收 ZWS 云发送的命令需要用到 AWFlow 中的 zws_iot 节点和 zws_iot_data_in 节点它们分别用于配置 ZWS 云以及接收数据，实现步骤如下：

步骤一：打开 AWFlow Designer，将”节点列表”中的”zws_iot_data_in”类型的节点拖到流图编辑区，并该节点的”设备配置”属性设置为上一节所添加的“zws_iot”，如下图所示：

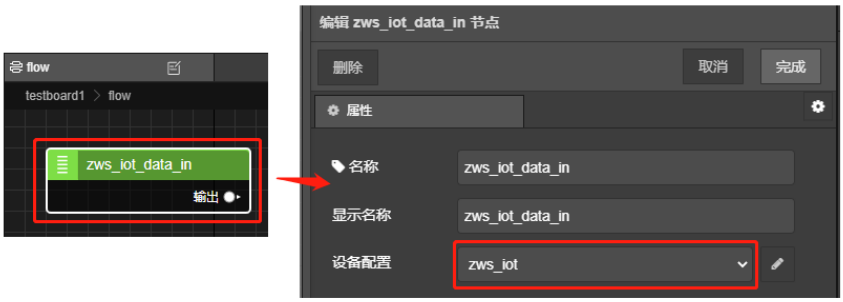

图 3.17　zws_iot_data_in 节点

步骤二：将”节点列表”中的”log”类型的节点拖到流图编辑区，并与上一步骤中创建的”zws_iot_data_in”节点相连，用于在模拟运行流图时打印”zws_iot_data_in”节点收到的数据，如下图所示：

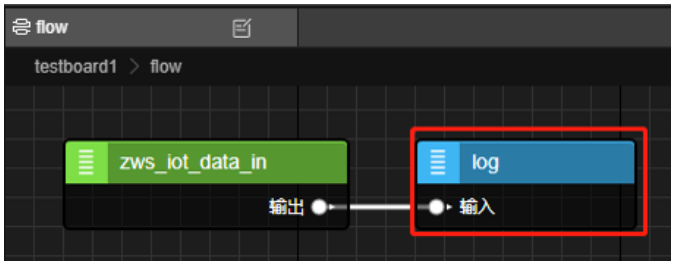

图 3.18　log 节点

3. 运行效果

此处上传数据与 GUI 界面无关，可以直接使用 AWFlow Designer 中内置的模拟器来运行流图，方法详见 3.2.2 章节。

运行成功后，按照上文中的步骤，在 ZWS 云中给指定设备（此处为 temp_iot_0）发送温度值 33。此时，可以在 AWFlow Designer 的调试页面看到”zws_iot_data_in”节点收到的数据，如下图所示：

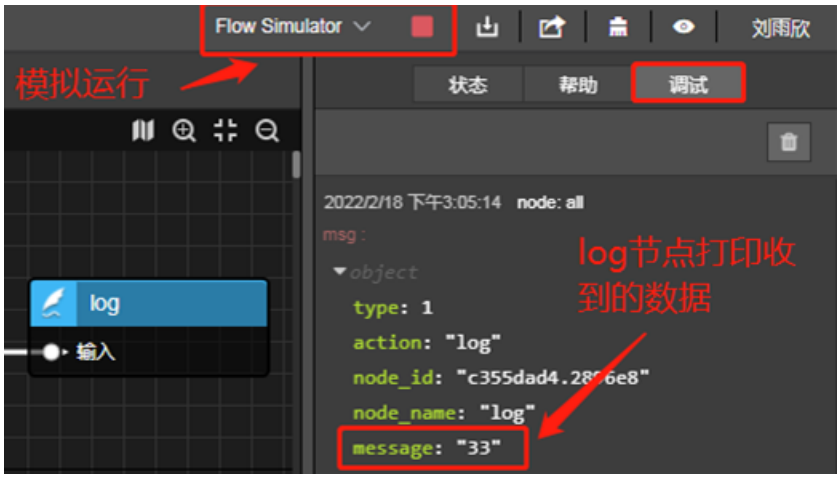

图 3.19　接收 ZWS 云发送的命令

需要注意的是，由于 ZWS 云目前仅支持透传命令，因此”zws_iot_data_in”节点只能收到字符串类型的数据，在实际应用中，用户可借助”fscript”节点自行解析。

4. 综合案例 1：温度采集终端

本章开始将介绍一些使用 AWStudio 开发的 AWFlow+AWTK 低代码应用的综合案例，帮助读者加深对 AWFlow Designer + AWTK Designer MVVM 开发模式的理解。

