**UiPath手把手教程七-数据类型基本知识**

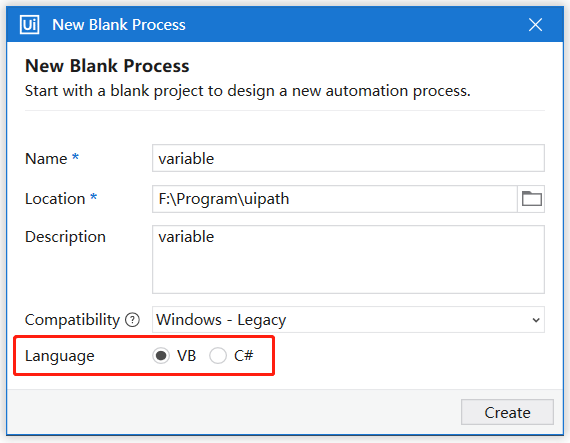
大家好这里是Python知识学堂。

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。

上期文章我们介绍了UiPath工作流中的状态机（State Machine）相关的知识,并使用一个数字猜测的例子来说明了如何使用State Machine来构建流程，大家可以温习一下。本次文章我们来学习一下UiPath中数据结构的相关知识。

1. **UiPath 编程语言**

UiPath支持的语言目前有VB.NET和C#两种语言，我们在新建一个空白的流程的时候可以从面板中看出我们可以选择的语言，如下所示：



在建立新流程的时候，如果指定了某一种语言，那么只能使用该语言中数据类型去创建相应的变量。

**二、UiPath中的数据类型**

跟C++,Python等语言一样，UiPath也有数据类型的概念，不同的数据类型会产生不同的存储结构，这也就会使得属于不同数据类型的变量的用法也会存在一定的差异。

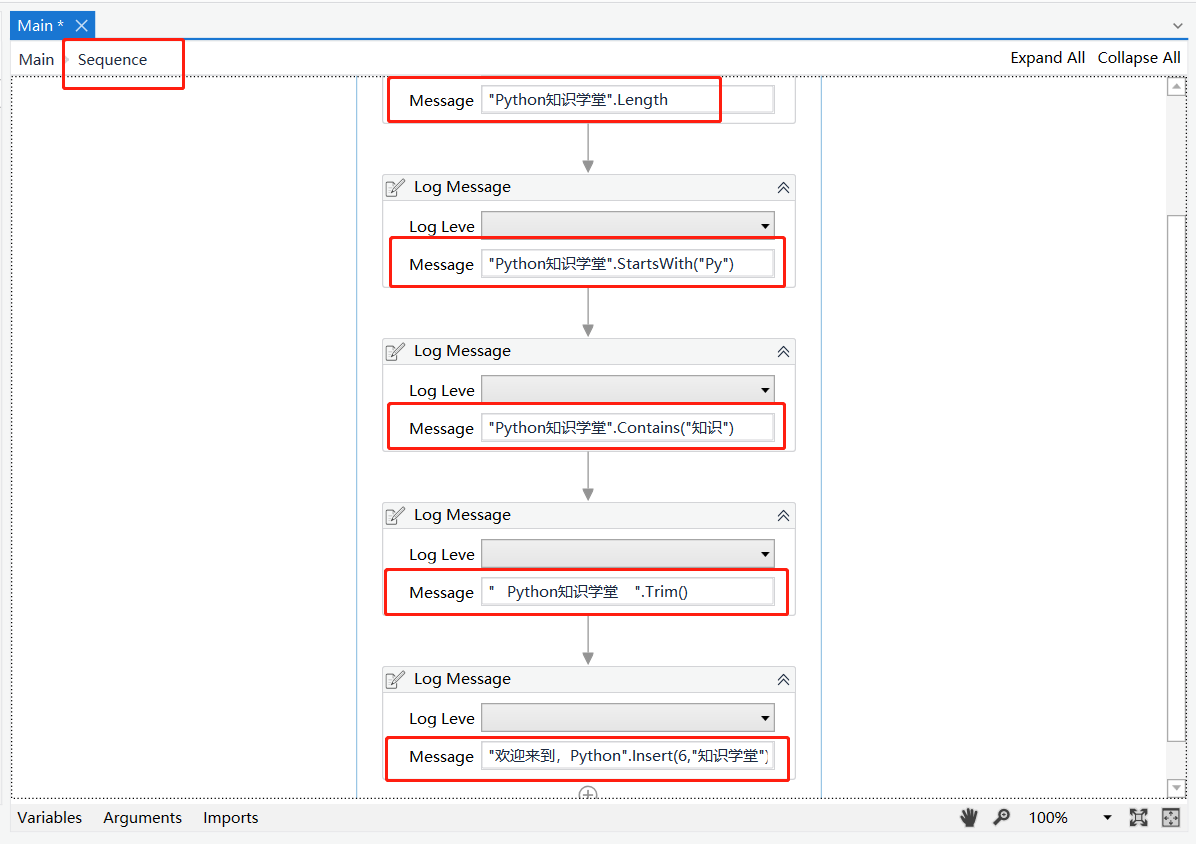
UiPath支持很多种不同的数据类型结构，这里我们介绍一些常用的数据结构：

**2.1 字符串类型**

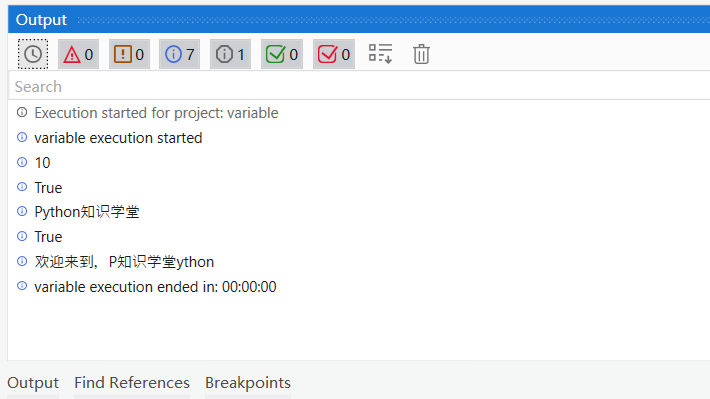
字符串类型String也即文本类型，一般用于存储固定的字符串，比如说在RPA流程中使用字符串来存储客户的姓名、身份证所属省市等等。UiPath中的所有字符串都必须放在英文的引号之间，之后大家可以从例子中查看到这一细节，这里列举几个在UiPath中常见的字符串处理方法：

1. Length:获取字符串的长度。
2. StartsWith():判断一个字符串是否以某一个字符串开始,如果是则返回True,反之返回False。
3. EndsWith():判断一个字符串是否以某一个字符串结束，如果是则返回True,反之返回False
4. Contains():判断一个字符串是否包含另一个字符串，包含返回True,反之返回False
5. Trim():去掉字符串左右的空格。
6. Insert(inx,str):在源字符串inx位置插入字符串str.
7. 其他方法等等。

我们来使用几个Log Message来打印几个看看：



输出的结果为：



可以看出，字符串类型的变量在使用的时候是需要使用英文双引号引起来的，也就是说使用英文双引号引起来的变量我们就认为它是字符串。当然，字符串还有很多其他的方法，这里就不在演示了。

**2.2 布尔型类型：**

用于存储True或者False变量，主要用于判断做出决策，从而更好的控制流程。

这里我们拿上述的例子输出来说明：

"Python知识学堂".StartWith("Py")

