**UiPath手把手教程十二-循环结构二(ForEach、Break和Continue)**

大家好这里是Python知识学堂。

作者在RPA领域工作已经两年多，看到UiPath产品也在不断的完善，有必要分享一下学习和使用UiPath的一些经验，帮助小伙伴踩坑。文章中有什么不对的地方还请指出，一起学习一起进步。接下来我们开始学习：

本次我们学习的内容是关于控制语句的相关知识，控制语句在任何编程语言中都是需要掌握的。一些小伙伴说自己的Uipath界面语言是英文的，不习惯中文的，因此在今后的教程中我们也直接使用英文界面的Uipath。

上期文章我们介绍了UiPath中循环语句的相关知识，介绍了Switch和Switch Case 的使用，具体文章可从以下链接获取:

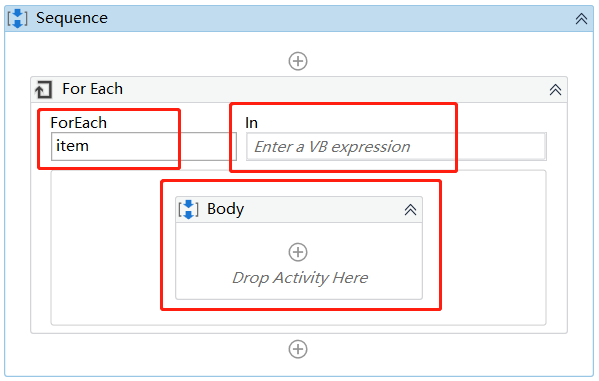
XXXXXXXXXXXXXXXXXXXXXXXXXXXXX

此次文章我们一起来学习一下UiPath中其他用于控制循环的。

**一、For Each活动**

For Each也是一种控制活动，一般来说我们使用For Each去遍历集合中的元素。比如说我们遍历一个列表对象中的元素，遍历一个Excel中的每一行数据等等，这些都可以使用我们的For Each活动来完成。

