链滴

node-red【函数】

作者：pinruxiutai

原文链接：https://ld246.com/article/1618490999845

来源网站：链滴

前言：

介于我学习node-red的艰难进展在此以我觉得最通俗的方式解释下函数里常用的代码含义及使用方法

附加一个说明：node-red使用的是js语言就是JavaScript在JS里“=”是赋值，“==”是判断是否相等不判断类型一样，而“===”则是判断是否相等也判断是否类型一样。

msg.payload：

我对于【msg.payload】的理解是控件输出的东西，可以理解成上一个控件的对函数输入的内容的称呼

例：

```
if (msg.payload == 'a'){
    msg.flag = 1//如果输入的值为a 则msg.flag就是1
}
```

var：

对于【var】呢其实就正常理解就可以了.他就是一个定义我可以用它定义很多东西下面是两个个例子

```
var x //定义一个变量x
var x = msg.payload //定义x=上一个控件输入的信息
```

用法还有很多就不一一列举了

if：

对于【if】就更好理解了，if的汉译就是如果

和大部分编程一样node-red的if也是if+(条件){ 成立则执行***}和上面一样下面是例子

```
var x
if (msg.payload == 10){
    x = msg.payload //定义一个x 如果输入=10则x=输入（也就是10）
}
```

global：

【global】一个神奇的全局变量，当我卡在两个输入进行比较的时候是【global】给予的灵感

【global】的意思不便描述直接上示例

```
global.set('x',msg.payload) //定义全局变量（set大概就是规定的意思）x 是输入来的东西
```

x可以是任何东西任何名，经过以上就定义了x这个全局变量，那我定义的全局变量有什么用呢？例子

下

```
global.get('x') //获取全局变量（get大概意思就是取；获取）x
```

那获取到x后有什么用呢？继续例子

```
var a = parseInt(global.get('x')) //这次定义一个仅限于这个函数的变量a这个a就是获取到的全局变x有了a我们就可以
if （msg.payload > a）{
    msg.flag = 10 //如果输入的值 > a 那么msg.flag就为10
}
```

注意！！！当使用【global】时我定义x的值的代码可以放在我的函数1里，而我获取x可以在函数2获取而且无需讲函数1与函数2连线即可获取

context:

【context】大意就是上下文，他可以取一个上文的量也可以保存一个上文的量，使用方法跟global不多，但是他和global的差别也很明显他只能在一条函数里获取和保存，二global可以跨函数。例子

```
var x = context.get('x')||60 ;
//读取x的值如果没有保存过x的值则设定成0
if (msg.payload == 'a'){
    x += 10;
    //当输入为a时x的值+10
}
context.set('x' , x);
//保存x的值=x
```

Buffer:

【Buffer】这是一条数据，（字符串）通常用来发送一组有规律的数和协议，用法如下，重点！！！符串的数据位是从0为起点 即:第一位是x[0] 第二位是x[1] 第三位是x[2] 以此类推

```
var x =  new Buffer([1,0,1,0,1,0])//定义x=字符串
if (msg.payload == 1){//如果输入为1
    x[1] = 1//x字符串的第二位就为1
    msg.payload = x//输出等于x
    return msg//此时连接debug会得到([1，1，1，0，1，0])这条数据
}
```

多输出

函数控件可以在setup选项里更改输出路数，当输出为两个及以上时可以对每个口输出的东西进行设下面以三输出为演示

```
var msg1 = {payload：1}
var msg2 = {payload：2}
var msg3 = {payload：3}//定义各个输出
if (msg.payload == 1){
    return [msg1,msg2,msg3]//当输入为1对应1 2 3口分别输出1 2 3
```

```
}
if (msg.payload == 2){
    return [msg1,null,msg3]//当输入为2时对应1和3口分别为1和3而二口为“null”即“空”
}
```

后记：

欢迎更正，此文章处于更新中……