

LECTURE 04A. COLLECTIONS. PART II

Robotic Process Automation

[22 October 2019]

Elective Course, 2019-2020, 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
- **Generic Value Type**
 - Methods
- **Variable Categories**
- **Array**
 - Details
 - Declaration. Instantiation. Initialization
 - Example 1, 2, 3
 - Demo 2
- **List**
 - Details
 - Declaration. Instantiation. Initialization
 - Example 1
- **Operations**
- **Example 2, 3**
- **Demo 3**
- **PART II**
- **Dictionary**
 - Details
 - Declaration. Instantiation. Initialization
 - Operations
 - Example 1, 2, 3
 - Demo 4
- **Data Table**
 - Details
 - Declaration. Instantiation. Initialization
 - Operations
 - Example 1, 2, 3
 - Demo 5
- **References**

Dictionary. Details

- **Dictionary** characteristics in UiPath:
 - **a dictionary has a flexible length;**
 - it implements the **IEnumerable** interface ==> can be iterated by using a **For Each** activity;
 - it is used to store:
 - multiple related pairs (key, value) that are passed as a **single argument** between workflows;
 - data in **Orchestrator queues**;

Dictionary. Declaration. Instantiation. Initialization

- ways to declare/instantiate/initialize a dictionary:
 - **Variables Panel:**
 - **Name:** bookDictionary; **Type:** Dictionary<String, String>;
 - **Default:** `new Dictionary (of String, String) from {"title", "Poems"}, {"author", "M.Eminescu"}, {"publisher", "Litera"}} //Count=3`
 - **Name:** gradeDictionary; **Type:** Dictionary<String, List<String>>;
 - **Default:** `new Dictionary(of Int32, List(of String)) from {{10, new List (of String) from {"Ana", "Anca"}}, {3, new List(of String) from {"me", "you", "her"}}} //Count=2`
 - **Assign** activity:
 - `monthDictionary = new Dictionary (of Int32, List(of String)) //Count=0`
 - `sDictionary = new Dictionary(of String, String) //Count=0, pairs are added later`

Dictionary. Operations

- way to add pairs in a dictionary:
 - **Assign** activity:
 - `bookDictionary("year") = "2019"` // overrides the value on key “year” or adds a new pair {"year", "2019"}
 - `monthDictionary(30) = new List(of String) from {"April", "June", "September"}`
 - **Add To Collection** activity:
 - properties set:
 - `Collection = monthDictionary(31);`
 - `Item = "March";`

Dictionaries. Example 1

The screenshot displays a UiPath Studio workflow for dictionary operations. The workflow consists of the following steps:

- bookDictionary (year = 2019)**: A variable declaration for a dictionary.
- A+B Assign**: An assignment block where `bookDictionary("year" = "2018")` is assigned to the `bookDictionary` variable.
- Write Line**: A task to write a line of text, with the text field containing `String.Join(" ", bookDictionary.Keys(1))`.
- Write Line**: Another task to write a line of text, with the text field containing `String.Join(" ", "key:" + bookDictionary.Keys(1) + " value: " + bookDictionary.Values(1))`.

A tooltip for the second **Write Line** task shows the full expression: `String.Join(" ", "key:" + bookDictionary.Keys(1) + " value: " + bookDictionary.Values(1))`.

At the bottom, the **Variables** pane shows the `bookDictionary` variable of type `Dictionary<String, String>` with its default value: `new Dictionary (of String, String) from`.

On the right, the **Output** console shows the following log messages:

- ⓘ Dictionaries execution started
- ⓘ [title, Poems] [author, M.Eminescu]
- ⓘ [publisher, Litera] [year, 2018]
- ⓘ key:author value: M.Eminescu
- ⓘ Dictionaries execution ended in: 00:00:00

Dictionaries. Example 2

The screenshot displays the UiPath Studio interface with a workflow designed to process a dictionary of student grades. The workflow consists of a 'For Each' loop that iterates over the keys of a variable named 'gradeDictionary'. Inside the loop's body, there is a 'Write Line' activity that outputs a string formatted as 'key: ' + key.ToString + ' Student: ' + String.Join(' ', gradeDictionary[key]) for each key.

Workflow Details:

- For Each:** Iterates over `key` in `gradeDictionary.Keys`.
- Body:**
 - Write Line:** Text is `"key: " + key.ToString + " Student: " + String.Join(" ", gradeDictionary[key])`.