UiPath中的For Each 活动的样式如下所示：



可以看出For Each 主要分三个部分。ForEach部分、In部分和Body部分，这三个部分的作用如下：

ForEach: 用于遍历元素。

In:要进行遍历（循环）的变量。

Body:对遍历出来的元素进行遍历操作。

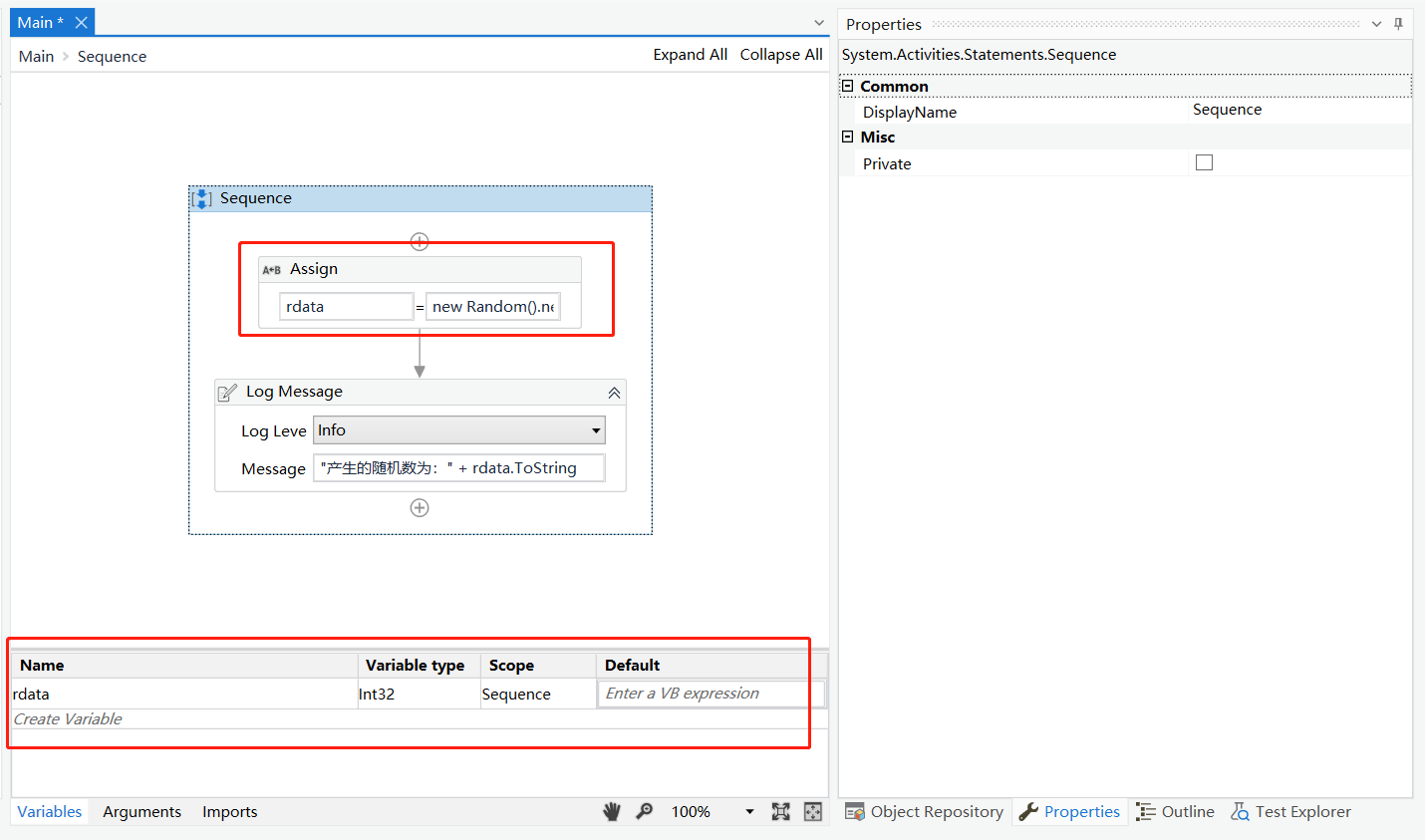
接下来我们看一个例子，使用ForEach打印的数字的流程，步骤如下：

1. 新建一个空白流程，命名为“ForEach\_Test”,之后拖动一个Sequence到主界面中。之后在此基础上拖动一个Assign活动，并输入一下内容：