4.1 案例介绍

本章介绍的综合案例为温度采集终端（FlowZwsIot-Demo），它业务逻辑流图和界面如图所示，主要功能如下：

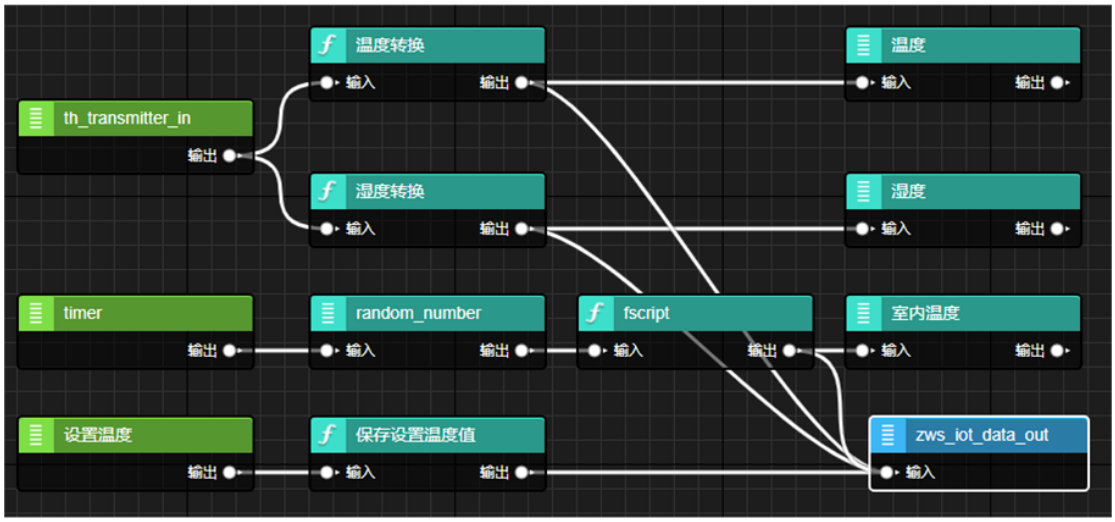

图 4.1　温度采集终端流图

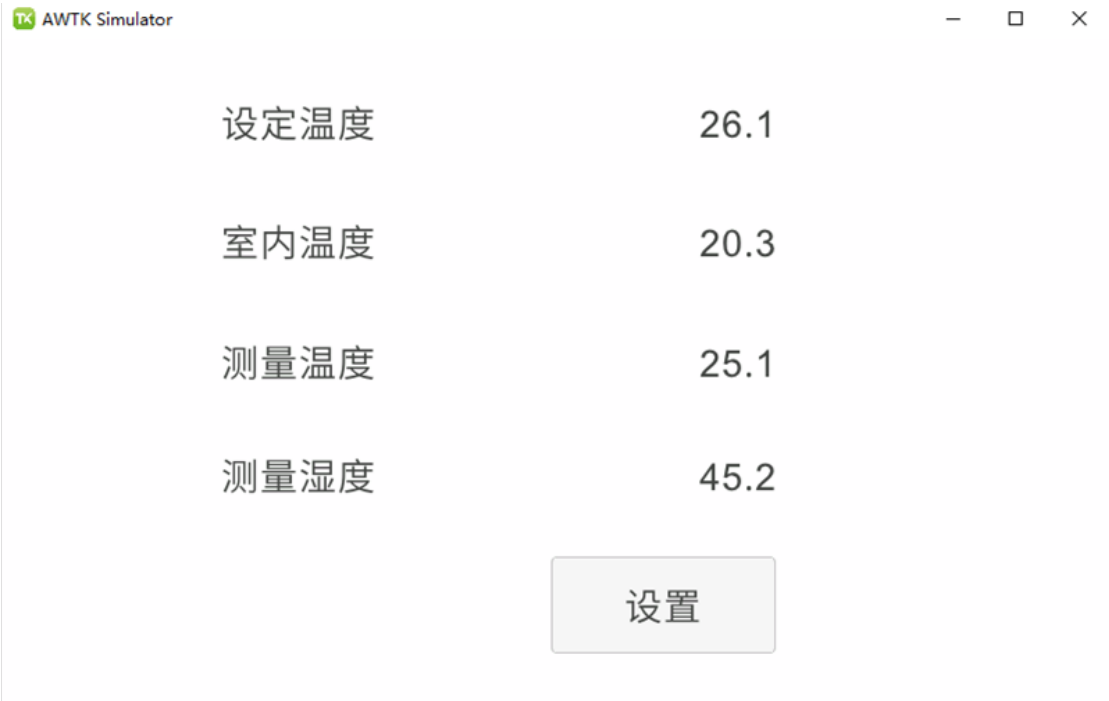

图 4.2　主界面

图 4.3　设置界面

- 使用 RS485 接口 ModbusRTU 协议通信，定时采集温湿度变送器的温度值、湿度值（测量温度、测量湿度），将其显示到主界面并上传到云端；
- 定时产生一个随机温度值（室内温度），将其显示到主界面并上传到云端；
- 点击”设置”按钮进入”设置界面”，在编辑框中输入温度值（设定温度），点击”确定”按钮，将其显示到主界面并上传到云端。

4.2　准备工作

4.2.1　硬件设备

本案例中用到的硬件设备主要有 **显控一体机**和 **温湿度变送器**，它们与上一章 Modbus 案例中使用的为同样的设备，详情可以查看 3.1 章节。

4.2.2　ZWS 云

在案例介绍中，我们提到需要将温度值数据上传到云端，这里的云端指的是 ZLG 物联网云平台——ZWS 云[13]，因此我们需要注册一个 ZWS 云平台的账号，并登录该平台添加云端设备，具体操作详见上文 3.2 章节。

本案例中我们需要上传以下数据，请参考 3.2 章节给云端设备添加对应的数据点：

- 温湿度变送器的测量温度，数据点字段名称为”temp”。
- 温湿度变送器的测量湿度，数据点字段名称为”humi”。
- 随机生成的室内温度，数据点字段名称为”indoor_temp”。
- 界面上的设定温度，数据点字段名称为”setting_temp”。

[13] https://www.zlgcloud.com

注: 如果不需要将数据上传到云平台，可以跳过本章节。

4.3 温度采集终端流图实现

用 AWStudio 新建一个 AWFlow 项目，双击打开该项目，进入 AWFlow Designer 后，新建一张流图并编辑。

4.3.1 读取温湿度变送器数据

显控一体机与温湿度变送器之间使用 RS485 接口 ModbusRTU 协议通信，因此我们需要根据温湿度变送器的使用手册进行 Modbus 的配置，定时采集温湿度变送器的数据并缓存起来显示到程序界面上。

1. 配置 th_transmitter_in

如下图所示，将”节点列表”中的”th_transmitter_in”类型的节点拖到流图编辑区，设置如下：

- 设置”从机 ID”属性为 1 ，此处为温湿度变送器的 ID，可根据实际情况填写。
- 设置”温度寄存器地址”属性为 0 。
- 设置”湿度寄存器地址”属性为 1 。此处两个寄存器地址由温湿度变送器决定。
- 设置”输出周期”属性为 1000 ，每 1 秒输出一次读取数据。
- 点击”modbus 配置参数”属性右边的编辑按钮，进入”modbus_master_rtu”节点的编辑。

注: 此处使用的节点为该温湿度变送器专用，如果需要使用其他设备，请使用”modbus_master_in”节点与”modbus_data_in_convert”节点搭配实现 modbus 通信。

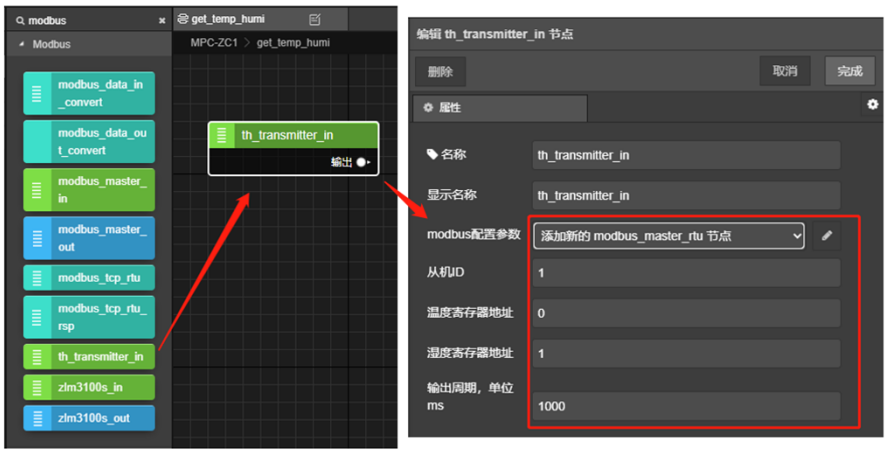

图 4.4 配置 th_transmitter_in

2. 配置 modbus 参数

点击编辑按钮进入 modbus_master_rtu 节点的编辑：

- 设置通信串口”端口”属性为”/dev/ttyZMP1”，这里填写的是设备与温湿度变送器连接后的串口驱动名称，读者请根据实际情况填写。
- 设置”波特率”为 9600，由温湿度变送器的通信协议规则决定。
- 设置”数据位”属性为”8bit”，由温湿度变送器的通信协议规则决定。