Output Log:

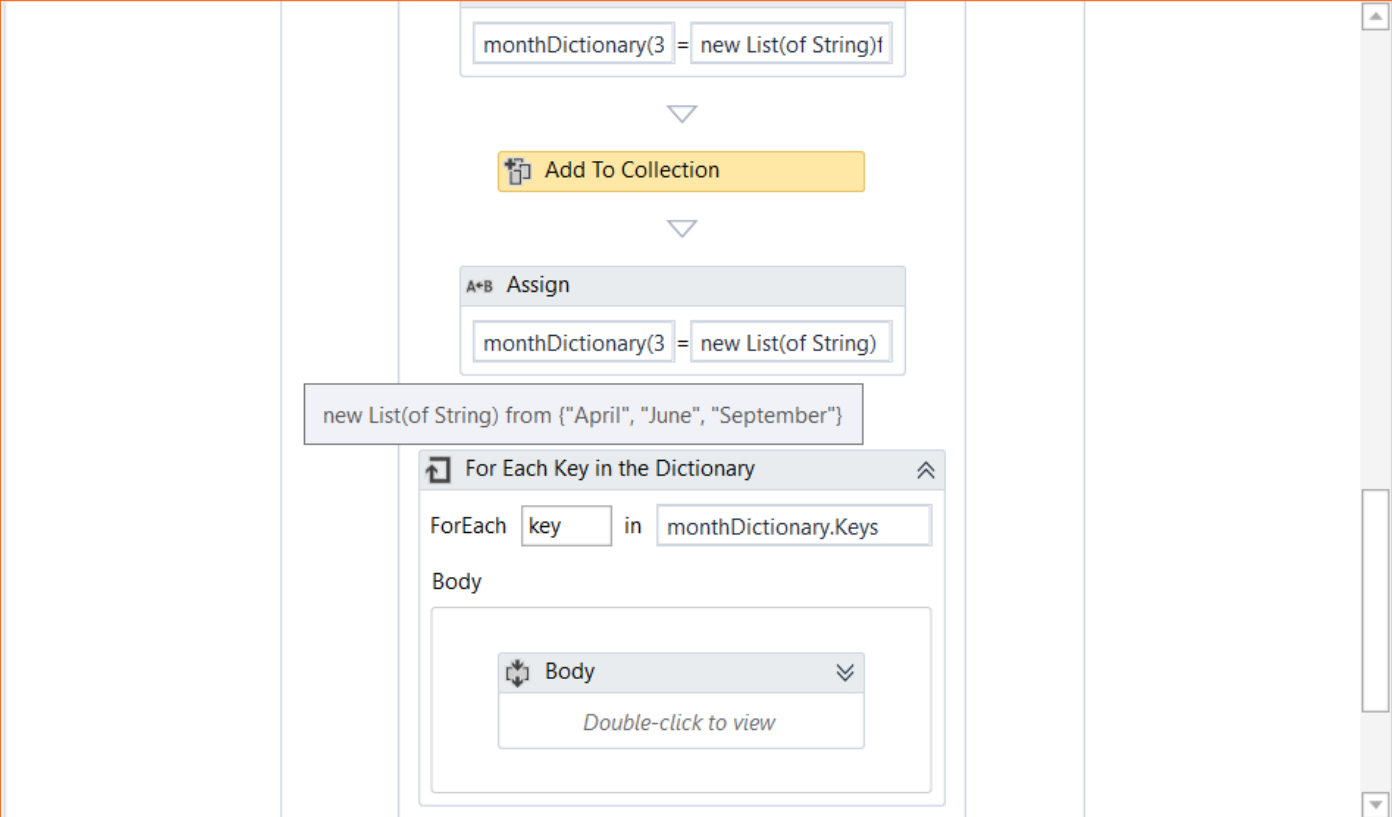
- ⓘ Dictionaries execution started
- ⓘ key: 10 Student: Ana Anca
- ⓘ key: 3 Student: me you her
- ⓘ Dictionaries execution ended in: 00:00:00

Variables Table:

Name	Variable type	Scope	Default
gradeDictionary	Dictionary<Int32,List<String>>	Sequence	new Dictionary(of Int32, List(of String)) from {{10, new List (of St

At the bottom of the interface, the 'Variables' tab is active, showing the variable 'gradeDictionary' with its type and scope. The status bar indicates 100% zoom and various tool icons.

Dictionaries. Example 3



The workflow diagram illustrates the process of adding items to a dictionary and iterating over its keys. It starts with an assignment of a new list to a dictionary key, followed by an 'Add To Collection' activity. Another assignment block is shown, followed by a 'For Each Key in the Dictionary' loop. The loop body contains a 'Body' activity with a 'Double-click to view' prompt.

Variables

Name	Variable type	Scope	Default
monthDictionary	Dictionary<Int32,List<String>>	Sequence	Enter a VB expression

Properties

Common

Property	Value
DisplayName	Add To Collection

Misc

Property	Value
Collection	monthDictionary(31)
Item	"March"
Private	<input type="checkbox"/>
TypeArgume...	String

Output

Search

- ⓘ Dictionaries execution started
- ⓘ days: 31 months: January March
- ⓘ days: 30 months: April June September
- ⓘ Dictionaries execution ended in: 00:00:01

Demo 4

- Create a process that performs the following actions:
 - 1. *read* pairs of (continent, country):
 - 1.1. *build* a dictionary of countries organized by continents;
 - E.g.: {"Asia-Japan", "North America-USA", "Europe-Romania", "South America-Argentina", "North America-Canada", "Asia-China", "Australia-Australia"}
 - 2. *print* the dictionary sorted by continents;
 - 3. *write* the dictionary details into a .txt file;
 - *use Append Line activity.*

Data Table. Details

- **Data Tables** characteristics in UiPath:
 - a **data structure with flexible length**;
 - it is similar to a Excel sheet consisting of rows and columns;
 - it can be iterated by using a **For Each Row** activity;
 - it is used for:
 - storing **data from Excel sheets** and .csv files;
 - web **data scrapping**.

