

LECTURE 03. COLLECTIONS. PART I

Robotic Process Automation

[17 October 2023]

Elective Course, 2023-2024, Fall Semester

Camelia Chisăliță-Crețu, Lecturer PhD

Babeș-Bolyai University

Acknowledgements

This course is presented to our Faculty with the support of UiPath Romania.



Contents

- **PART I**

- **Arguments**

- Definition. Types. direction
- Invoke Workflow File Activity
- Demo 1

- **Variable Categories**

- **Array**

- Details
- Declaration. Instantiation. Initialization
- Demo 2
- Demo 3

- **List**

- Details
- Declaration. Instantiation. Initialization
- Operations
- Demo 4
- Demo 5

- **PART II**

- **Dictionary**

- Details
- Declaration. Instantiation. Initialization
- Operations
- Demo 6
- Demo 7

- **Data Table**

- Details
- Declaration. Instantiation. Initialization
- Operations
- Demo 8
- Demo 9

- **References**

Arguments. Definition

- **Arguments** are
 - used to pass data from a project to another;
 - similar to parameters in method definition;
- **advantages:**
 - increase workflow readability;
 - increase sequence/flowchart reusability in workflows;
- **arguments vs variables:**
 - **argument** – it stores data dynamically and passes it on, between *automations*, i.e., projects;
 - **variable** – it passes data between *activities*.

see **Demo1 - Arguments**

Arguments. Types. Direction

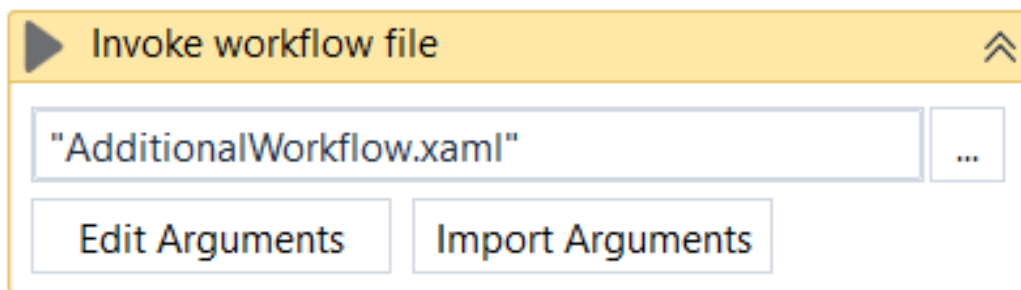
- **argument types** – similar to *variable types* available in UiPath;
- **argument direction** – it indicates *where* the information stored in them is supposed to go;
- possible argument directions:
 - **In** – data can be used in the current project only; it was sent by another project;
 - **Out** – data can be used outside the current project;
 - **In/Out** – **In** + **Out**.

see **Demo1 - Arguments**



Arguments. Invoke Workflow File Activity

- **Invoke Workflow File** activity
 - allows to invoke a specified workflow, passing a list of input/output arguments (parameters);
- **Properties to set:**
 - **WorkflowFileName** - a string with the file path of the **.xaml** file to be invoked; the file path is relative to the current project folder;
 - **Arguments** - the parameters that are passed to the invoked workflow;
 - **Isolated** - if checked, the invoked workflow runs in a separate Windows process; this is useful when isolating a faulty workflow from the main workflow (invoker).



see **Demo1 - Arguments**

Demo 1

- Create a process that performs the following actions:
 - 1. *read* two integer numbers;
 - 2. *compute* the maximum value between the given numbers;
 - *define a distinct workflow having:*
 - *two input arguments;*
 - *one output argument;*
 - *invoke the defined workflow;*
 - 3. *print* the computed maximum value.

see Demo1 - Arguments

Variable Categories

- **Scalar** - single value of a fixed type:
 - **Int32, Boolean, Character, Date Time**, etc.;
- **Collections** – *single dimension* structures, where multiple values are viewed as an entity:
 - **Array, List, Queue;**
 - **String;**
 - **Dictionary;**
- **Tables** – *two dimensional* structures, where multiple values are viewed as an entity:
 - **Data Table.**

Arrays. Details

- **Array** characteristics in UiPath:
 - **an array has a fixed length**, set at instantiation/initialization time;
 - it implements the **IEnumerable** interface ==> can be iterated by using a **For Each** activity.