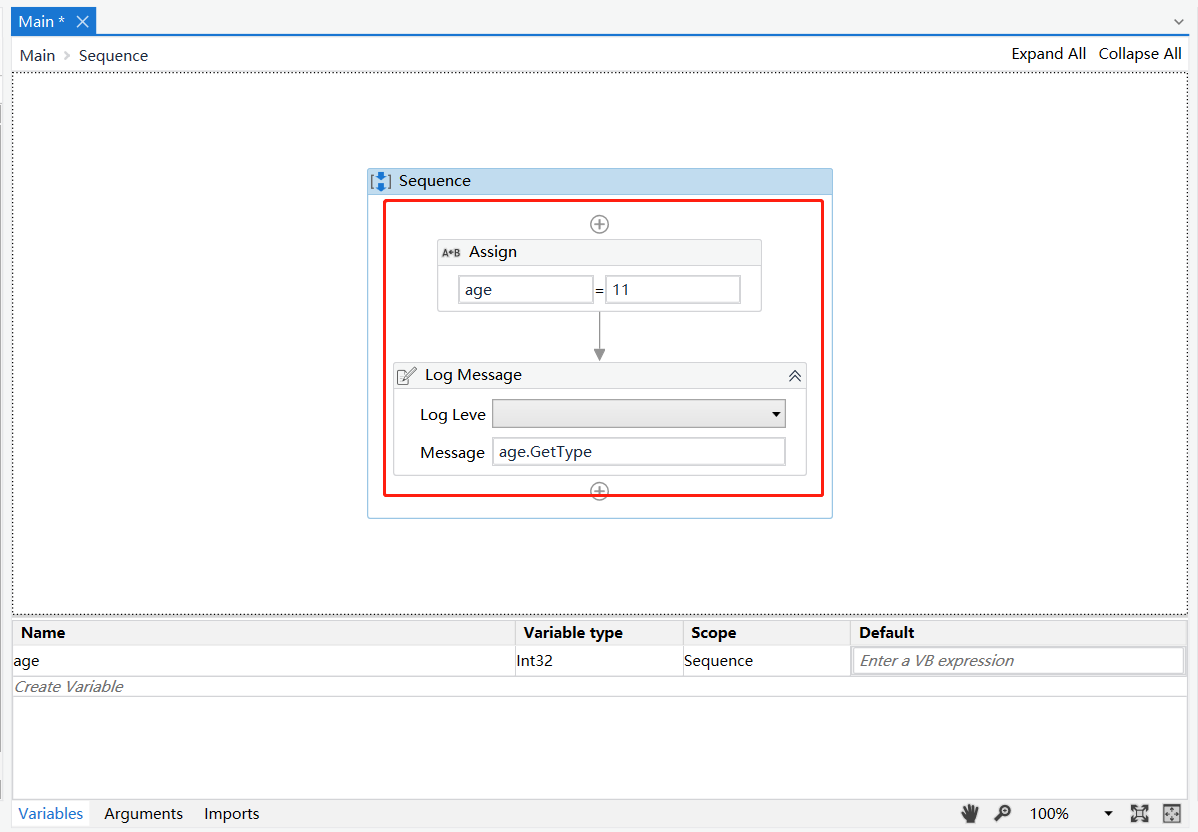
这行代码输出的就是**True**,也就是这个判断是正确的。

"Python知识学堂".StartWith("hello")

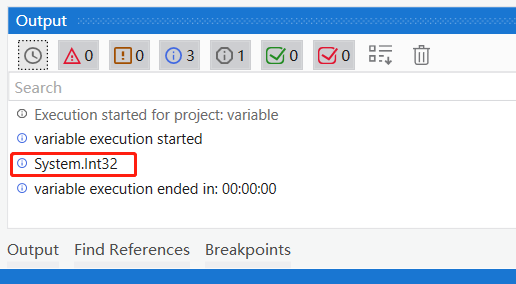
这行代码输出的就是**False**,也就是这个判断是错误的。

**2.3 整数类型**

在UiPath中，整数类型即Int32类型。在此数据类型下我们存储的是整数，比如说存储定量数量，客户年龄等。我们来看一个简单的例子，我们先赋值age一个整形数据，然后使用GetType方法打印出这个age变量的类型：



输出为：

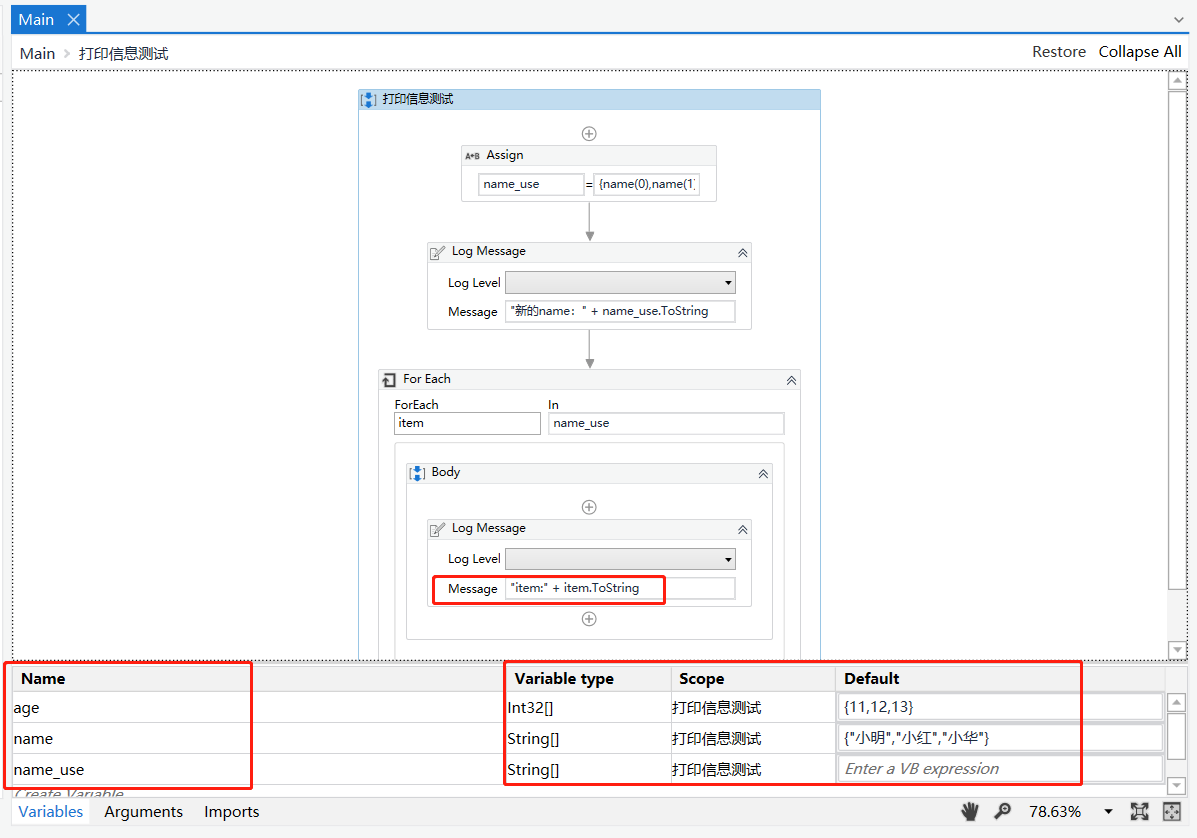


可以看出输出的结果显示我们的age是System.Int32类型的数据，这跟我们在变量面板中设置的结果是一致的。

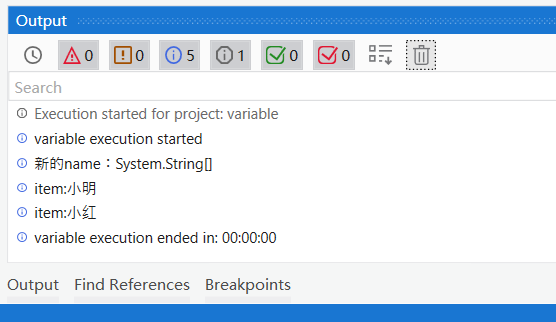
**2.4 数组类型**：

数组类型在UiPath中也是比较常见的一个数据类型，数组中可存储多个**相同类型**的多个值，数组中的各个元素之间使用“,”分隔开而已。在UiPath中数组类型也可以分为整数类型数组、字符串类型数组等等，只是不同的数组中存储的数据存在一定的区别罢了。

那么数组中的值该怎么获取呢？跟其他语言一样，我们使用下标来获取元素，下标的取值从0开始，我们来看几个小例子：我们先在Variable面板，即变量面板中初始化几个数组，然后通过赋值和For Each循环来进行取值的操作。