- 设置”校验位”属性为”none”，由温湿度变送器的通信协议规则决定。
- 设置”停止位”属性为”1bit”，由温湿度变送器的通信协议规则决定。

图 4.5　配置 modbus 参数

3. 传递温湿度数据

使用两个 fscript 节点将温湿度赋值给“msg.temp”、“msg.humi”，作用是传递数据到云平台时可以使用该字段名直接传递，如下图所示：

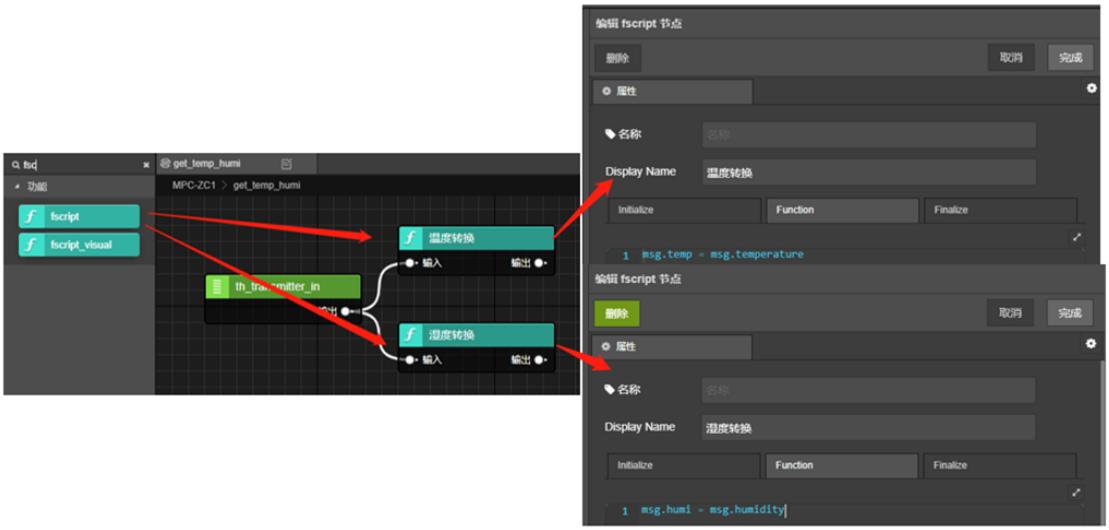

图 4.6　传递温度值数据

4. 保存测量温湿度（用于 AWTK 访问）

上个步骤拿到了测量温度与测量湿度，接下来要将它们显示到界面上，因此我们要用锁存器保存它们，用于 AWTK 访问，将”节点列表”中”hold”类型的节点拖到流图编辑区，设置如下：

- 设置”名称”属性为”temp”与”humi”。
- 设置”显示名称”属性为”温度”与”湿度”。
- 设置需要保存的数据”Key”为上一个节点（fscript 节点）传输过来的”temp”与”humi”。

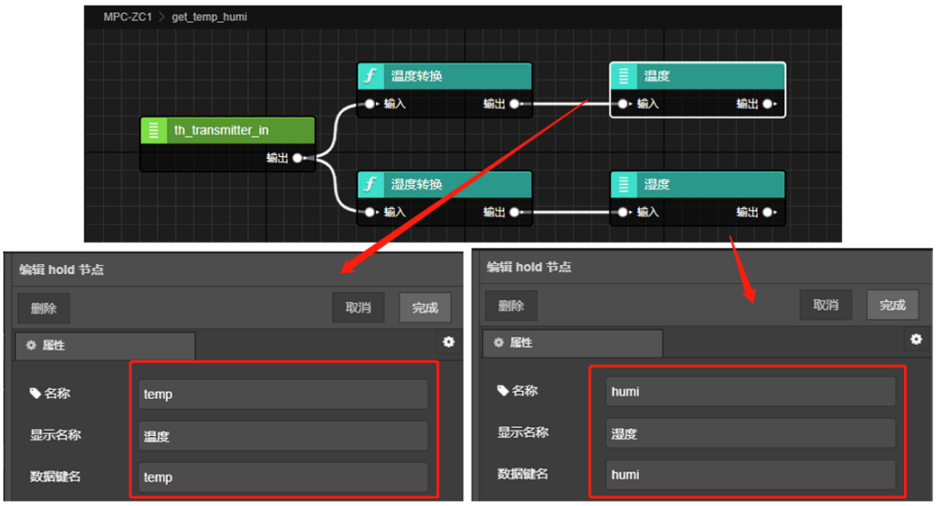

图 4.7　使用锁存器保存测量温度与湿度

4.3.2　生成随机温度值

本案例中的室内温度是随机生成的，我们只需借助”timer”类型的节点定时触发生成事件，并使用”random_number”类型的节点生成随机数即可，具体步骤详见下文。

1. 定时生成随机温度值

将”节点列表”中的”timer”与”random_number”类型的节点拖到流图编辑区，将两节点相连，设置定时器的定时周期为 1 秒，设置随机温度的取值范围为 0 - 36，如下图所示：