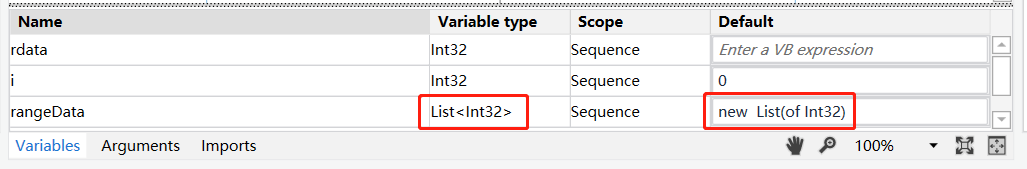
rdata = new Random().next(1,10)

这句表达式的作用是产生1-10之间的某一个数值，然后将这个数赋值类型为int32的变量rdata。

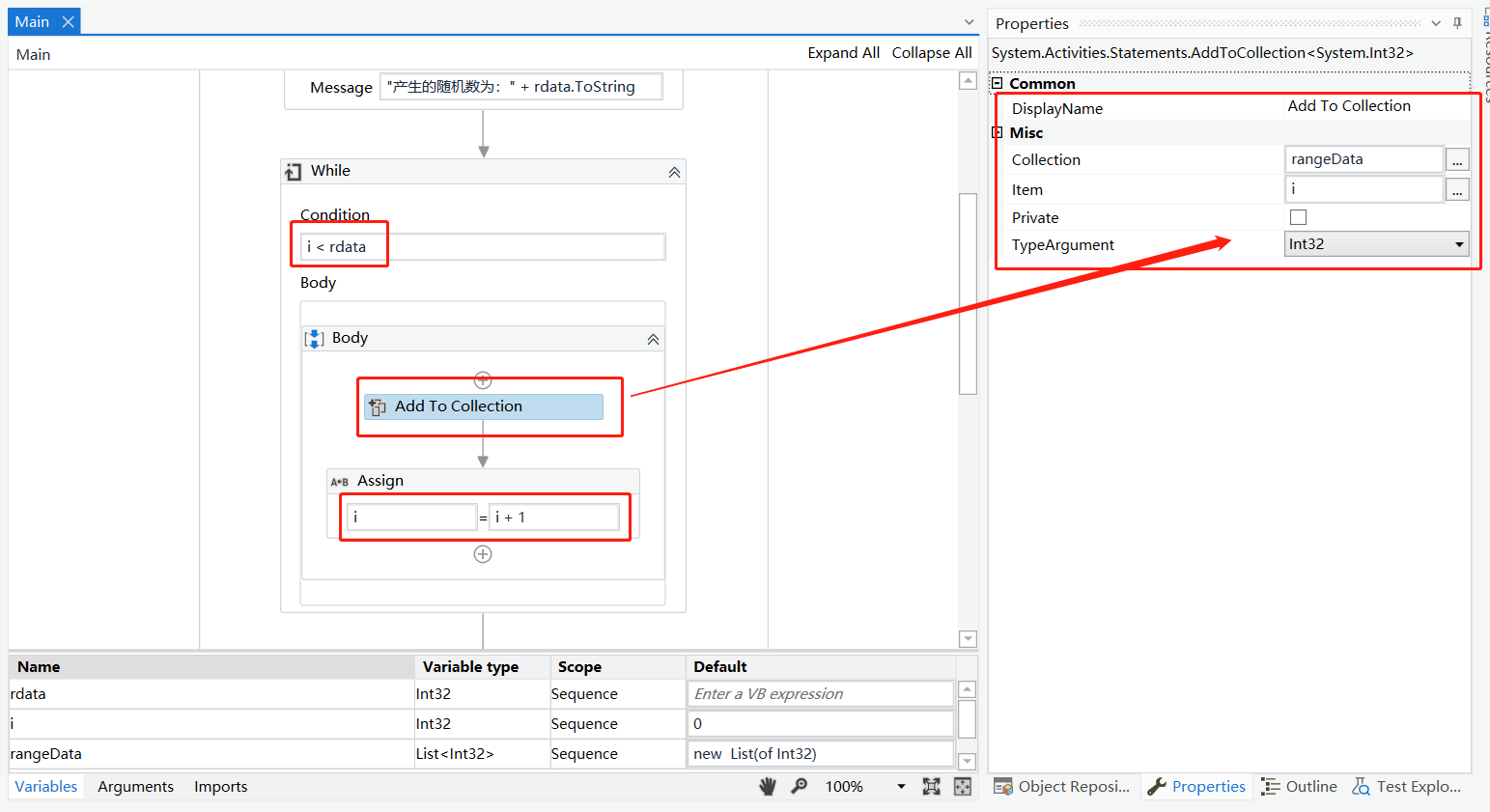
1. 注意在赋值的时候，我们要在变量面板中设置变量（即主要先定变量后使用变量），之后我们再拖动一个Log Message活动到Assign下，并在Message输入框中输入以下内容："产生的随机数为：" + rdata.ToString。