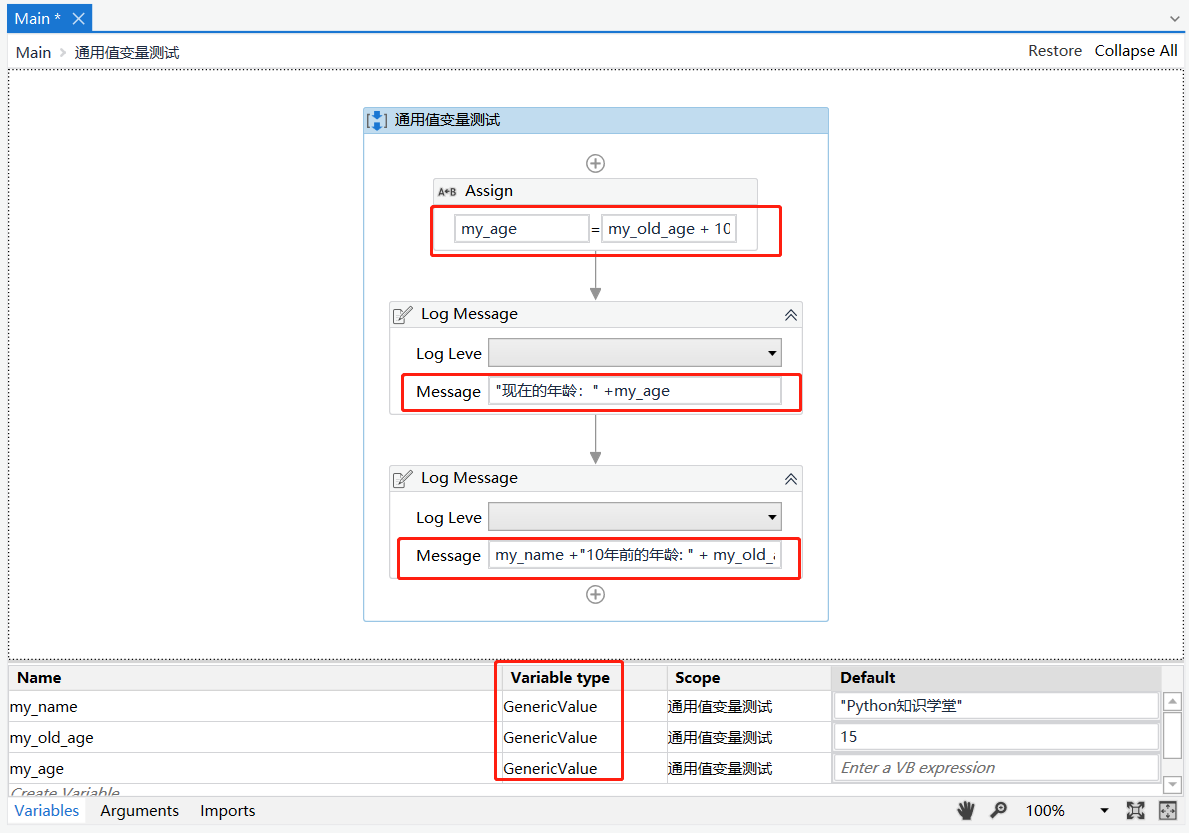
运行后输出的结果为：



其他的一些关于数组的用法，我们将在具体的例子中再补充说明。

**2.5 通用值变量**

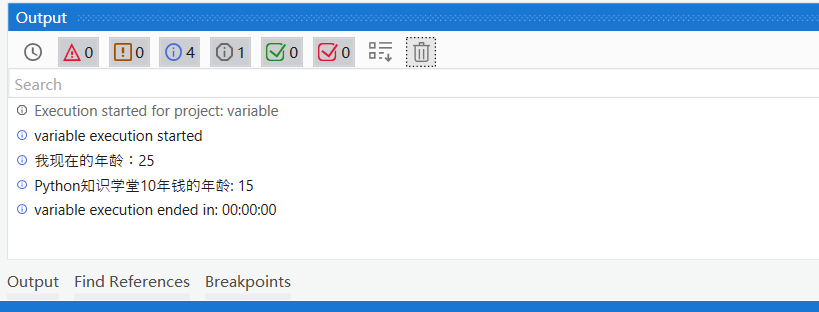
有一种UiPath特有的数据结构，可以存储任何类型的数据，包括文本，数字，日期和数组等。UiPath提供了一个GenericValue变量的自动转换的机制，可以自动的将不符合的类型转换为符合的数据类型，从而实现数据的加减乘除的操作：GenericValue变量可以自动进行变量的转换以符合其他变量的类型。我们来看一个例子：



在这个例子中我们创建了3个GenericValue的类型的变量，可以看出变量my\_name的值是字符串类型的，my\_old\_age的类型为整形的，只是我们“强制”的将其的类型认定为GenericValue类型的。

在Assign活动中，我们将my\_old\_age + 10之后的结果直接赋值给my\_age。即直接地将GenericValue和整形的变量相加。第一个Log Message活动将字符串与my\_age进行拼接后打印，第二个Log Message将字符串直接与GenericValue（int自动转换）进行拼接输出。

结果输出如下：

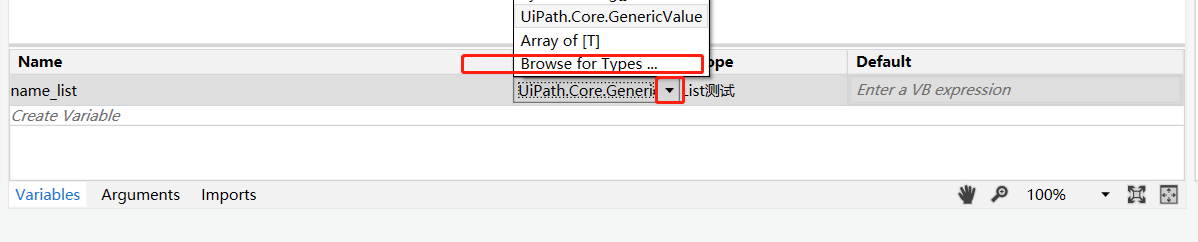


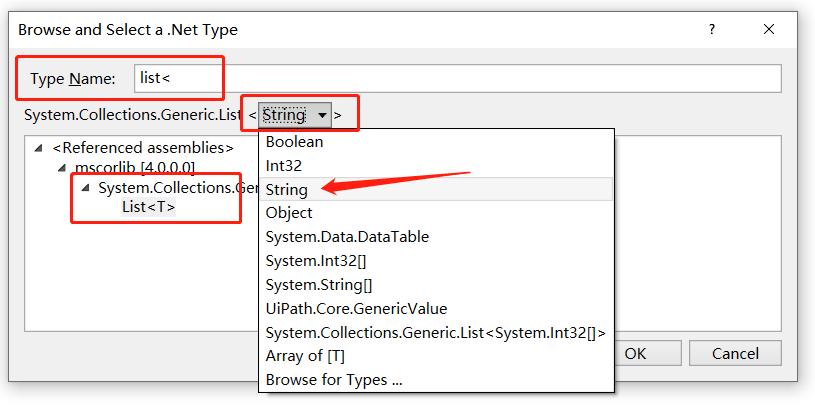
这个通用型变量的作用还是很大的，不建议大家过于依赖这个数据类型的使用，最好的就是该使用什么类型的就使用什么类型的。

**2.6 列表类型**

接下来我们来看一下列表（List）类型，列表用于在RPA流程中存储一系列数据，数据用来实现变量的交换，四则运算等操作。List支持对元素的增加和删除，这个特性使得列表在实际使用的时候非常便捷，UiPath提供了很多用于操作List的方法，比如说常见的取数、排序等等。接下来我们来看一个简单的操作：

我们在变量面板中新建一个name\_list的变量，并设置其数据结构为List。设置变量类型的时候，先点击Browse for Types...查找更多的类型，接着从“Browse and Select a .Net type”的“Type Name”中输入list<T>即可找到，我们这里要使用System.Collection.Generic下的List,并设置List中元素的类型为String,如下所示：