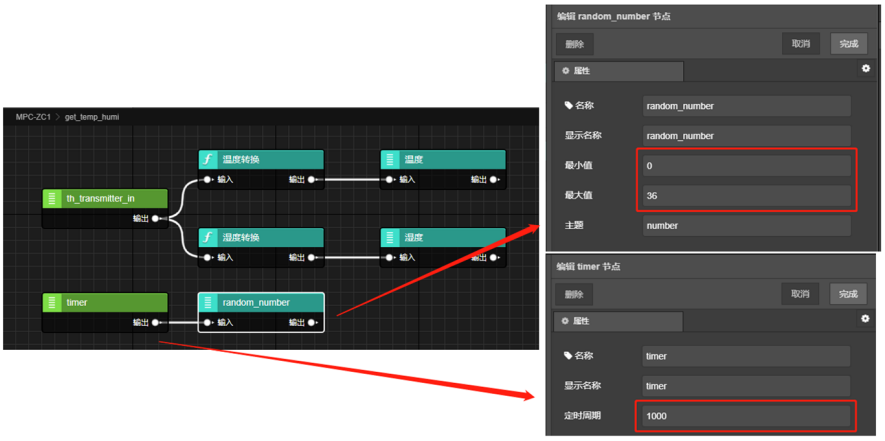

图 4.8　定时生成随机温度值

2. 传递温度数据

与读取数值一样，此处需要使用”fscript”节点将数据赋值给”msg.indoor_temp”以便后续传递上云使用，如下图所示：

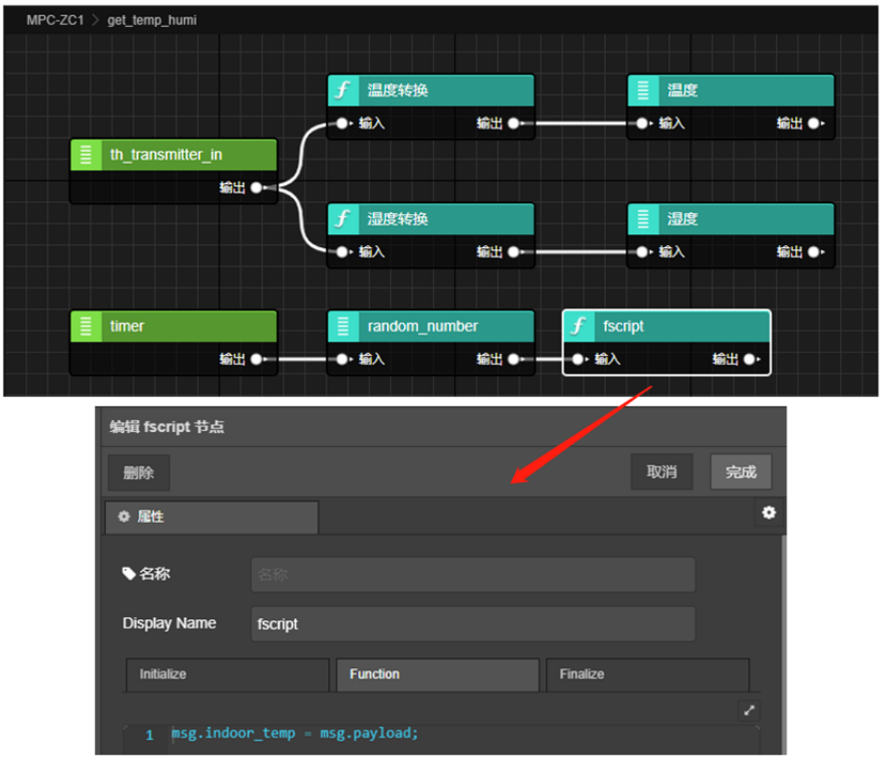

图 4.9　传递室内温度数据

3. 保存室内温度（用于 AWTK 访问）

与上文保存测量温度一样，此处也需要用锁存器将随机生成的室内温度保存起来，用 AWTK 访问，方便将其显示到界面上。如下图所示，将”节点列表”中”hold”类型的节点拖到流图编辑区，设置如下：

- 设置”名称”属性为”indoor_temperature”。
- 设置”显示名称”属性为”indoor_temperature”。
- 设置需要保存的数据”Key”为上一个节点（random_number 节点）传输过来的”payload”。

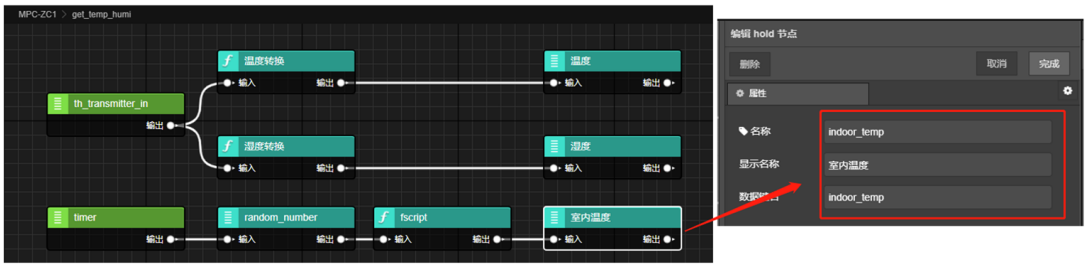

图 4.10　使用锁存器保存室内温度

4.3.3 响应界面的设定温度按键

在本案例中，只有当我们点击”设置界面”中的”确定”按钮，才会把编辑框中输入的温度值（设定温度）显示到主界面，因此我们需要在流图中添加一个界面事件的接收器（uiinput 节点），用于界面的对应按键进行绑定。

1. 添加界面事件接收器

将”节点列表”中的”uiinput”类型的节点拖到流图编辑区，然后将节点的”名称”属性设置为”set_temp”和”显示名称”属性设置为”设置温度”，如下图所示：

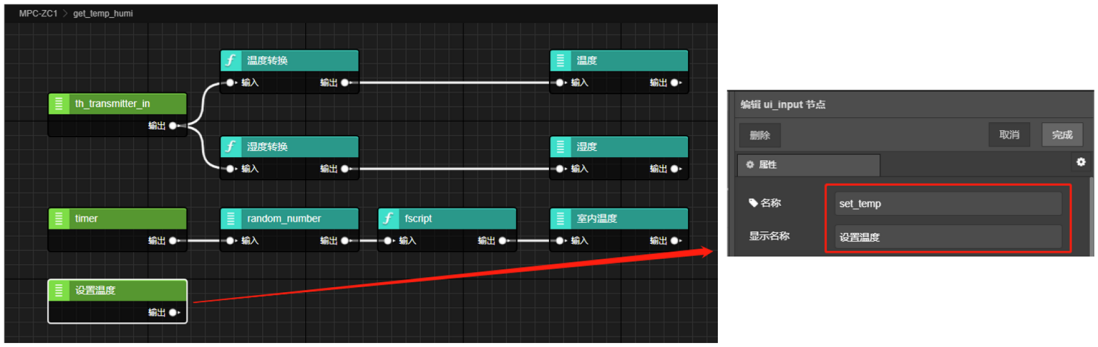

图 4.11 添加 UI 事件接收器

2. 保存设定温度

如下图所示，将”节点列表”中”fscript”类型的节点拖到流图编辑区，设置如下：

- 设置节点”名称”属性和显示名称”Display Name”为”save_setting_temperature”。
- 切换到”Function”页面，添加保存脚本，代码如下：

```
/* msg.payload 为上一个节点传过来的字符串，解析后赋值给msg.setting_
↪temp用于传递上云*/
var obj = command_args_to_object(msg.payload)
msg.setting = obj.temp;
/* 将设定的温度值保存到全局 global 对象中 */
global.setting_temp = obj.temp;
/* 通知界面更新 */
notify_view_props_changed();
```

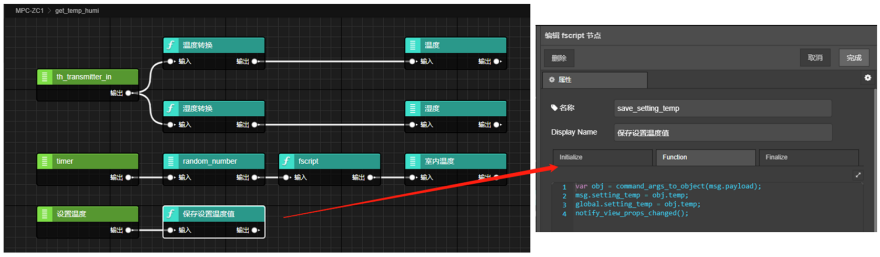

图 4.12 处理设定温度

如果需要设置初始值，那么可以切换到”Initialize”页面，添加初始化代码，例如此处将设定温度 setting_temperature 初始化为 26.5，代码如下：

```
app_conf.setting_temperature = 26.5;
```

4.3.4 把温度数据保存到云端

我们在以上章节中一共拿到四个数据：温湿度变送器的测量温湿度、随机生成的室内温度和界面上的设定温度，接下来需要将这些数据上传至 ZWS 云[14] 保存，分别对应四个数据点 temp（测量温度）、humi（测量湿度）indoor_temp（室内温度）和 setting_temp（设定温度）。