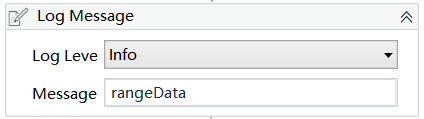
1. 由于之后我们还是需要一些变量的，因此我们这里先建一个int32类型的变量i默认为0,和一个List<int32>类型的列表变量rangeData，rangeData我们赋予默认值，即rangeData = new List(of Int32)。



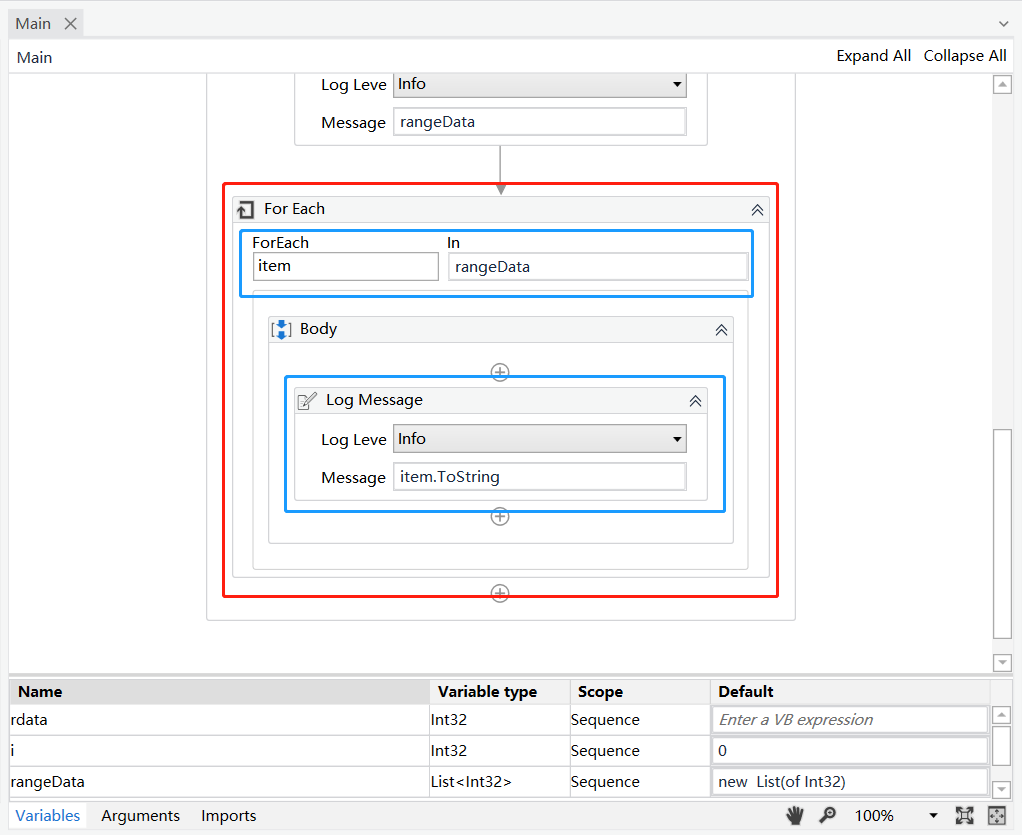
1. 变量建立好之后，我们就继续后续的操作，在上述的Log Message活动下拖动一个While活动。设置Condition为i<rdata，Body中我们拖动一个Add To Collection，并在Properties属性面板中设置相应的参数值其中Collection:rangData, Item:i, TypeArgument:Int32。之后我们使用Assign来对变量i进行增加1的操作，具体如下所示：