see **Demo2 - Arrays**

Arrays. Declaration. Instantiation. Initialization

- ways to **declare/instantiate/initialize** an array:
 - **Variables Panel:**
 - **Name:** colourArray; **Type:** String[]; **Default:** {"red", "green", "blue"} //Length=3
 - **Assign** activity:
 - colourArray= new String[]{"red", "green", "blue"} //Length=3
 - colourArray= new String(2>{"red", "green", "blue"} //Length=3
 - colourArray= new string(2){} //values are set later, Length=3, valid indices: 0, 1, 2
- ways to **set/change the values** in arrays:
 - **Assign** activity:
 - colourArray(0)= "red" // overrides or initializes the value on index = 0

Arrays. Example 1

The screenshot displays the UiPath Studio interface with a workflow designed to create and display an array. The workflow consists of three activities:

- Assign:** A variable named `colourArray` is assigned the value `new String(2){"red"}`.
- Write Line:** The first instance displays the text `colourArray(0)+" "+colourArr`.
- Write Line:** The second instance displays the text `colourArray.Length.ToString`.

The **Properties** window on the right shows the configuration for the `colourArray` variable:

Common	
DisplayName	Sequence Array c
Misc	
Private	<input type="checkbox"/>

The **Output** window shows the execution results:

- Arrays execution started
- red green blue
- 3
- Arrays execution ended in: 00:00:00

At the bottom, the **Variables** window shows the details for `colourArray`:

Name	Variable type	Scope	Default
colourArray	String[]	Sequence	Enter a VB expression

see Demo2 - Arrays

Arrays. Example 2

The screenshot displays the UiPath Studio interface with a workflow designed to initialize and output an array. The workflow consists of four steps:

- Assign:** `colourArray = new string(2){}`
- Assign:** `colourArray(0) = "red"`
- Assign:** `colourArray(1) = "green"`
- Write Line:** `colourArray(0) + " " + colourArray(1)`

The **Properties** pane on the right shows the **Common** tab with **DisplayName** set to **Sequence** and **Private** checked. The **Output** pane shows the execution results:

```
Arrays execution started
red green
3
Arrays execution ended in: 00:00:00
```

The **Variables** pane at the bottom shows the variable **colourArray** of type **String[]** with a **Scope** of **Sequence** and a **Default** value of `Enter a VB expression`.

Name	Variable type	Scope	Default
colourArray	String[]	Sequence	Enter a VB expression

see Demo2 - Arrays

Arrays. Example 3

The image shows the UiPath Studio interface with a workflow designed to manipulate an array. The workflow consists of the following steps:

- Assign:** `intArray = new Int32(2){1,2,3}`
- Write Line:** `String.Join(" ", intArray)`
- Assign:** `intArray = intArray.AsEnumerable().Concat({4}).ToArray()`
- Write Line:** `String.Join(" ", intArray)`

The **Properties** window on the right shows the following details for the `intArray` variable:

- Common:** DisplayName: Sequence-A
- Misc:** Private: ☐

The **Output** window shows the execution results:

- Arrays execution started
- 1 2 3
- 1 2 3 4
- Arrays execution ended in: 00:00:00

The **Variables** window at the bottom displays the following table:

Name	Variable type	Scope	Default
intArray	Int32[]	Sequence-Array St..	Enter a VB expression

At the bottom of the interface, there are tabs for **Variables**, **Arguments**, and **Imports**, along with a zoom level of 100%.

see Demo2 - Arrays

Demo 3

- Create a process that performs the following actions:
 - 1. *instantiate* an array of 5 integer numbers;
 - 2. *set* the values of the generated numbers from 1 to 10 and:
 - 2.1. *print* the generated number;
 - 2.2. *check* if the number is even:
 - 2.2.1. if TRUE then *print* number+1 ;
 - 2.2.2. if FALSE then *print* number-1 ;
 - *use For Each activity to iterate the array;*
 - *use Log activity to print the generated values;*
 - 3. *compute* the sum of numbers in the array;
 - 4. *print* the sum;
 - 5. *print* the array;
 - *use a String-based method.*
 - Discuss the presence and usage of the local variable *item*

see Demo3 – RandomNumbersArray

Lists. Details

- **List** characteristics in UiPath:
 - **a list has a flexible length**;
 - it implements the **IEnumerable** interface ==> can be iterated by using a **For Each** activity.

see **Demo4 - Lists**

Lists. Declaration. Instantiation. Initialization

- ways to **declare/instantiate/initialize** a list:
 - **Variables Panel:**
 - **Name:** weekDaysList; **Type:** List<String>;
 - **Default:** `new List(of String) from {"Monday", "Tuesday"} //Length=2`
 - **Assign** activity:
 - `weekDaysList = new List(of String)from{"Monday", "Tuesday"}//Length=2`
 - `weekDaysList = new List(of String) //Length=0, values are added later`
- way to **change values already added to the list:**
 - **Assign** activity:
 - `weekDaysList(0)= "Friday" // overrides the value on index = 0`