注: 请首先在 ZWS 云中新建设备，并添加数据点，操作步骤详见本文 2.3 章节。

实现该功能需要用到 AWFlow 中的 zws_iot_data_out 节点，它用于上传数据，步骤详见下文。

1. 将数据上传至 ZWS 云

如下图所示，将”节点列表”中的”zws_iot_data_out”类型的节点拖到流图编辑区，设置如下：

- “设备配置”属性如果没有添加过”zws_iot”节点，则选择”添加新的 zws_iot 节点”，点击右边的按钮进行编辑。
- 设置”Output Type”属性为”Key-Values”，通过该形式上传数据。
- 设置”目标键值对”属性为”temp,humi,setting_temp,indoor_temp”，这四个数据与 zws 云所添加的四个字段名一一对应，上传的数据就为前面为 msg 对象所添加的四个属性。

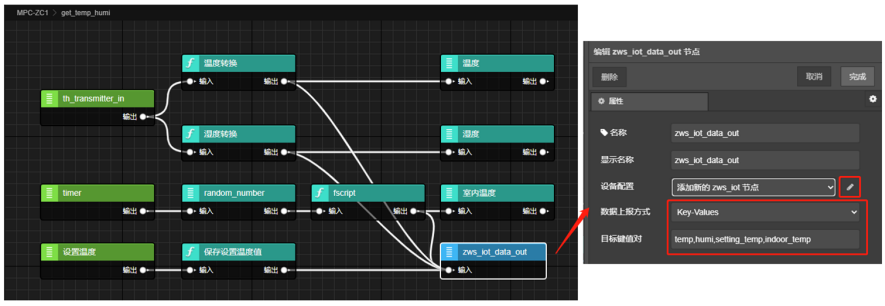

图 4.13 将数据上传至 ZWS 云

“zws_iot”节点设置如下

- 设置”是否连接 zws 平台”属性为”connected”，即链接 ZWS 云。
- 设置”设备类型”属性为”temp_iot1”，即 ZWS 云中添加的设备类型。
- 设置”设备 id”属性为”temp_iot_0”，即 ZWS 云中添加的设备 ID。
- 设置”设备密钥”属性为 ZWS 云中添加设备时自动生成的秘钥，可在 ZWS 云的设备列表中复制。

以上信息均需要读者根据实际情况填写，无需跟本文一致。

[14] https://www.zlgcloud.com

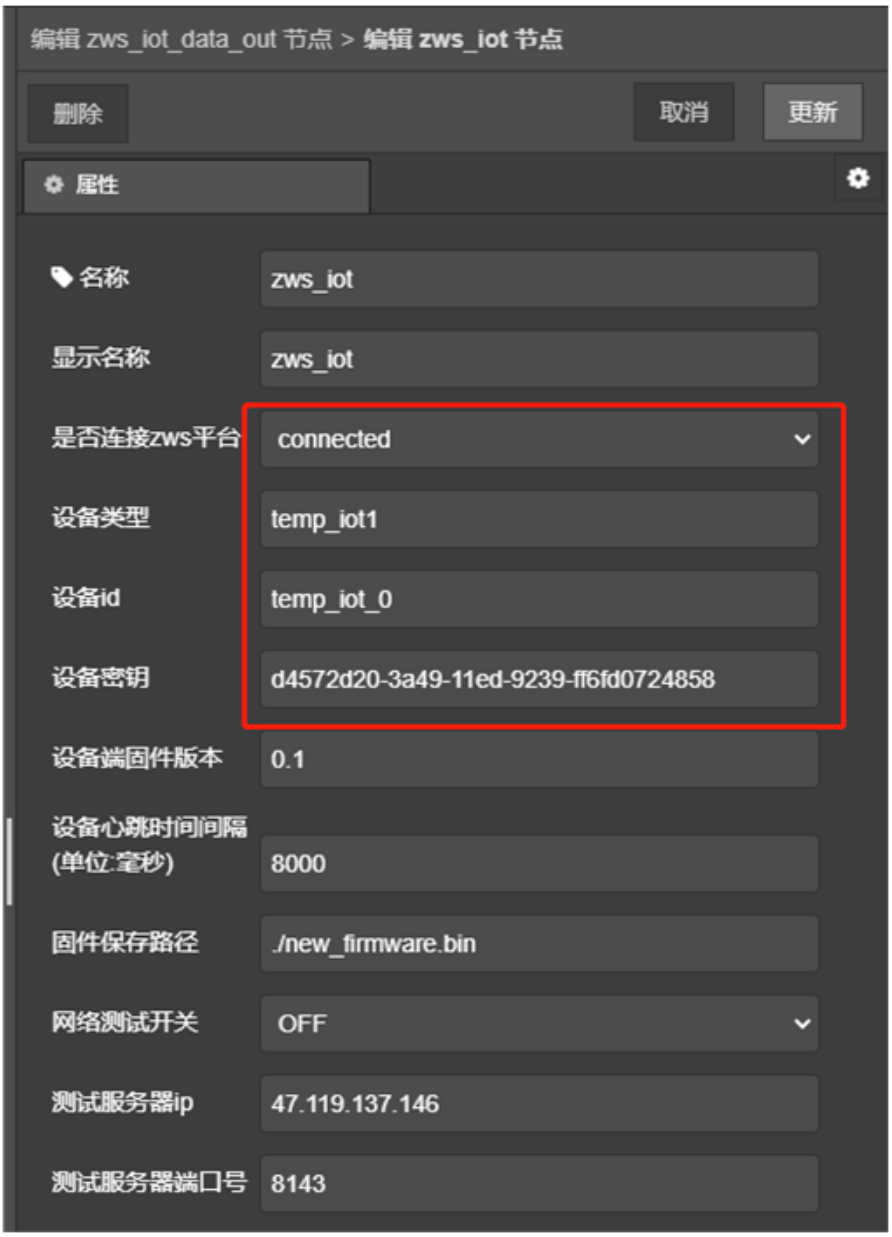

图 4.14　配置 ZWS 云

4.4　温度采集终端界面实现

用 AWStudio 新建一个 MVVM JS 项目，双击打开该项目。进入到 AWTK Designer 后，按照 1.6.2 的步骤将上一章节编辑好的流图导入到项目里，之后开始设计界面。

4.4.1 主界面显示温度值

1. 显示设定温度

根据上文实现的流图，设定温度保存在全局对象 global 中，此处需要将”控件列表”中的”静态文本”控件（即 label 控件）拖到编辑区，并将该控件的”text”属性与流图的”global.setting_temp”数据进行绑定，如下图所示：