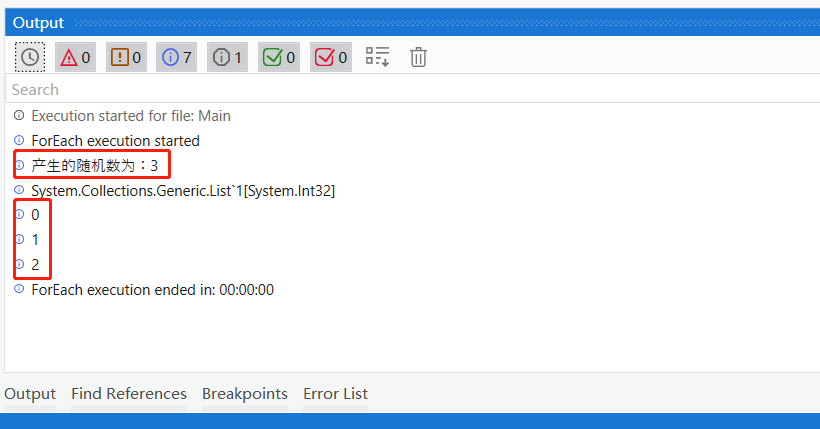
1. 上述第（4）个步骤的作用就是以随机产生的一个数,然后将0到这个数的所有整数都存于一个名为rangeData的列表中。上述步骤完成以后，我们要做的就是使用ForEach循环来取数。我们拖动一个Log Message到While下，打印一下rangeData。



1. 之后我们在第（5）步骤的Log Message下拖动一个ForEach活动，并按照下图所示进行一些参数设置，其中的ForEach...in部分就是遍历rangeData中存储的数据。



1. 上述设置完成以后，我们来执行一下流程，查看输出情况：



可以看出本次产生的随机数为3，并且依次打印了其中rangeData中的数。由于本流程我们使用了随机数，因此我们每一次运行可能的结果是一样的。

1. **Break活动**

在循环中，我们经常遇到一种情况：我们对一个列表的元素进行遍历的时候，当我们正好遍历到某一种元素的时候，就不想再遍历之后的元素了。比如说现在有一个列表mylist = {"这里","是","Python","知识学堂"}

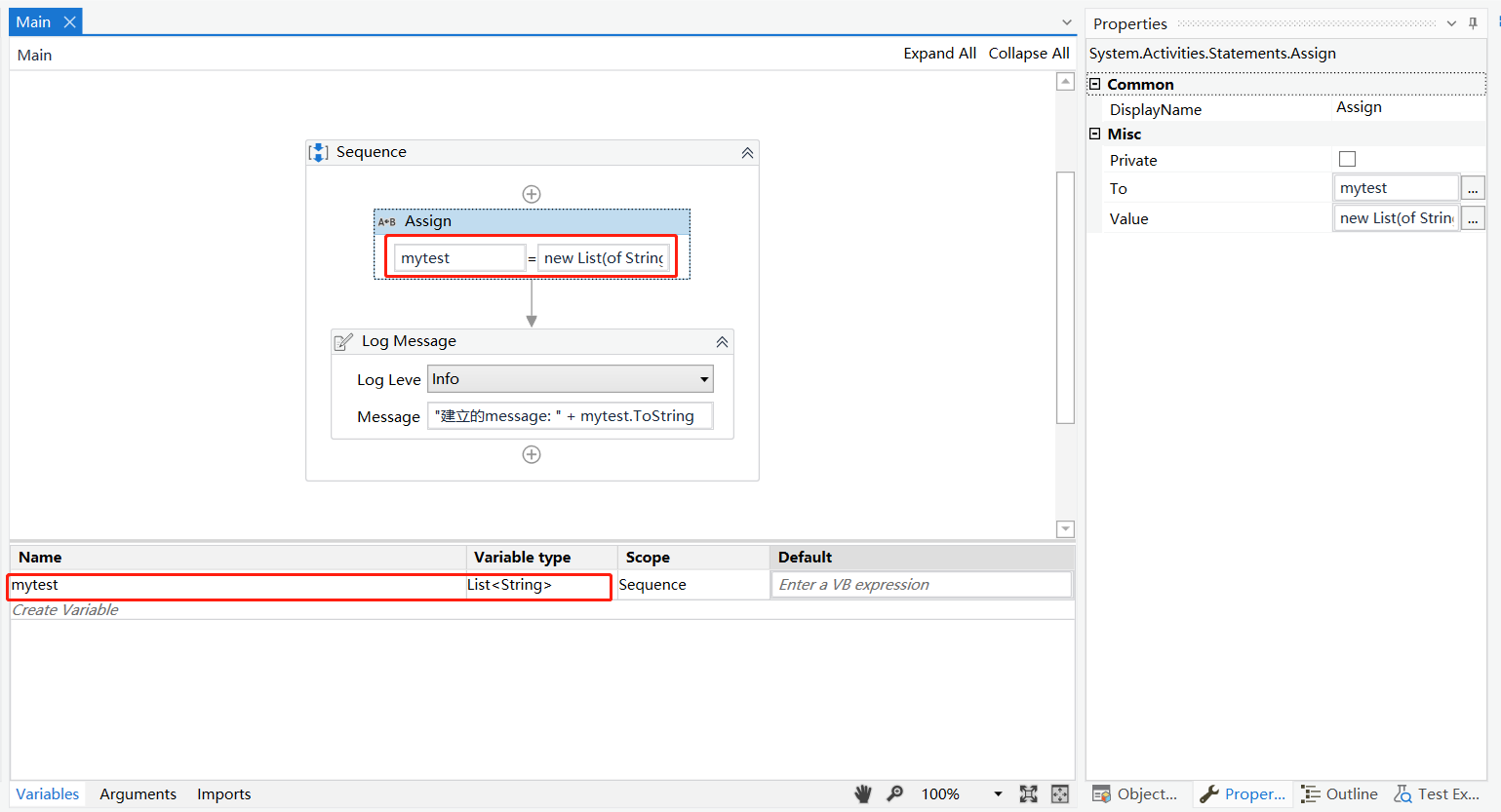
当我们使用ForEach去遍历里面元素的时候，如果遍历到"Python"之后，就不想再继续后面的遍历，此时我们该怎么处理呢？