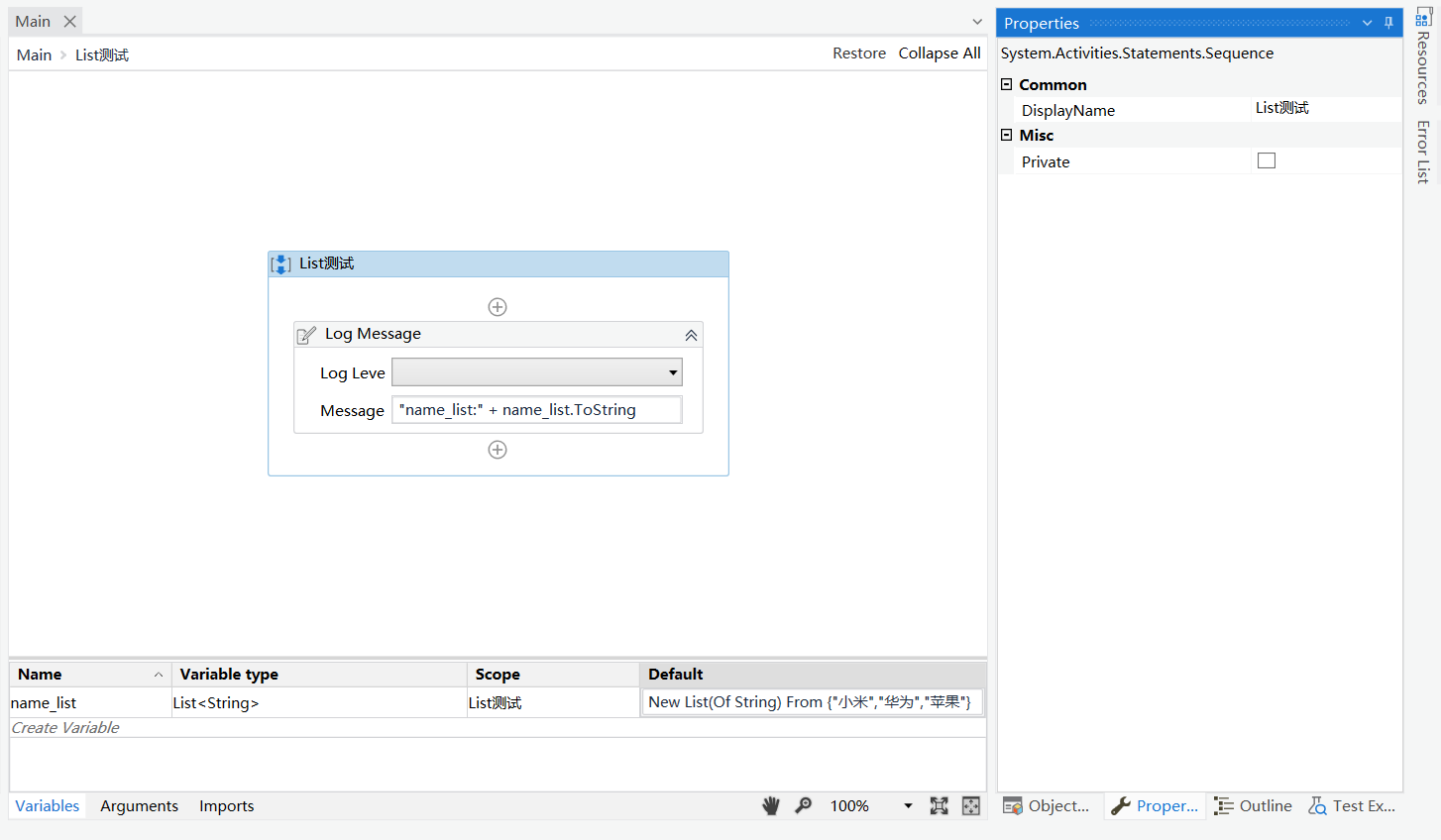
List<T>中的T表示的是List中数据的类型，我们这里使用String类型。

在UiPath中使用列表是需要**初始化**的，我们设置其默认值为 {"小米","华为","苹果"}的初始化语句为：

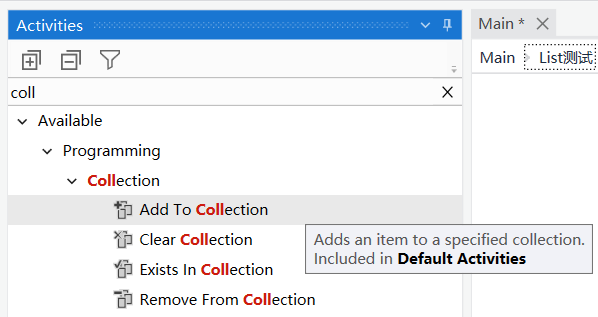
New List(Of String) From {"小米","华为","苹果"}

当然我们也可以不进行初始化的操作,此时只需省去From以及之后的语句即可。

设置之后的结果如下所示：



在UiPath中，列表属于Collection的一种,因此Collection活动的的一些方法可以用于列表：



上图显示的四个Collection方法都可以使用于列表。

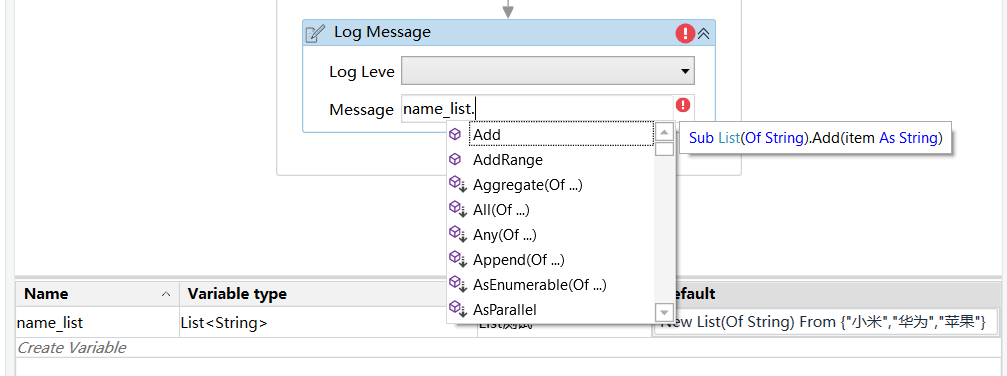
（1）Add To Collection: 将item项添加到collection中，如list中。

（2）Clear Collection: 清空collection中的item。

（3）Exists in Collection: 判断item项是否在collection中。

（4）Remove From Collection: 从collection中移除指定的item项。

当然了，还有一些使用频繁的方法，我们可以通过UiPath编辑器自动显示来查看一下：

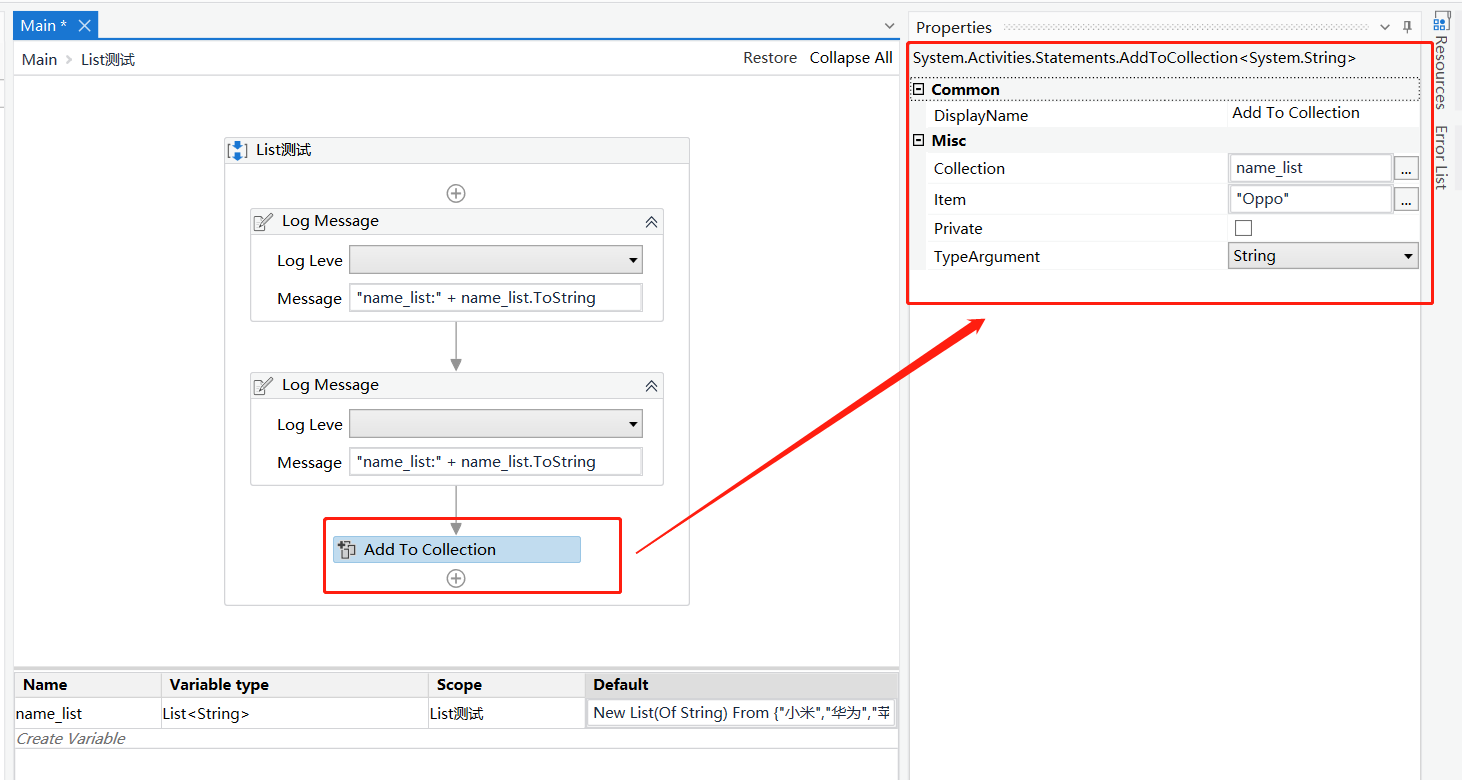


在列表之后使用“.”号就可以引出一些常用的方法，上图显示了诸如Add、AddRange、Aggregate()等方法。并且显示了方法的使用教程，比如说Add()方法用于添加元素等等。这里也稍微列举几个：

