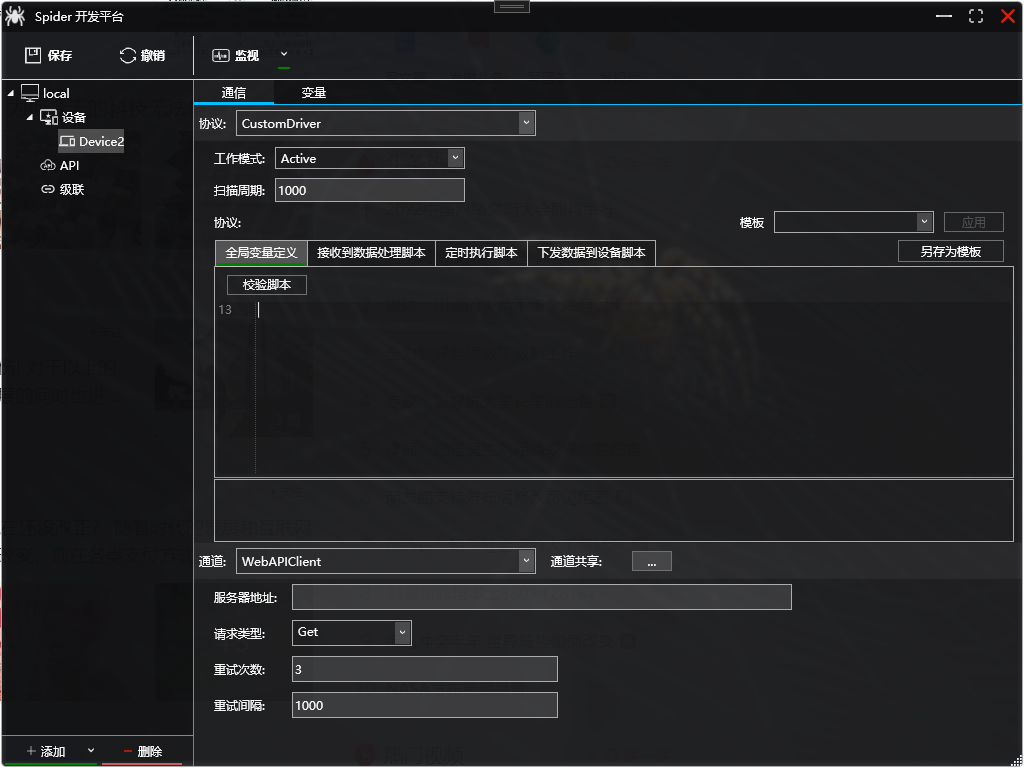
自定义驱动开发文档

Spider中存在一种特殊驱动”CustomDriver”，通过这个驱动用户可以通过嵌入脚本的方式来实现自主协议驱动的开发。它提供了一种简单、灵活的驱动开发的方式，支持的链路通信协议有：MQTT、Coap、WebApi等。