对于这种需求，我们使用Break活动即可完成我们的需求。接下来我们看一下在UiPath中怎么实现Break的功能，步骤如下：

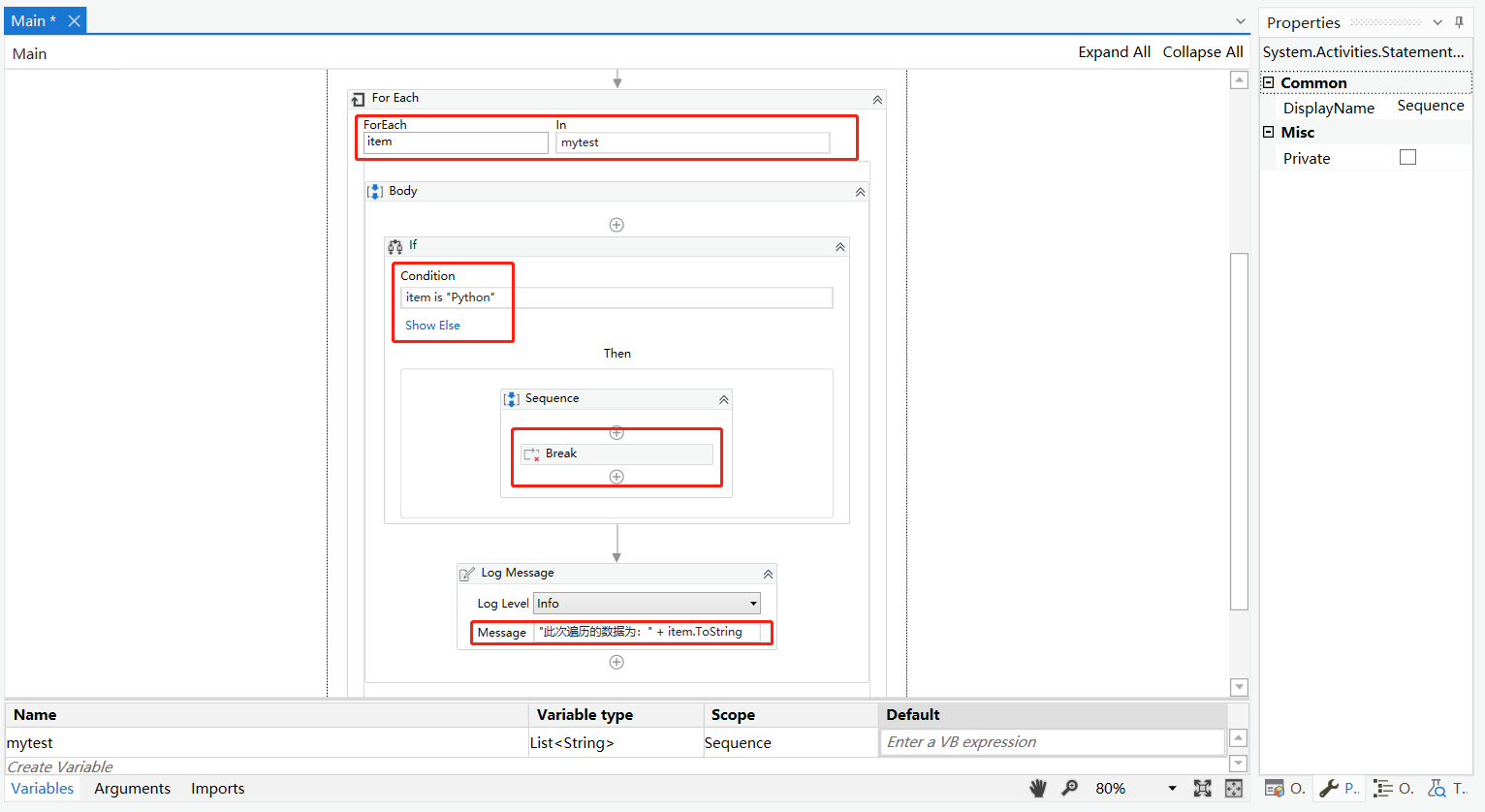
1. 先建立一个新的流程，然后拖动一个Assign活动到主界面中，内容输入：

mytest = new List(of String) From{"这里","是","Python","知识学堂"}

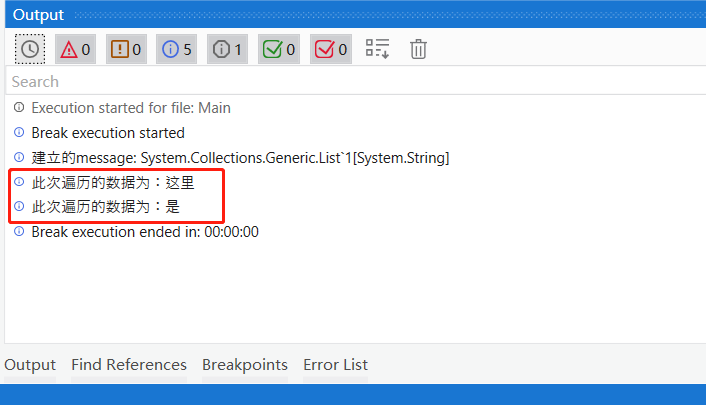
同时我们在变量面板中建立一个List<String>类型的变量mytest。之后我们在此Assign下新建一个Log Message打印一下建立的变量，具体的设置如下图所示：



1. 建理好了变量之后，接下来我们的操作就是遍历列表，这里演示Break的功能，所以我们在遍历到元素“Python”之后就停止遍历，我们要看一下怎么实现。拖动一个ForEach设置遍历的对象为mytest，在ForEach的Body中我们先拖动一个if语句到Body中，Condition条件设置为item is “Python”,Then部分中我们拖动一个Break活动置于其中，最后我们在if活动下增加一个Log Message，并设置Message的内容为："此次遍历的数据为：" + item.ToString。具体设置后的各个活动情况如下所示：