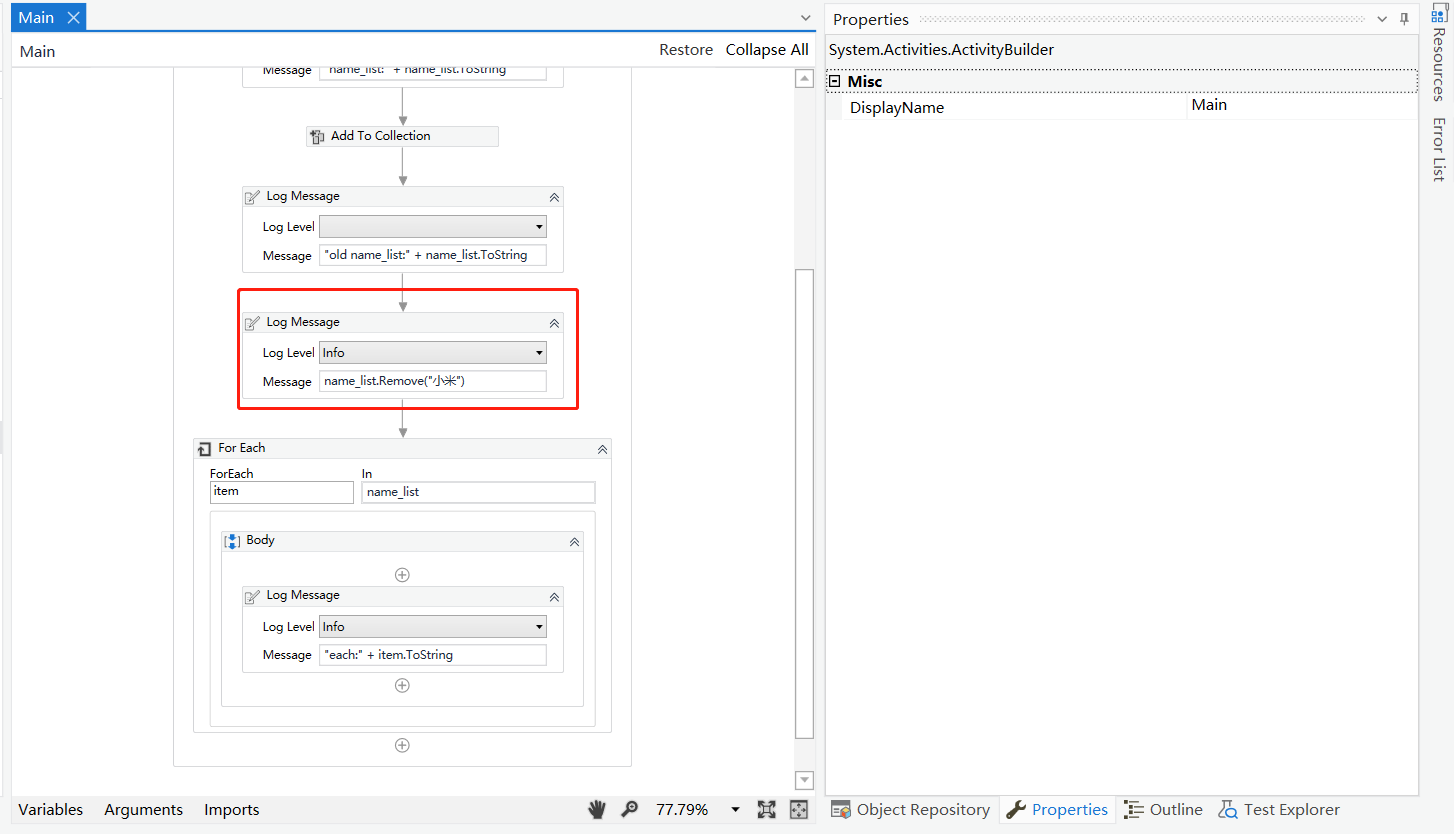
1. Add(item)：添加一个元素到列表中
2. Remove(item):移除元素
3. Contains(item):判断是否包含某个item项(元素)
4. ...

设置好之后，我们进行列表的一些操作了，这里我们不演示一些高深的操作（高深的操作通常要熟知VB或C#的语法），这里仅仅先举例简单的用法：

我们先拖动一个“Add To Collection”活动到设计面板中，并向初始化的list中添加一个新的元素“Oppo”:



接着我们把列表中的“小米”移除：



在一个列表中移除一个元素的方法很简单，比如我们将元素“小米”从列表中移除的只需要这样编写代码即可：

name\_list.Remove("小米")

**2.7 字典类型**

最后我们一下来看一下UiPath中的字典类型。

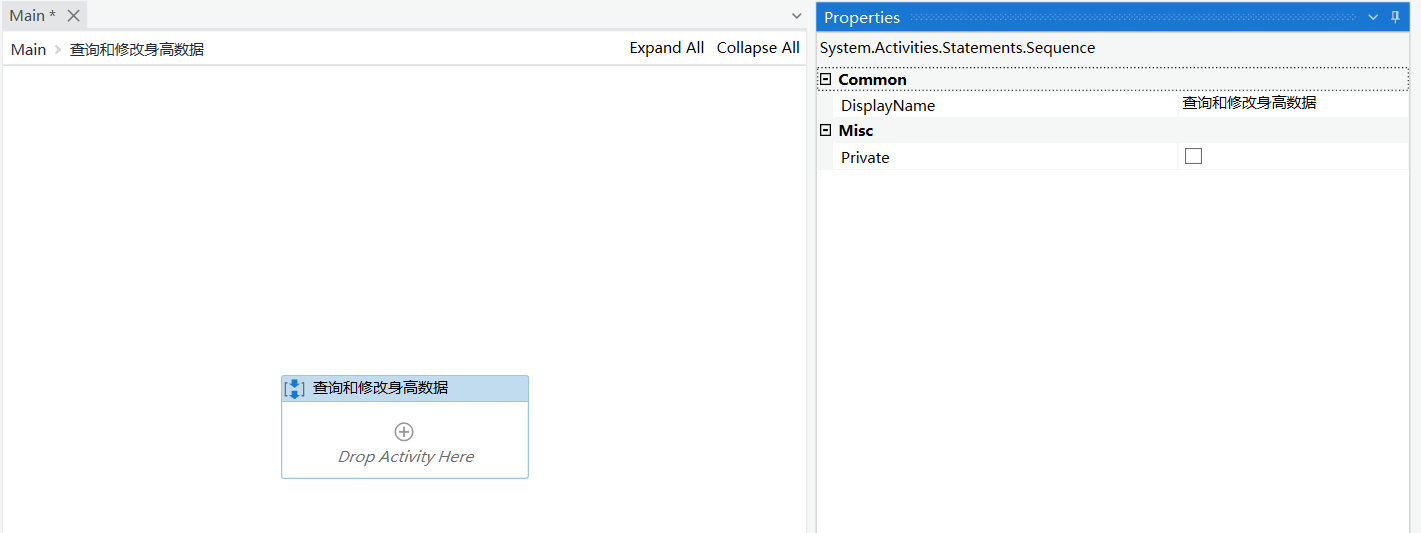
不管是在UiPath中还是在其他的语言当中，字典数据类型都是一个非常重要的数据类型。首先字典类型是一种Object类型，在运行程序中用于存储键值对，也就是说字典由键（key）和值（Value）来构建的，键是唯一的。