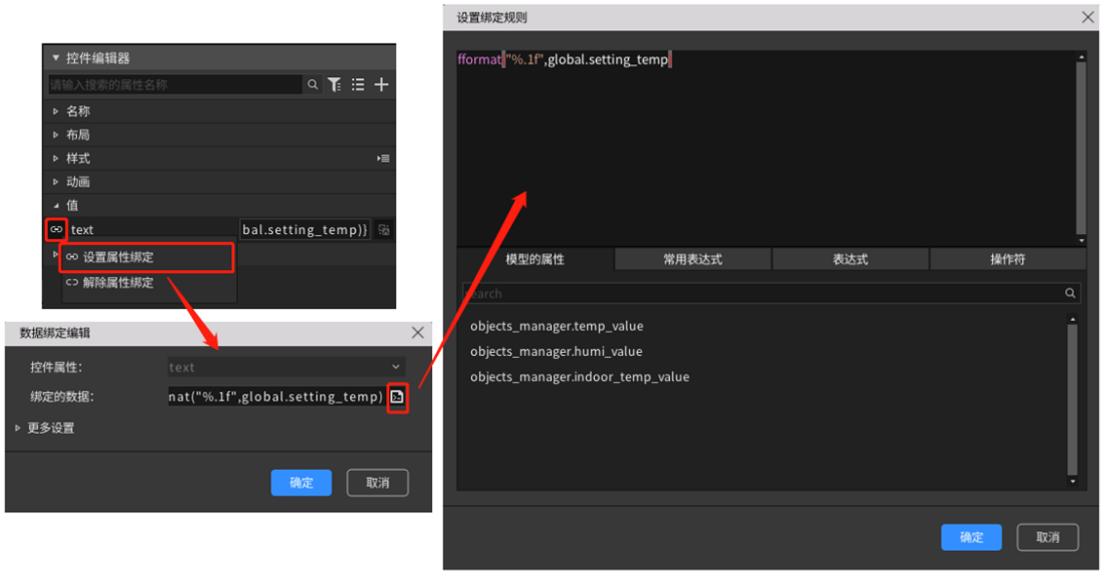

图 4.15 绑定 label 控件的 text 属性

此处需要显示格式化的温度值，绑定代码如下：

```
fformat("%.1f",global.setting_temp)
```

注: 绑定代码中的 _flow 对象是导入的流图对象，它是一个全局对象，通常用于 AWTK 访问流图中的数据。

2. 显示室内温度

根据上文实现的流图，室内温度保存在流图的”indoor_temp”节点中，其绑定步骤与设定温度类似，详见本章第一小节，此处同样需要显示格式化的温度值，绑定代码如下：

```
fformat("%.1f",objects_manager.indoor_temp_value)
```

3. 显示测量温度

根据上文实现的流图，测量温度保存在流图的”temp”节点中，其绑定步骤与设定温度类似，详见本章第一小节，此处同样需要显示格式化的温度值，绑定代码如下：

```
fformat("%.1f",objects_manager.temp_value)
```

4. 显示测量湿度

根据上文实现的流图，测量温度保存在流图的”humi”节点中，其绑定步骤与设定温度类似，详见本章第一小节，此处同样需要显示格式化的温度值，绑定代码如下：

```
fformat("%.1f",objects_manager.humi_value)
```

4.4.2 点击按钮打开设置界面

如下图所示，将”控件列表”中的”按钮”控件拖到编辑区，设置如下：

- 设置显示文本”text”属性为”设置”。
- 切换到”事件”页面，点击右上角的”+“按钮，添加点击事件”click”，并将事件的”动作”属性设置为”打开窗口”以及”窗口的名称”设置为窗口的 UI 文件名”config_page”。

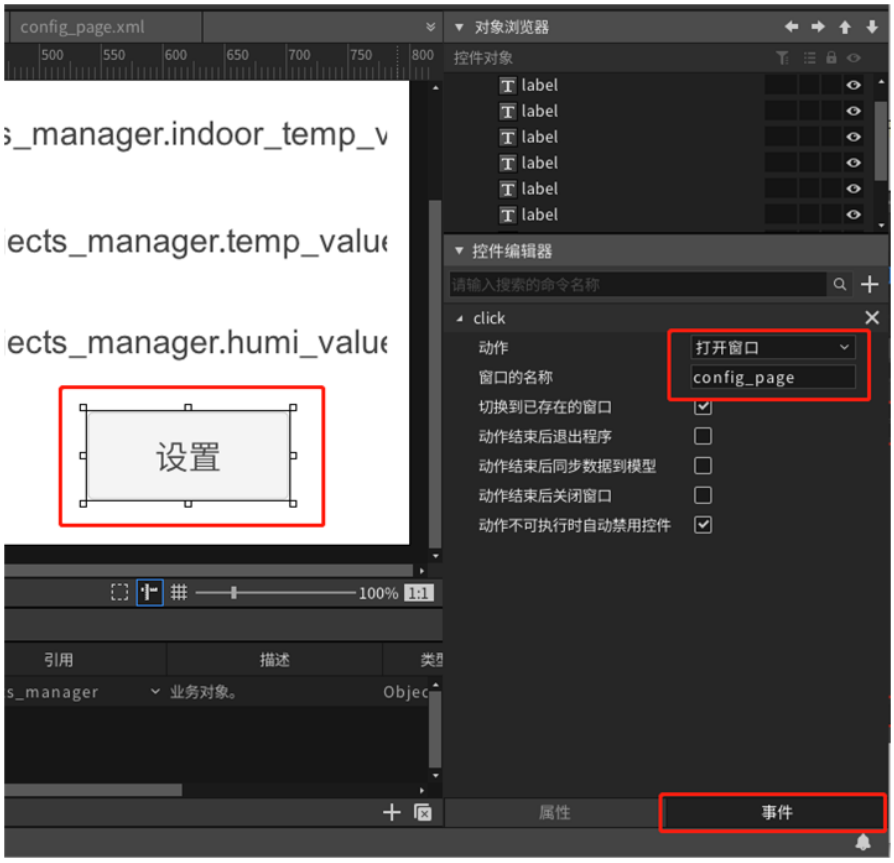

图 4.16 添加点击事件打开窗口

4.4.3 点击按钮上传设定温度

在 AWTK Designer 中新建并打开”Window”类型的设置界面”config_page”，实现点击”确定”按钮上传设定温度步骤如下，最终效果图详见上文 3.1 章节。

1. 将流图数据显示到编辑框

如下图所示，将”控件列表”中的”单行编辑”控件（即 edit 控件）拖到编辑区，并将该控件的”text”属性与流图的”global.setting_temp”数据进行绑定且将其设置为仅视图初始化时绑定（仅绑定一次，即打开窗口时将流图数据读取到界面上），如下图所示：