Lists. Example 1

The screenshot shows the UiPath Studio interface with a workflow named 'Main'. The workflow consists of three activities: two 'Assign' activities and one 'Write Line' activity. The first 'Assign' activity assigns a new list of strings to 'weekDaysList', initialized with 'Monday' and 'Tuesday'. The second 'Assign' activity assigns a new list of strings to 'aList'. The 'Write Line' activity uses 'String.Join' to concatenate the elements of 'weekDaysList' with a space separator and outputs the result to the console.

Variables Table:

Name	Variable type	Scope	Default
weekDaysList	List<String>	Sequence	new List(of String) from {"Monday", "Tuesday"}
aList	List<String>	Sequence	Enter a VB expression

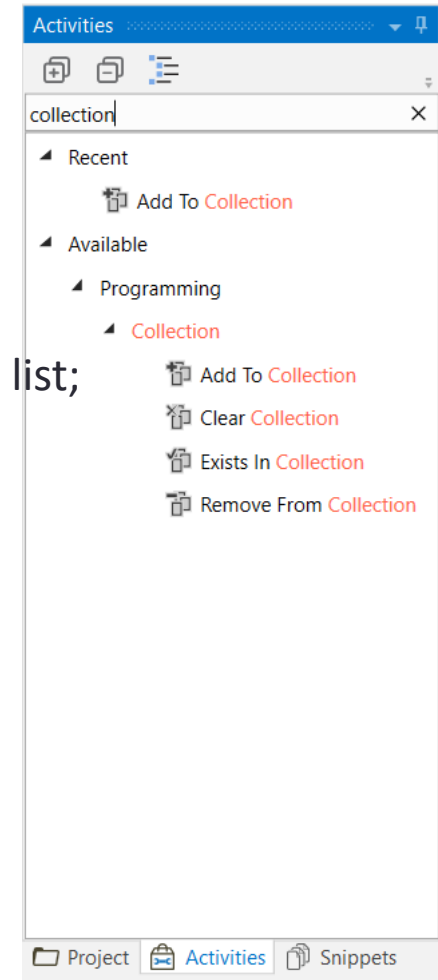
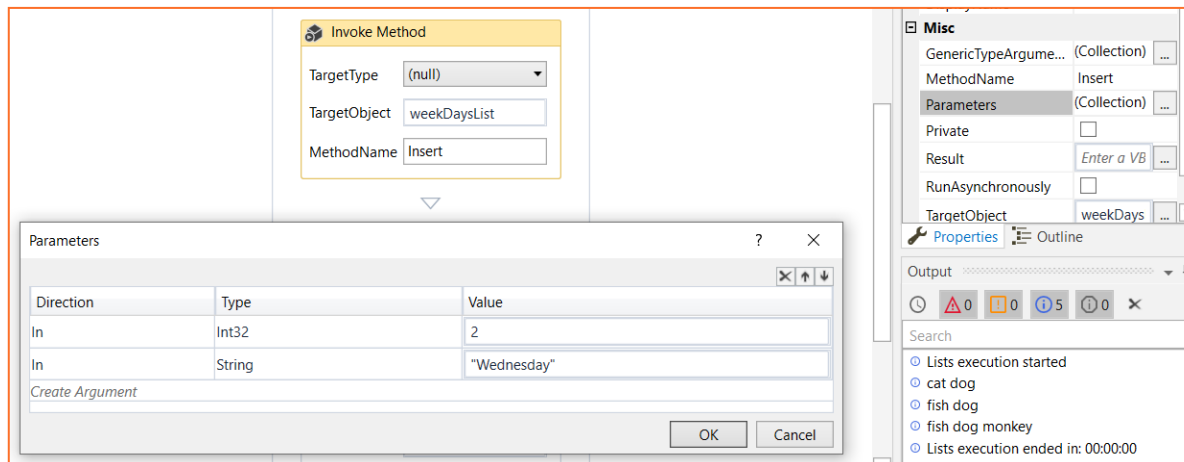
Output Log:

- Lists execution started
- Monday Tuesday
- Lists execution ended in: 00:00:00

see Demo4 - Lists

Lists. Operations

- ways to apply predefined operations on lists:
 - Invoke Method** activity:
 - the user needs to know the signature of the use method;
 - it can be used to *Add, Insert, Remove, etc.* an item to/from a list;
 - Collection** package of activities:
 - Add To Collection;**
 - Clear Collection;**
 - Exists In Collection;**
 - Remove From Collection;**



see Demo4 - Lists

Lists. Example 2A. Set/update an item

Main Expand All Collapse All System.Activities.Statements.Sequence

Common

DisplayName Sequence-Li

Misc

Private ☐

Properties Outline

Output

⌚ 0 0 5 0 ✕

Search

- Lists execution started
- cat dog
- fish dog
- fish dog monkey
- Lists execution ended in: 00:00:00

Main

A*B Assign

aList = new List(of String)

new List(of String) from{"cat", "dog"}

Write Line

Text String.Join(" ",aList)

A*B Assign

aList(0) = "fish"

Name	Variable type	Scope	Default
aList	List<String>	Sequence	Enter a VB expression
item	String	Sequence-List Exa...	"monkey"

see Demo4 - Lists

Lists. Example 2B. Add an item

- **Invoke Method** activity can be used to *add, insert, remove, etc.* an item to/into/from a list;

The screenshot displays the UiPath Studio interface. In the center, the 'Invoke Method' activity is configured with 'TargetType' set to '(null)', 'TargetObject' set to 'aList', and 'MethodName' set to 'Add'. Below it, a 'Write Line' activity is visible with the text 'String.Join(" ", aList)'. A 'Parameters' dialog box is open in the foreground, showing a table with the following data:

Direction	Type	Value
In	String	item

Below the table is a 'Create Argument' section. At the bottom of the dialog are 'OK' and 'Cancel' buttons.

On the right side, the 'Properties' pane shows the 'Invoke Method' activity's properties. The 'Misc' section is expanded, showing 'GenericTypeArgument' as '(Collection)', 'MethodName' as 'Add', 'Parameters' as '(Collection)', 'Private' as unchecked, 'Result' as 'Enter a VB', 'RunAsynchronously' as unchecked, and 'TargetObject' as 'aList'.

Below the properties pane is the 'Output' pane, which shows a search bar and a list of execution logs:

- Lists execution started
- cat dog
- fish dog
- fish dog monkey
- Lists execution ended in: 00:00:00

At the bottom of the screen, a table lists the variables used in the workflow:

Name	Variable type	Scope	Default
aList	List<String>	Sequence	Enter a VB expression
item	String	Sequence-List Exa...	"monkey"

see Demo4 - Lists

Lists. Example 3. Collection Package

- **Add To Collection** activity is used add an item to the list;

The screenshot displays a UiPath workflow editor. The workflow consists of two activities: 'Add To Collection' (highlighted in yellow) and 'Write Line'. The 'Write Line' activity has a text box containing the expression `String.Join(" ", weekDaysList)`. Below the workflow, a table defines the variables used:

Name	Variable type	Scope	Default
value	String	Sequence-List Exa...	"Thursday"
weekDaysList	List<String>	Sequence	new List(of String) from {"Monday", "Tuesday"}
aList	List<String>	Sequence	Enter a VB expression

Below the table is a link labeled 'Create Variable'. To the right of the workflow editor, the 'Properties' window is open, showing the 'Common' tab for the 'Add To Collection' activity. The 'Collection' property is set to 'weekDays' and the 'Item' property is set to 'value'. The 'Output' window shows the execution log:

```
Lists execution started
Monday Tuesday Thursday
Monday Tuesday Wednesday Thursday
Tuesday Wednesday Thursday
Lists execution ended in: 00:00:00
```

see Demo4 - Lists

Demo 5

- Create a process that performs the following actions:
 - 1. *create* a list of the following items: {"still water", "sparkling water", "herbal tea", "coffee", "wine", "juice", "green tea", "black tea"};
 - 2. *generate* an index value (from 0 to 7);
 - 3. *print* the message "*Do you like to drink some juice/green tea/...?*" and enter the answer (*yes* or *no*):
 - 3.1. if "YES" move the item to the front of the list;
 - 3.2. if "NO" move the item to the end of the list;
 - 4. *repeat* step 3. *four* times;
 - 5. *print* the bill, i.e., the list of accepted drinks.

References

- UiPath Docs - <https://docs.uipath.com/>
- UiPath Studio Docs - <https://docs.uipath.com/studio/standalone/2023.4>
- UiPath Forum - <https://forum.uipath.com/>
- UiPath Academy - <https://academy.uipath.com/>