在编写程序中，我们经常使用到的关于字典的操作有：

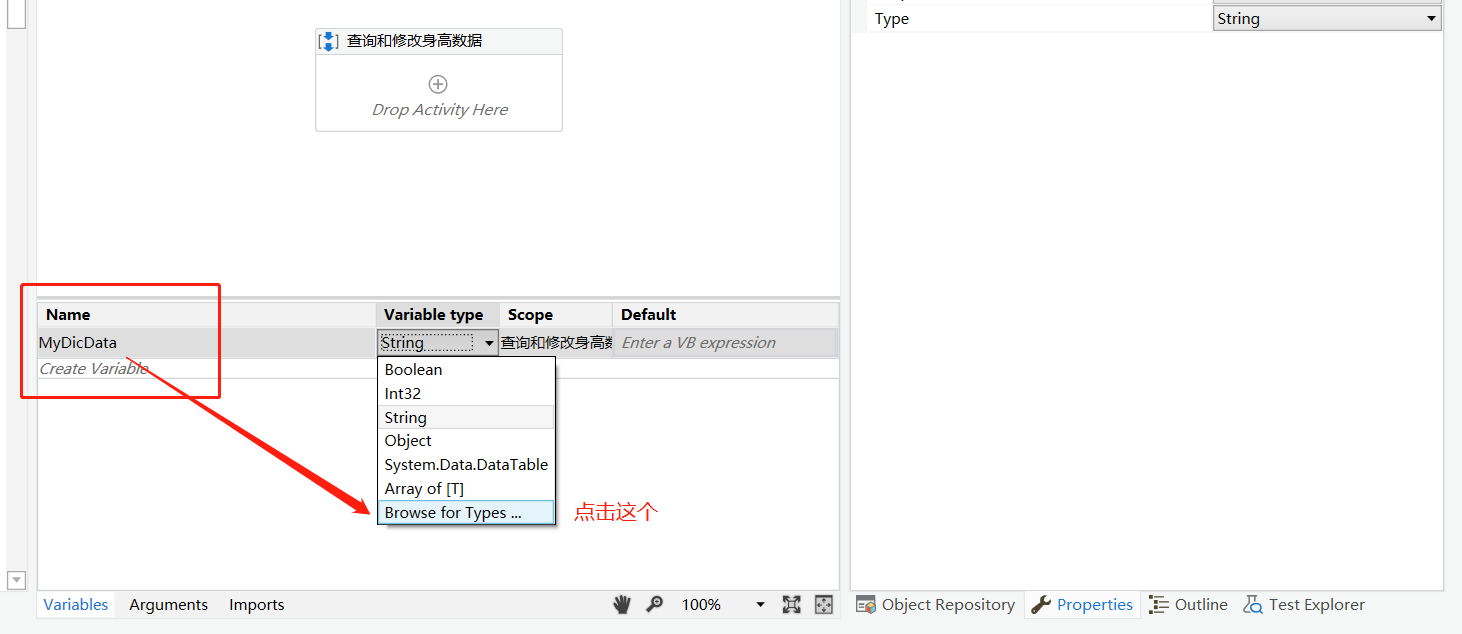
1. 增加键值
2. 修改键值
3. 删除键值
4. 查找键值
5. ......

接下来我们进行一个简单的关于字典的实操，实操之后我们再进行相应的总结，操作步骤如下：

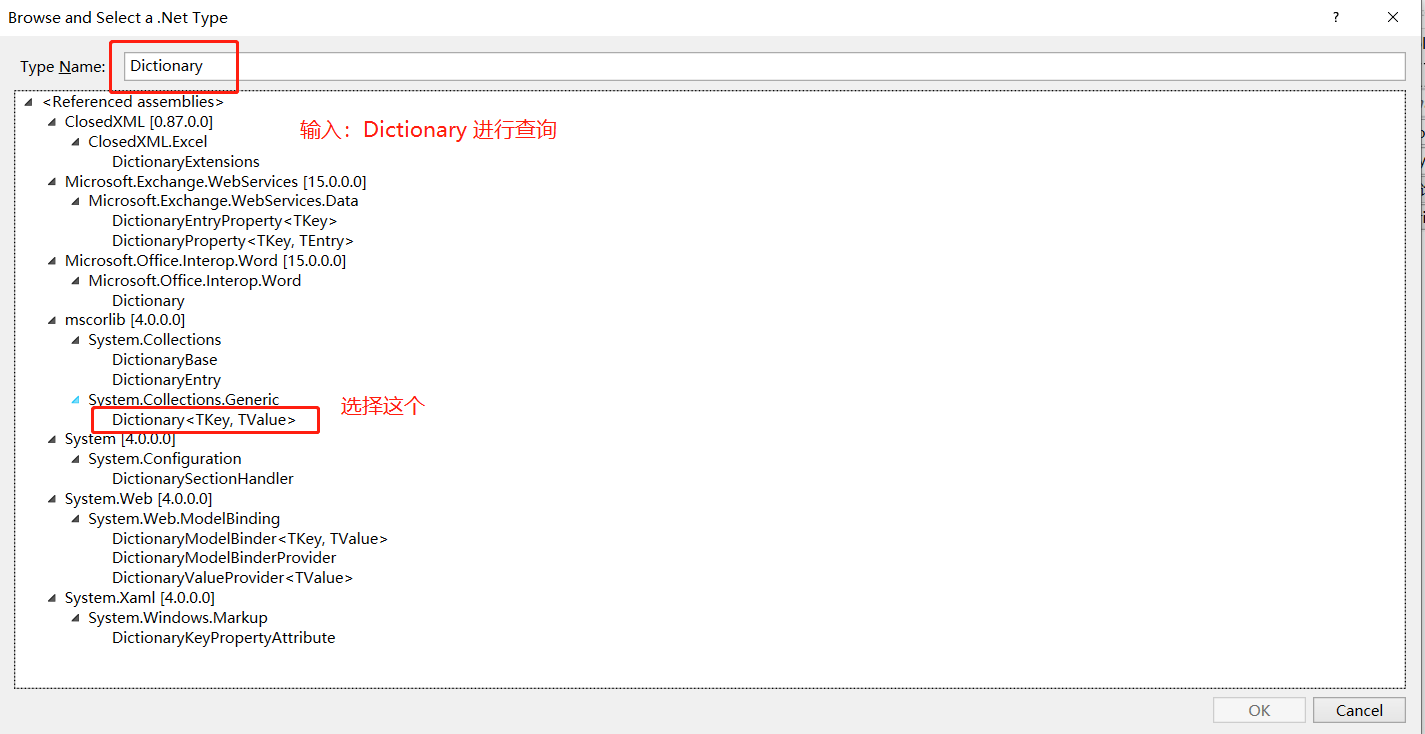
1. 点击“Process”创建一个新的流程，流程名称设置为DicTest（大家也可以 自己命名自己想要的名称），之后拖动一个Sequence活动到设计器面板，并将活动的DisplayName重新命名为：“查询和修改身高数据”。如下所示：



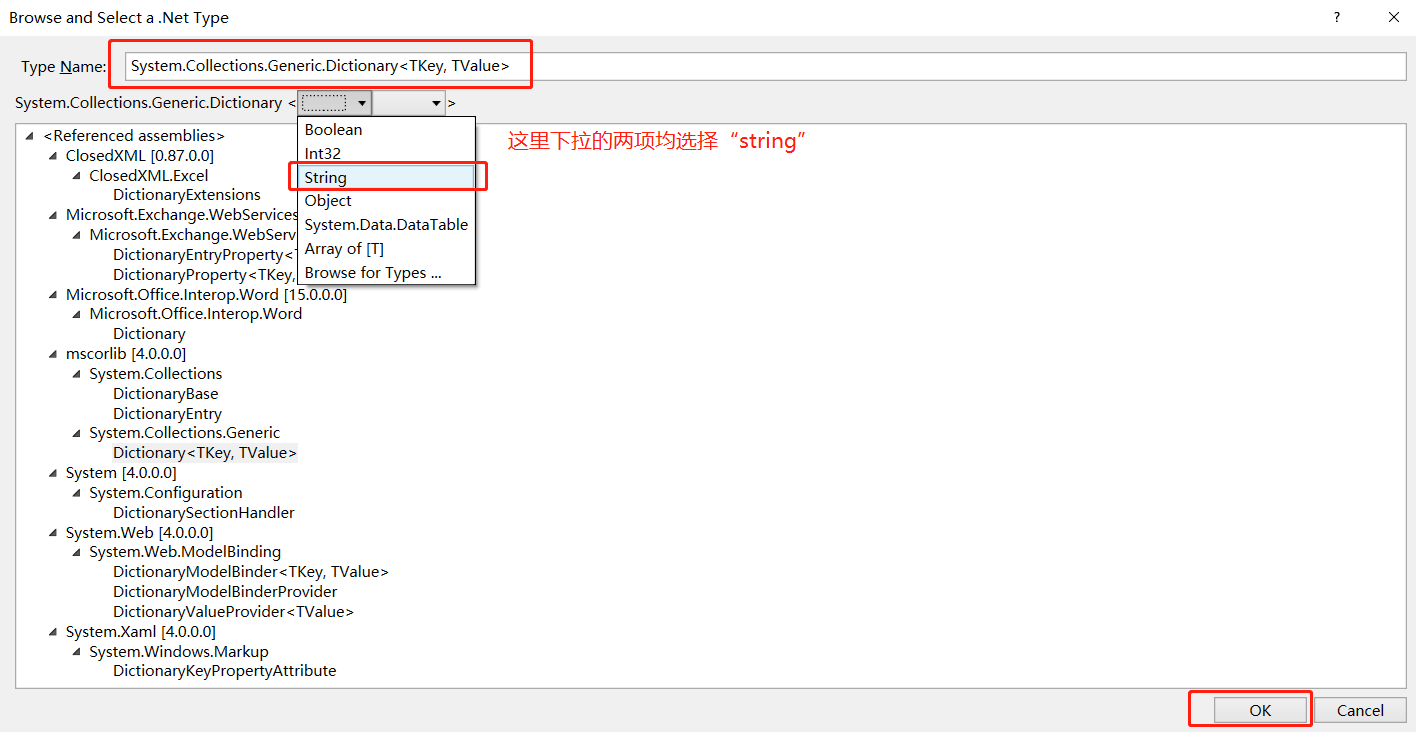
1. 接下来我们在变量面板（Variables）中创建一个字典变量MyDicData，变量的类型选择System.Collection.Generic下的Dictionary<Tkey,Tvalue>。（此步骤与列表变量的创建有相似之处，可参考上节）：