1. 我们运行建立好的流程，输出截图如下所示：



可以看出上述流程已经满足了我们的需求，指定之后的元素以及其之后的元素将不在打印出来。

1. **Continue活动**

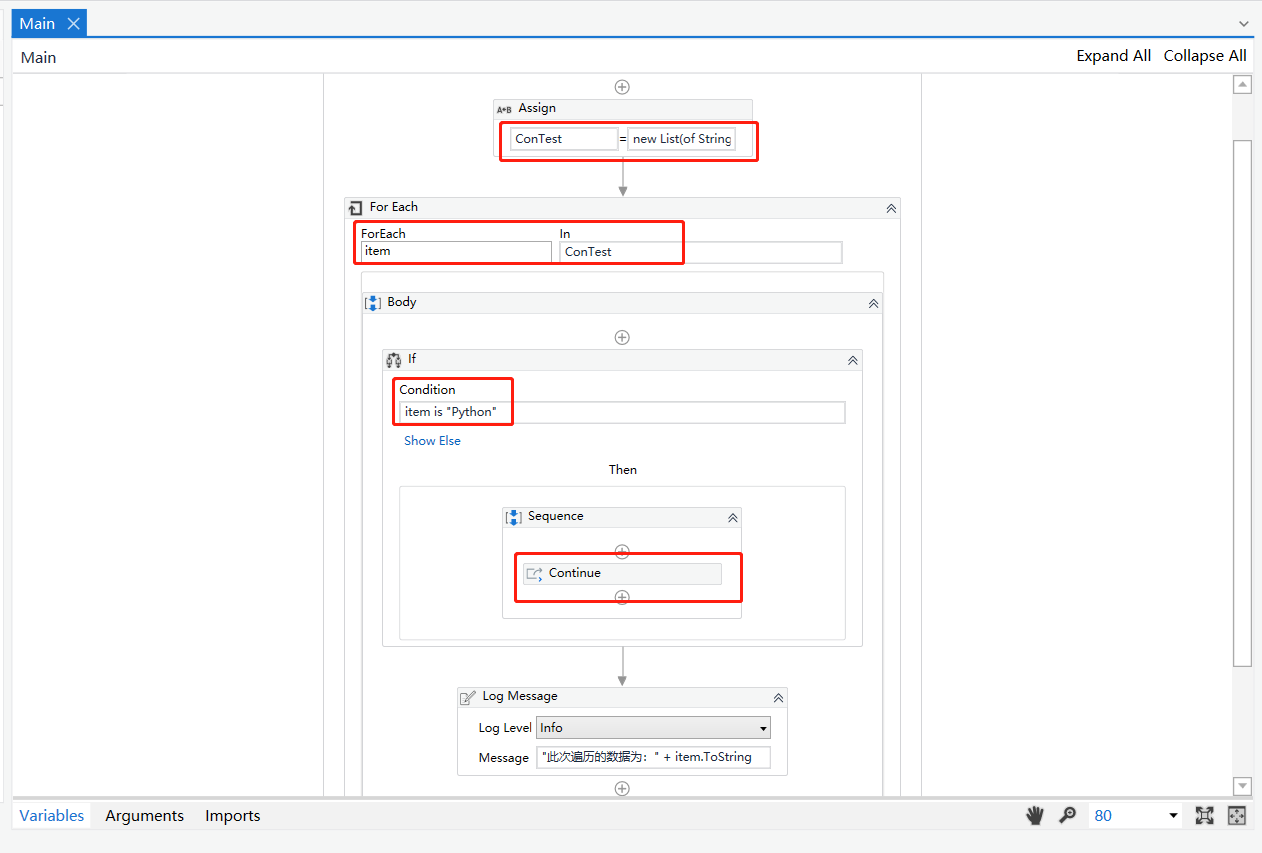
在UiPath中还有一种叫做Continue的活动。这种活动是干嘛的呢？也是控制循环进行的一种活动。这里我们来看一个例子大家就明白了:

（1）新建一个流程命名为ContinueTest，然后再主界面中拖动一个Assign活动到主界面中，内容输入：

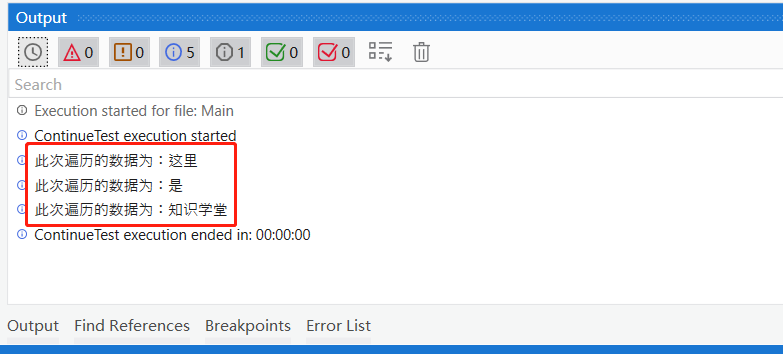
ConTest = new List(of String) From{"这里","是","Python","知识学堂"}

同时我们在变量面板中建立一个List<String>类型的变量ConTest 。

1. 之后我们的操作与上述Break的案例一致，只是将if活动中的Break活动替换为Continue活动，如下所示：



1. 接着我们运行上述的流程，查看一下输出情况：

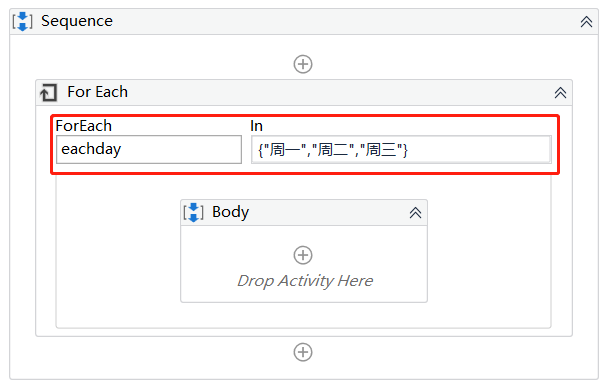


可以看出，使用Continue与使用Break的区别很明显，Continue可以继续后面的循环，而Break直接截止了后面的循环。在实际编程的时候这两点很有用，我们在开发循环的相关流程中，把握好这两点可以很好的让流程更加灵活。