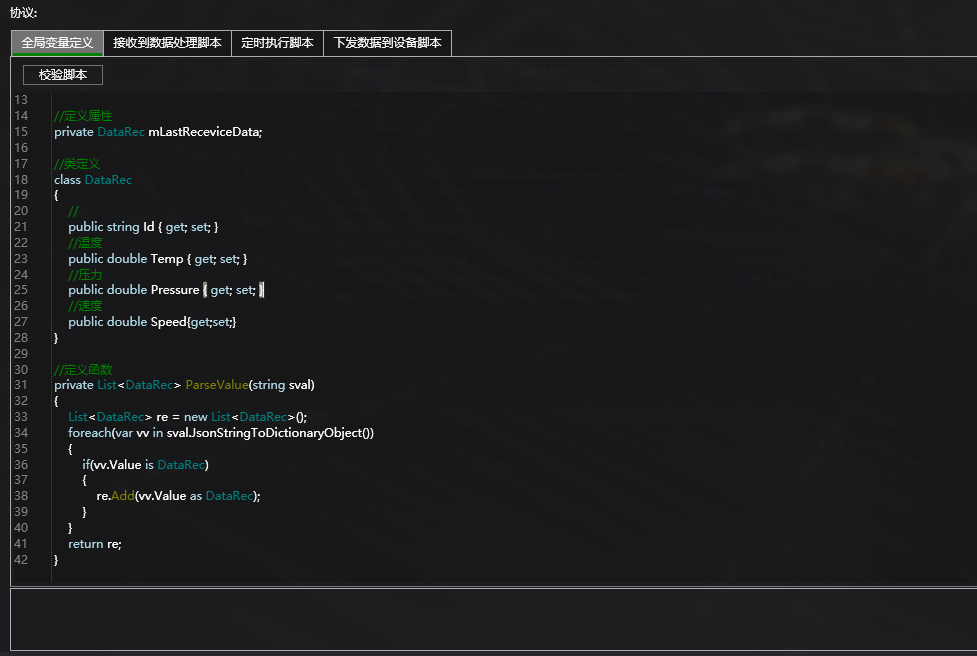


一、协议开发

整个自定义个驱动的开发的界面如上图，分成了“全局变量定义”“接收到数据处理脚本”“定时执行脚本”“下发数据到设备脚本”4部分。CustomDriver驱动将设备通信过程中不同的阶段提取出来，让用户可以通过脚本注入的形式自主控制每一部分，从而完成整个设备驱动的开发。

1. 全局变量定义

此处用于定义一些全局的变量、自定义类、以及自定义函数等。这些定义会在后面的几个脚本中可见并被使用。



1. 接收到数据处理脚本。

当收到设备主动发送数据到设备的时候，会触发该脚本的执行。通过解析data 数据，将每个寄存器的变量的值读取出来，并写入到变量中去。

data 的数据类型依赖通道的上传数据的类型，常见类型有string、byte[] 两种。

解析举例:下面一通过MQTT发送JSON格式的字符串进行通讯的方式进行举例。

string str;

//先将数据转换成字符串

if(data is byte[])

{

str = BytesToStirng(data,Encoding.UTF8);

}

else{

str = data.ToString();

}

//将字符串转换成对象

var data1 = str.JsonStringToObject<DataRec>();

//写入温度

UpdateValue("tag1",data1.Temp);

//写入压力

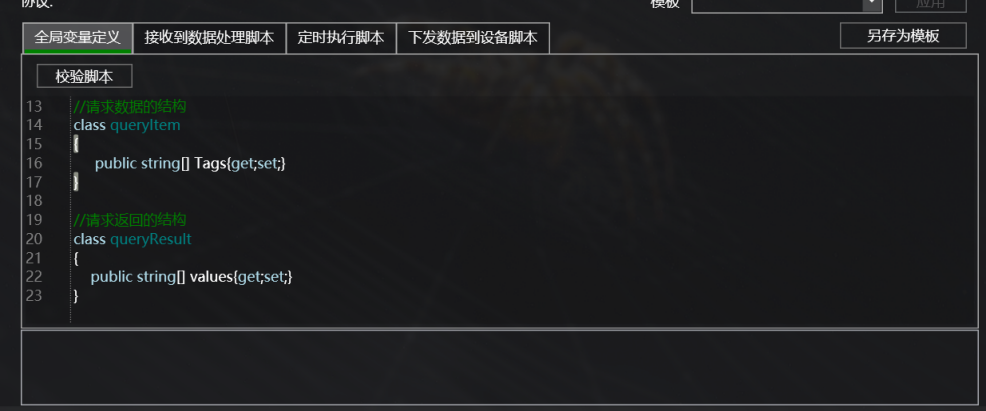
UpdateValue("tag2",data1.Pressure);