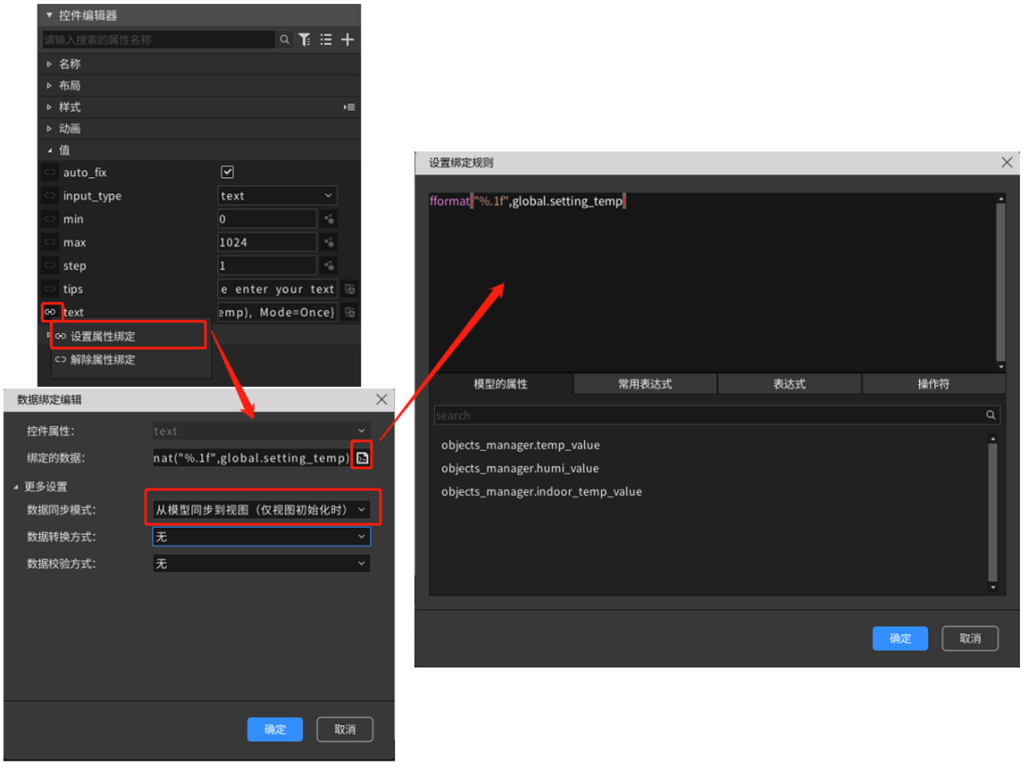

图 4.17　将流图数据显示到编辑框

此处需要格式化设定温度，绑定代码如下：

```
fformat("%.1f",global.setting_temp)
```

2. 将编辑框数据上传至 ZWS 云

根据上文实现的流图，此处只要执行 set_temperature 节点并将编辑框中的温度值传入该节点即可将数据上传至 ZWS 云，因此我们需要给一个按钮的点击事件绑定这个命令，步骤详见下文。

如下图所示，将”控件列表”中的”按钮”控件拖到编辑区，设置如下：

- 设置显示文本”text”属性为”确定”。
- 切换到”事件”页面，点击右上角的”+“按钮，添加点击事件”click”，并将事件的”动作”属性设置为”执行视图模型中的命令”。
- 设置”命令的名称”为”objects_manager.set_temperature_run”。
- 设置”命令的参数”为 edit 控件中的文本，操作如下图所示：

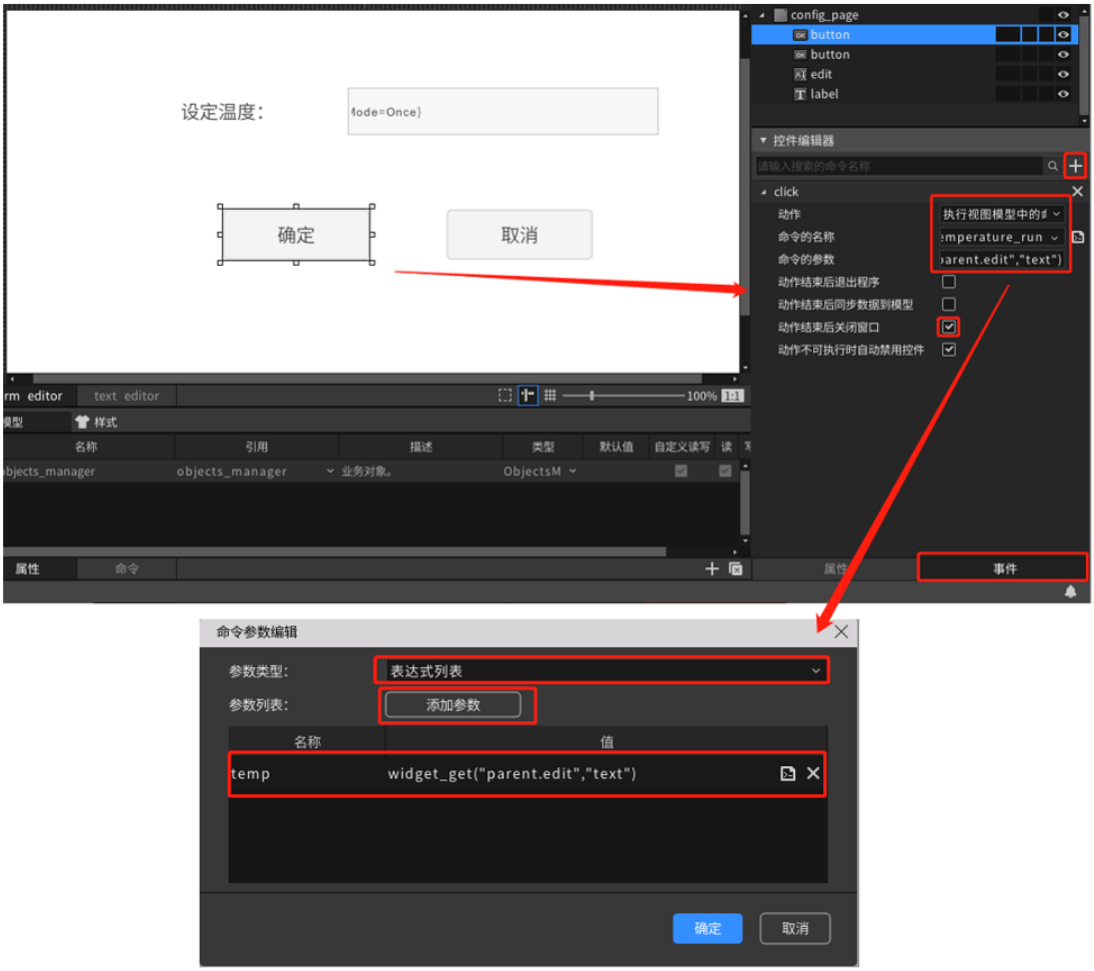

图 4.18　将编辑框数据上传至 ZWS 云

4.5　下载并运行温度采集终端

根据上文完成温度采集终端的业务逻辑流图和界面设计后，我们可以将该应用下载到显控一体机上查看效果，步骤详见下文。

4.5.1　将应用下载到显控一体机

1. 连接显控一体机

将运行有 runFlowAWTK 的显控一体机通过网口连接到电脑，此处主要是让电脑与显控一体机在同一个网段下，比如让二者连到同一个路由器也可以。由于温度采集终端需要将数据上传到 ZWS 云[15]，因此它还要连接到互联网。