之后：



之后：

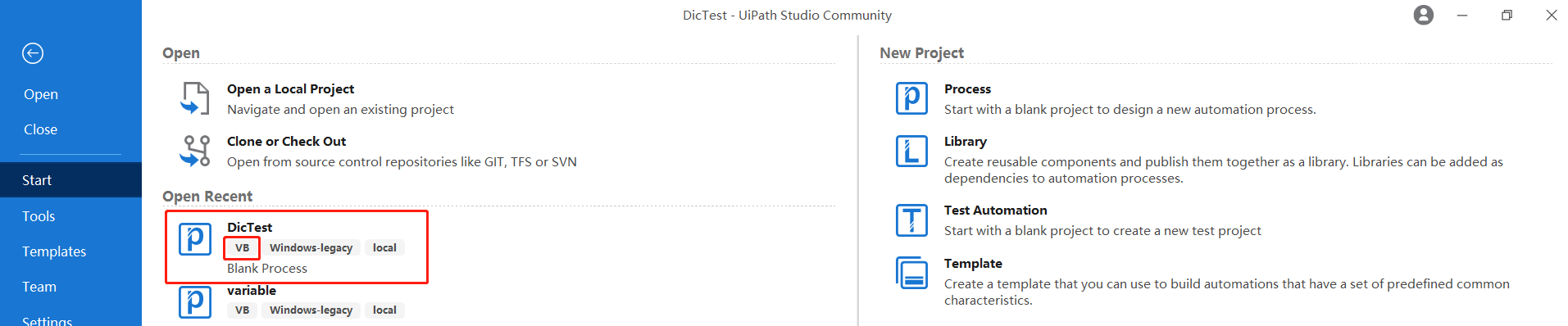


点击OK

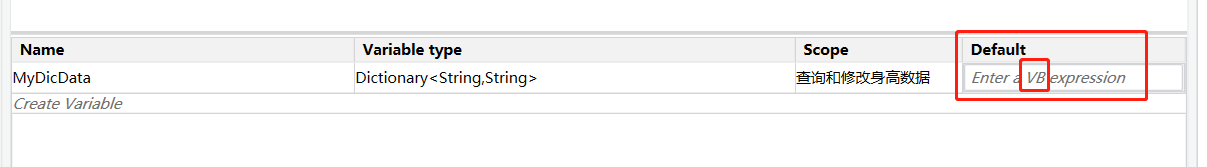
可以看出我们的字典的数据类型就选择完毕，可以看出键值都为字符串：



1. 设置好我们的字典之后，接下来的操作就是初始化我们的字典数据对象MyDicData。初始化字典数据对象的语句是根据不同语言的语法来进行的，我们可以从这里查看当前的工程使用的语言：从“HOEM”页面查看：



或者从变量的“Default”来查看：



可以看出使VB语言语法。此处我们在这里输入的VB expression初始化表达式为，初始化的时候我们使用一些数据：

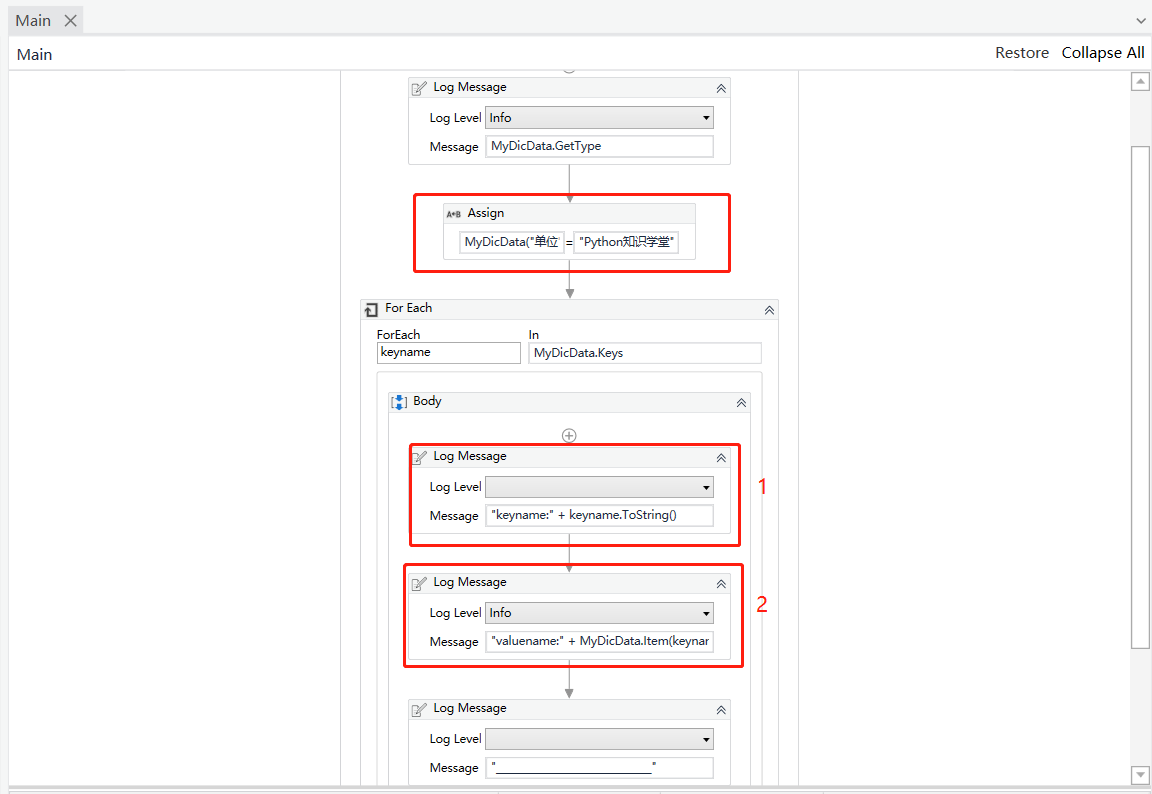
new Dictionary(of String ,String) From {{"name","小明"},{"身高","175cm"}}

如果不想用数据初始化我们可以丢弃包含From之后的数据：

new Dictionary(of String ,String)

这样我们就完成了字典数据的初始化操作了。

1. 接下来我们学习进行一些常用的操作。我们拖动一个Assign活动到界面中，并设置To为MyDicData("单位")，Value为"Python知识学堂"。之后打印出已有字典中的键值对情况。这里提供截图大家照着操作就可以了：

