## 虚拟联调

### Unity3D设置

**（1）HslCommunication插件**

HslCommunication是一个工业物联网的底层架构框架，专注于底层的技术通信及跨平台，跨语言通信功能，实现各种主流的PLC数据读写，实现modbus，机器人的各种协议读写等等， 支持快速搭建工业上位机软件，组态软件，SCADA软件，工厂MES系统，助力企业工业4.0腾飞，实现智能制造，智慧工厂的目标。

官方网站：http://www.hslcommunication.cn/，包含组件的在线API地址以及一个MES DEMO的项目展示。

**（2）通讯脚本**

* 1.实例化一个西门子的S7协议的通讯对象：

public SiemensS7Net(

SiemensPLCS siemens

)

* 2.实例化一个西门子的S7协议的通讯对象并指定ip地址：

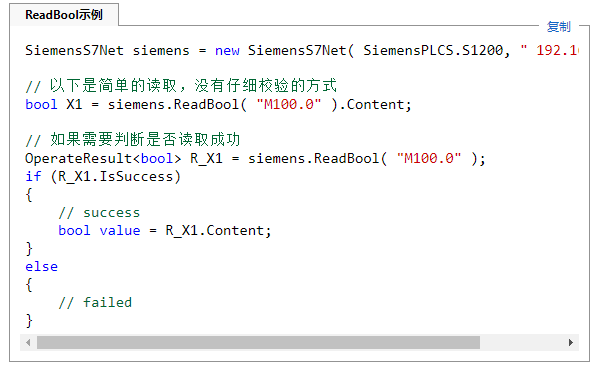
Public SiemensS7Net(

SiemensPLCS siemens,

String ipAddress

)

* 3.读取PLC数据可以选用Read、ReadBool、ReadByte及ReadString等方法，此处以ReadBool方法为例展开介绍。



**图1.1 通讯脚本**

### 协程

协程，从字面意义上理解即协助程序的意思，在主任务进行的同时需要一些分支任务配合工作来达到最终的效果。

在编写通讯脚本过程时，由于上述的通讯操作需要实时性、不间断，因此对于资源的消耗较大，如果每一帧中都实现这样的操作，Unity的Game界面将会变得十分卡顿，因此需要通过协程在一定帧内完成通讯工作，从而避免影响主任务的进行。

**（1）协程的原理**

协程并不是线程，协程依旧是在主线程中进行。此外，协程是通过迭代器来实现功能的，通过关键字IEnumerator来定义一个迭代方法，注意使用的是IEnumerator，而不是IEnumerable，两者之间的区别在于：IEnumerator是非泛型的，也是协程认可的参数；IEnumerator通过泛型实现的迭代器，协程不适用该迭代器。

在迭代器中最关键的是yield的使用，这是实现协程功能的主要途径，通过该关键方法，可以使得协程的运行暂停、记录下一次启动的时间与位置等等。

**（2）协程的使用**

//通过迭代器定义一个方法

IEnumerator Demo(int i)

{

//代码块

yield return 0;

//代码块

} //在程序种调用协程

public void Test()

{

//第一种与第二种调用方式,通过方法名与参数调用

StartCoroutine("Demo", 1);

//第三种调用方式， 通过调用方法直接调用

StartCoroutine(Demo(1));

}

**图1.2 协程脚本**