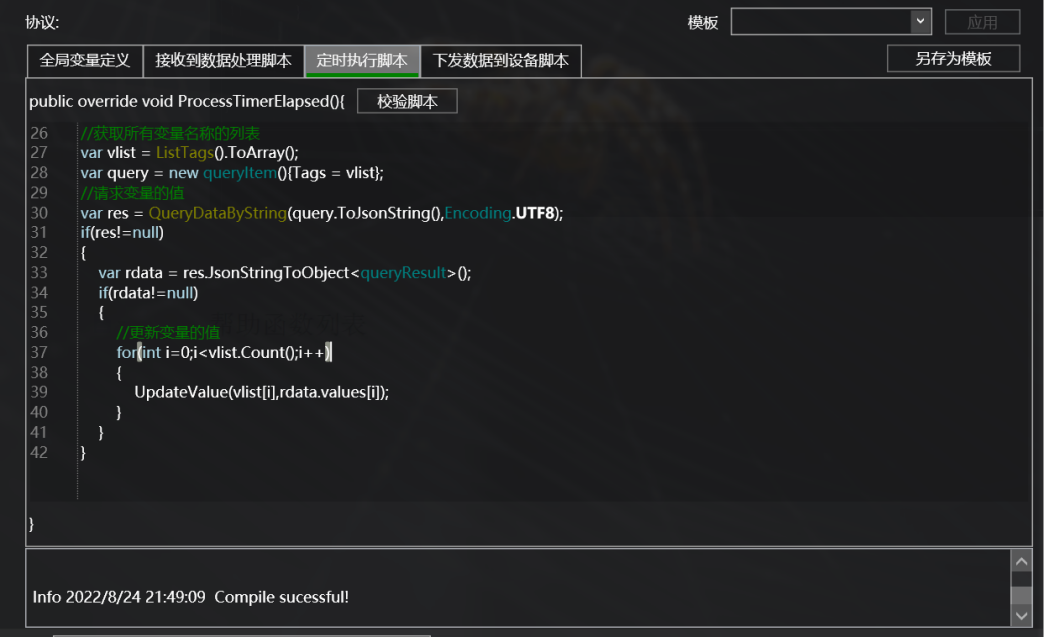
//写入速度

UpdateValue("tag3",data1.Speed);

1. 定时执行脚本

定时执行多用于驱动向设备主动请求数据的方式，我们在此执行请求发送指令。





1. 下发数据到设备脚本

当数据需要下发到设备时，则会进入到此段脚本内。函数定义的格式如下：

public override void WriteValue(string deviceInfo, object value, byte valueType)

{

}

其中 deviceInfo 是变量的名称,value 是值，valueType 是值的类型。我们这里根据设备协议的定义，组织报文然后发送出去。

举例：

我们还是以JSON内容为协议定义的方式，来组织发送数据。我们定义发送数据到设备协议格式如下：

{

“Name”:”name”,

“Value”:”val”

}

我们先在全局变量定义部分先定义一个类对象：ValueSet， 具体如下：

class ValueSet

{

public string Name{get;set;}

public string Value{get;set;}

}

具体实现下发的代码如下：

ValueSet vs = new ValueSet(){Name=deviceInfo,Value=value.ToString()};

vs.Name=deviceInfo;

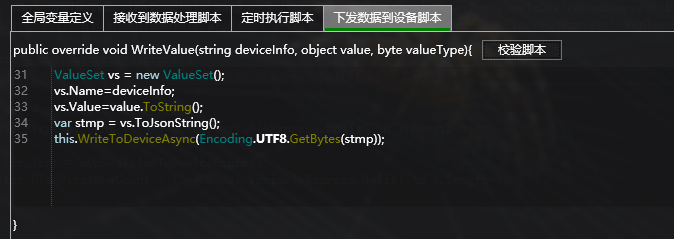
vs.Value=value.ToString();

var stmp = vs.ToJsonString(); //将对象序列化成JSON格式的字符串

this.WriteToDeviceAsync(Encoding.UTF8.GetBytes(stmp));//发送出去

配置如下图所示：





1. 驱动模板

可以将开发好的设备驱动，另存为模板，实现不同自定义驱动间内容的快速共享。

二、帮助函数

为了自义定的驱动开发，Spider内置了一些函数。通过这些函数可以方便、快捷的进行内容开发。具体如下：

1. 主动向设备请求

//通过字符串的形式向设备请求数据

Protect string QueryDataByString(string value,Encoding encoding)

//通过字节数组的形式向设备请求数据

protected byte[] QueryDataByBytes(byte[] value)

//写入数据到设备并等待数据的返回

public byte[] WriteToDeviceSync(byte[] value)

//写入数据到设备无需等待

public bool WriteToDeviceAsync(byte[] value)

1. 数值转换

// Json 字符串转换成对象

protected T JsonStringToObject<T>(string value)

//Json 字符串转换成字典对象

protected Dictionary<string,object> JsonStringToDictionaryObject(string value)

// 将对象转化成Json 字符串

public static string ToJsonString(this object value)

// 如果 values 是 byte 数组 或者 List 以指定编码转换成字符串； 其他类型直接调用ToString

public string BytesToStirng(object values, Encoding encoding)

1. 更新变量的值

//将从设备读取的值写入到变量

protected void UpdateValue(string deviceInfo, object value)

1. 辅助功能

//获取该设备定义的变量名称列表

public IEnumerable<string> ListTags()

// 打印日志 Info

protected void Info(string msg)

// 打印日志 Erro

protected void Erro(string msg)