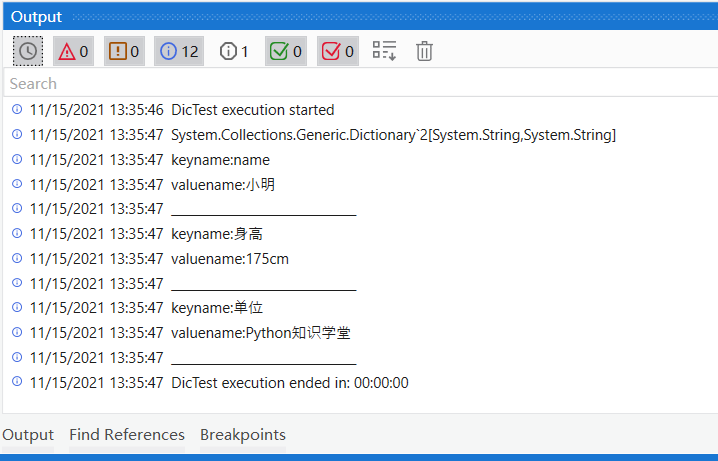


其中1,2 处的Log Message的内容分别为：

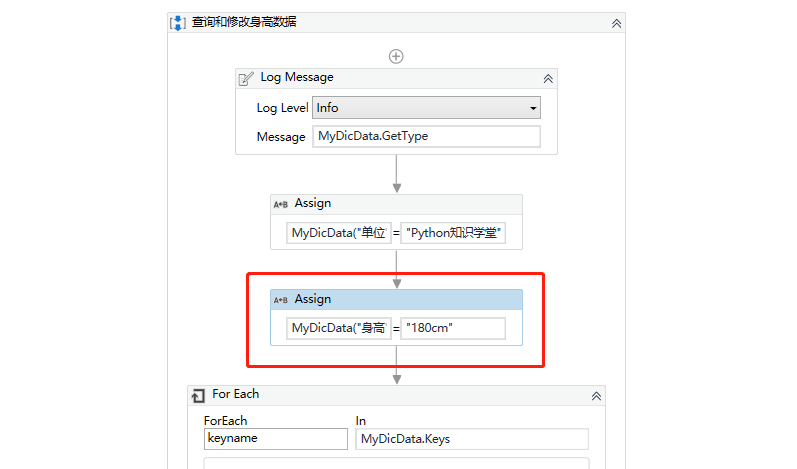
"keyname:" + keyname.ToString()

"valuename:" + MyDicData.Item(keyname.ToString())

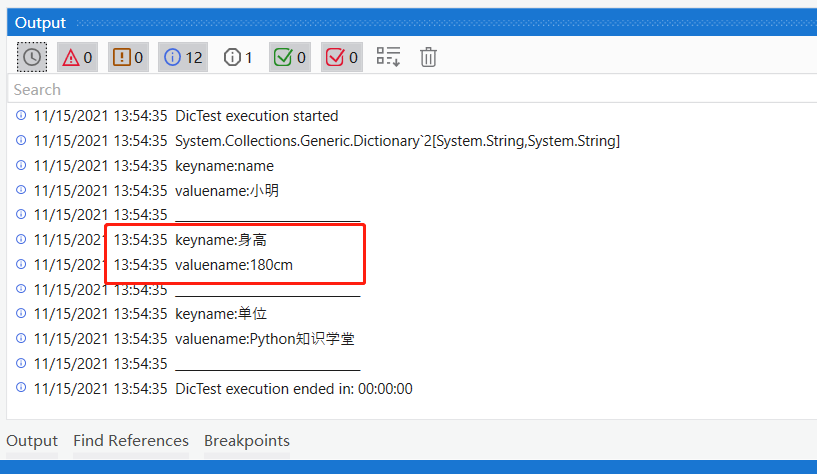
运行后输出的内容如下：



1. 上述操作是给一个字典新增一个key-value键值对。新增的原则在于：如果原字典中没有新增的键，则新建键值对；如果原字典中存在一样的键则新的值将覆盖原值。接下来我们再拖动一个Assign活动到之前的Assign下，并设置To为MyDicData("身高")，Value为"180cm":

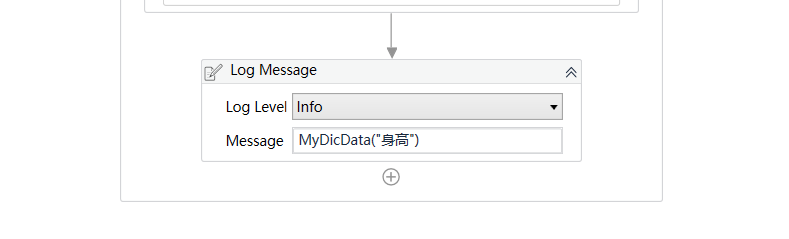


接着我们运行，查看输出为：



可以发现身高已经改变为180cm了。

1. 那么当我们有一个字典的时候，我们该怎么进行定向的查询操作呢？比如上述的MyDicData字典。我们想知道身高是多少，理论上我们只需要进行输入key的数值，就可以找到Value的数值。没错，实现这个功能，只需要这样一句话：Dictionary(Key)即可，具体的在这里使用MyDicData("身高")即可。

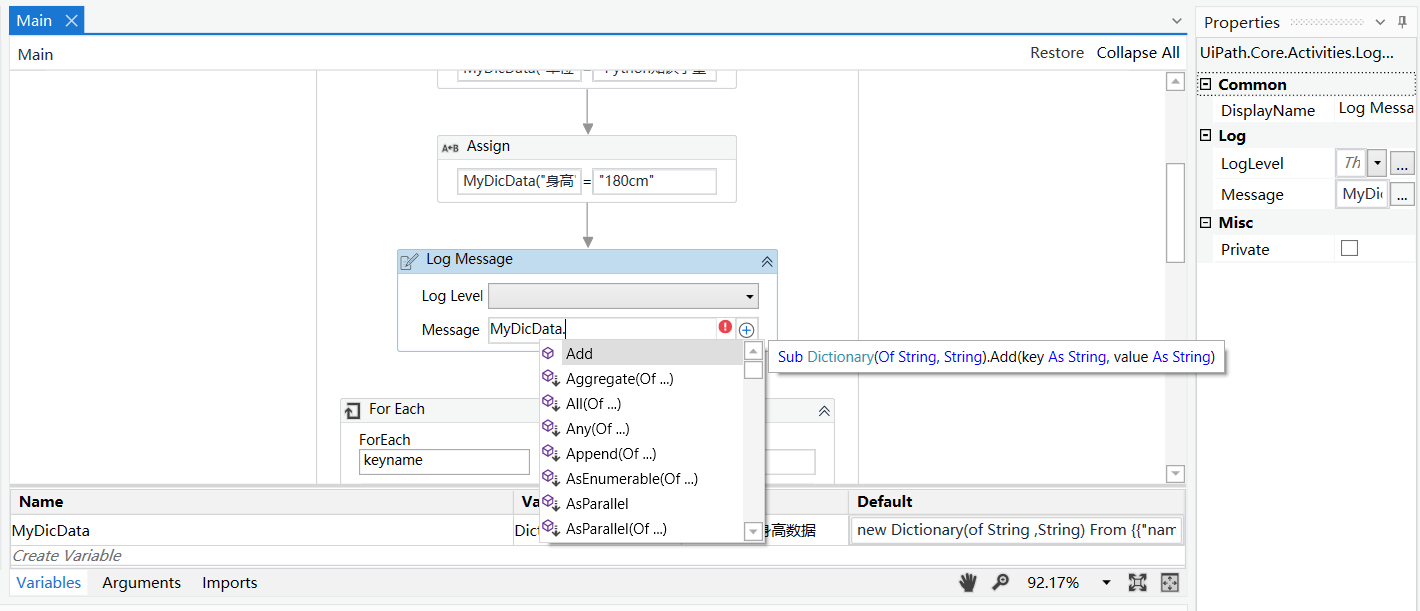


至此，我们就完成了上述的功能。

最后我们总结一下字典的常用方法，假设现在我们有两个字典数据：MyDic1和MyDic\_2：

1. MyDic\_1(key)：获取字典MyDic\_1中key对应的value值。
2. MyDic\_1.item(key)：获取字典MyDic\_1中key对应的value值,同（1）。
3. MyDic\_1.Keys：获取MyDic\_1中所有的key组成的集合，可使用for遍历。
4. MyDic\_1.value：获取MyDic\_1中所有的value组成的集合，可使用for遍历。
5. MyDic\_1.Keys(inx)：获取指定inx下标的key(inx从0开始)。
6. MyDic\_1.Values(inx)：获取指定inx下标的key(inx从0开始)。
7. MyDic\_1.Count：字典中键值对数量。
8. MyDic\_1.ContainsKey(key)：判断字典中是否包含指定key。
9. MyDic\_1.ContainsValue(value)：判断字典中是否包含指定value。

其他的方法可以使用.号运算符进行查看，如下所示：



当然了更多的方法大家可以参考官网文档进行学习。

1. **总结**

本次的文章内容主要是关于UiPath中的数据类型的，内容还是不少的大家慢慢消化。本文中案例不算少，希望学习的小伙伴可以按照教程中进行学习，有什么疑问的也欢迎沟通探讨，我们下次再见。