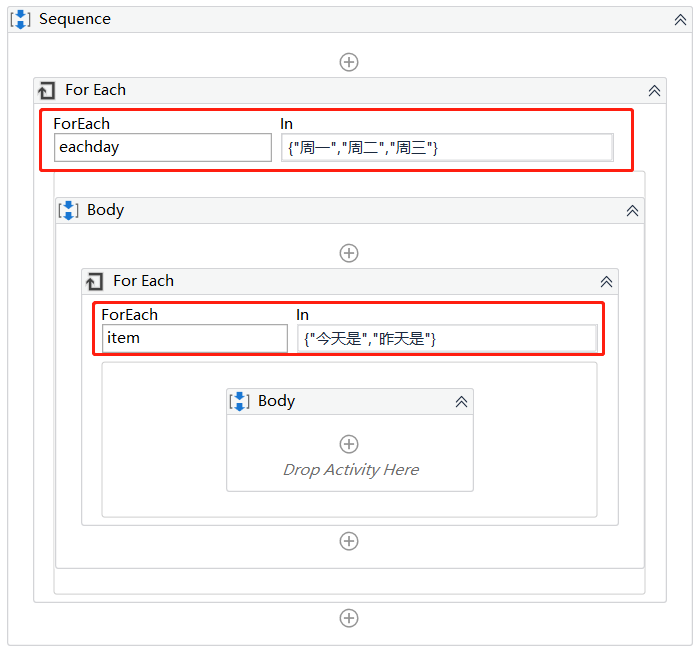
1. **嵌套循环**

在实际的开发过程中，我们经常会使用到两个循环、三个循环等等。这就是我们所说的嵌套循环。这里我们演示一个简单的案例：

1. 新建一个流程DoubleForEach,拖动一个ForEach置于主界面，ForEach中输入eachday,在in中输入{"周一","周二","周三"}

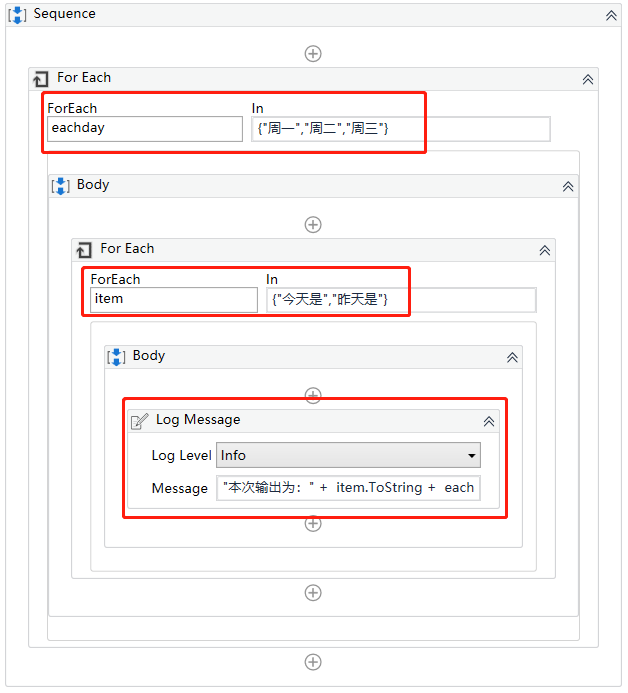


1. 之后我们在拖动一个ForEach到上述的ForEach的Body中，并设置相应的数据，如下所示：

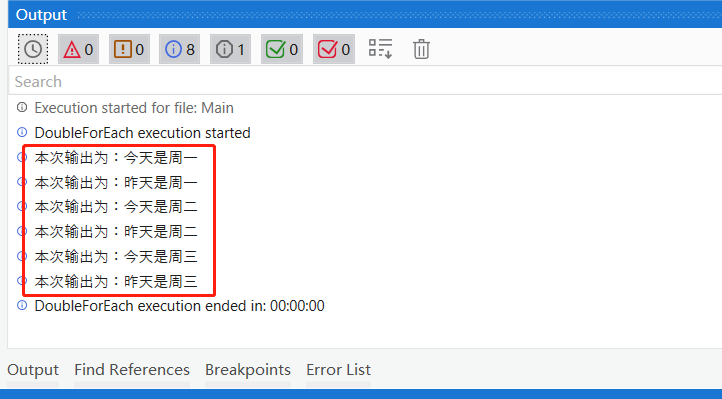


1. 之后我们加入一个Log Message打印一些日志信息，Message填写的内容如下所示：

"本次输出为：" + item.ToString + eachday.ToString



1. 完成上述步骤之后，我们简单的流程就完成了，接下来我们运行一下代码查看一下实际的结果：



至于此我们的嵌套的ForEach循环就完成了，这里是双层循环，在实际的业务中我们还可以编写三层和更多层的循环。

1. **总结**

本次文章主要是关于UiPath中的循环活动，具体介绍了ForEach、Break和双重ForEach，也通过例子向大家演示了在UiPath中该怎么使用这三者，希望大家能明白每一个活动的使用方法。以上就是本次文章的所有内容，我们下次再见。