电脑与显控一体机连接成功后，可以在 AWStuio 中找到该设备，如下图所示：

[15] https://www.zlgcloud.com

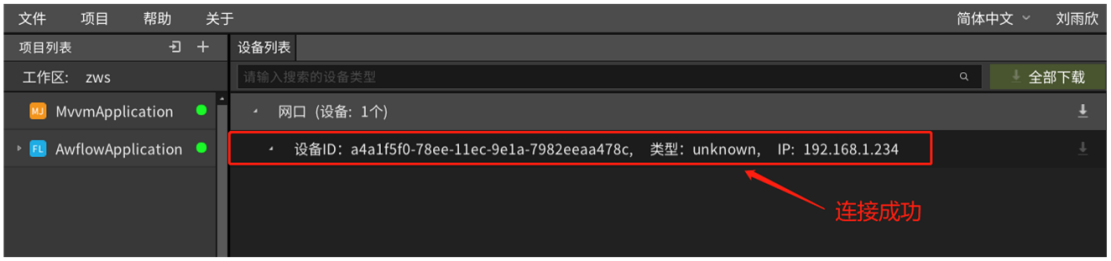

图 4.19　连接显控一体机

2. 下载应用

将 AWStudio 左侧项目列表中的温度采集中 MVVM JS 项目拖拽到项目列表的对应设备上，点击右侧的”↓”按钮，可以将应用下载到设备中，如下图所示：

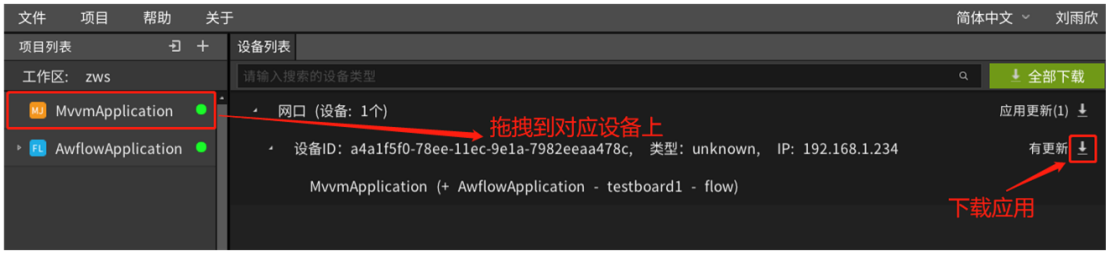

图 4.20　下载应用

4.5.2　查看 ZWS 云的数据

下载成功后，重启显控一体机将自动启动 runFlowAWTK，运行温度采集终端，效果如下图所示：

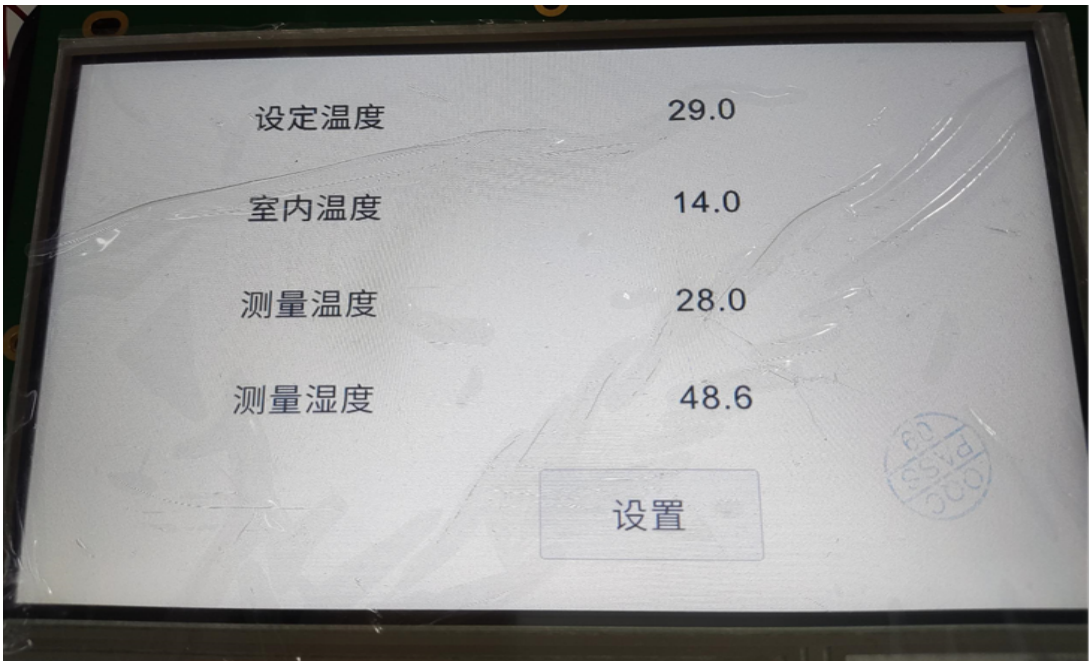

图 4.21　温度采集终端

此时我们可以在 ZWS 云[16] 的设备列表中进入温度采集终端（设备 ID：temp_iot_0）详情页面，如下图所示：

图 4.22　查看设备详情

然后切换到实时数据页面，并选择”data 数据”，可以看到设备上传的实时数据，主要有 temperatuer（逆变器温度）、indoor_temperature（室内温度）和 setting_temperature（设定温度），如下图所示：

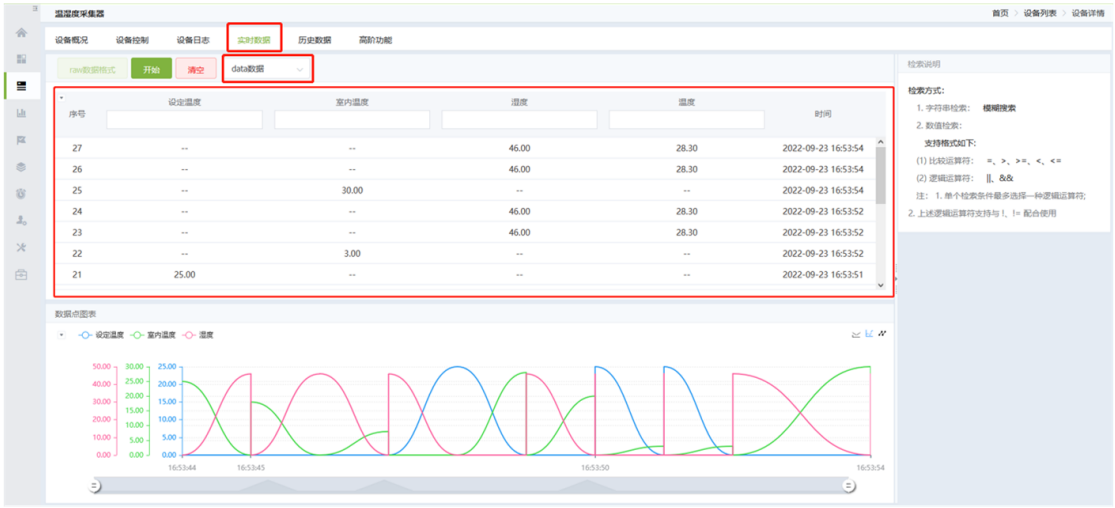

图 4.23　查看实时数据

[16] https://www.zlgcloud.com

诚信共赢，持续学习，客户为先，专业专注，只做第一

广州致远电子股份有限公司

更多详情请访问 www.zlg.cn

欢迎拨打全国服务热线 400-888-4005