Row/ Column	First	Last	Club Member
0	"John"	"Doe"	Yes
1	"Jane"	"Doe"	No
2	"Jane"	"Doe"	Yes
3	"John"	"Doe"	No

Data Table. Declaration. Instantiation. Initialization

- ways to declare/instantiate/initialize a dictionary:
 - **Variables Panel:**
 - **Name:** studentsTable; **Type:** DataTable;
 - **Read CSV** activity:
 - properties set:
 - FilePath = "members.csv";
 - IncludeColumnNames = *checked*;
 - DataTable = membersDataTable.

Row/ Column	First	Last	Club Member
0	"John"	"Doe"	Yes
1	"Jane"	"Doe"	No
2	"Jane"	"Doe"	Yes
3	"John"	"Doe"	No

Data Table. Operations (1)

- ways to **convert data table to String**:
 - **Output Data Table** activity:
 - properties set:
 - Input = membersDataTable;
 - Output = <a String variable>;
- ways to **access data by rows** in a data table:
 - **For Each Row** activity:
 - variable to iterate rows, e.g., **row**;
 - accessing a field in a **row** formed of [First, Last, Club Member] attributes:
 - firstName= row(**“first”**).ToString;
 - field name is case insensitive, e.g., first, First;

Data Table. Operations (2)

- ways to **access data** by indexing rows/columns in data table:
 - firstName = membersDataTable.Rows(1)("first").ToString;
 - status = membersDataTable.Rows(0)("club member").ToString;
 - lastName = membersTable.Rows(0)("Last").ToString;
 - lastName = membersTable.Rows(0)(1).ToString;
- ways to **filter data** by using rules in a data table:
 - **Select** method:
 - **Array of [DataRow]** filtered = membersTable.Select("first>'M' AND"+"[club member]='YES'");
 - the result is an **Array** iterated by **For Each** activity;
 - **Filter Data Table** activity:
 - it allows to follow a wizard that states the rules and the output columns;
 - the result is a Data Table iterated by **For Each Row** activity;

Data Tables. Example 1. Output Data Table

The image displays a UiPath workflow diagram and its corresponding interface. The workflow consists of the following steps:

- Read CSV**: Reads the file "members.csv".
- Output Data Table**: Outputs the data to a variable named "membersTable".
- Write Line**: Writes the value of "output" to the console.
- Write Line**: Writes the value of `membersTable.Rows(0)("club member").ToString` to the console.
- Write Line**: Writes the value of `membersTable.Rows(0)(1).To` to the console.

The **Variables** pane at the bottom shows the following variables:

Name	Variable type	Scope	Default
membersTable	DataTable	Sequence	Enter a VB expression
output	String	Sequence	Enter a VB expression

The **Properties** pane on the right shows the following settings:

- Common**: DisplayName: Output Data Table
- Input**: DataTable: membersTable
- Misc**: Private: ☐
- Output**: Text: output

The **Output** pane shows the following output:

```
First,Last,Club Member
Jay,Gavin,Yes
Zachary,Craig,No
Sherry,Hooks,Yes
Marcella,Knipp,Yes
Melvin,White,No
Nancy,Fox,Yes
Kristina,Turner,No
Craig,Saucier,No
Raymond,Wilson,No
Kent,Danley,No
Yes
Gavin
```

Data Tables. Example 2A. Select method

The screenshot displays a UiPath workflow designed to filter and display data from a Data Table. The workflow consists of the following steps:

- Assign:** A yellow box labeled "A*B Assign" contains the expression `filteredDataRows = membersTable.Select`.
- Expression:** A box below the Assign step contains the full expression: `membersTable.Select("first>'M' AND"+"[club member]='YES'")`.
- For Each:** A loop block labeled "For Each" iterates over `filteredDataRows`. The loop body contains a **Write Line** activity with the text `item("first").ToString`.

On the right side of the interface, the **Properties** pane shows the **Misc** section with the following settings:

- Private:** ☐
- To:** `filteredDataRows`
- Value:** `membersTable.Select`

The **Output** pane shows a list of names: Marcella, Nancy, Sherry, Zachary, Kristina, Craig, Raymond, and Kent. Below the list, it states "DataTables execution ended in: 00:00:00".

Name	Variable type	Scope	Default
filteredDataRows	DataRow[]	Sequence	Enter a VB expression

At the bottom of the interface, the **Variables** tab is active, showing the variable `filteredDataRows` with its type `DataRow[]` and scope `Sequence`. The status bar at the bottom indicates a zoom level of 100%.

Data Tables. Example 2B. Filter Data Table

The screenshot displays the UiPath Studio interface for a workflow titled "Filter Data Table". The workflow is structured as follows:

- Filter Data Table** (Yellow box): Contains a **Filter Wizard...** button.
- For Each Row** (Grey box): Iterates over **filteredDataTable**. The **ForEach** loop is configured with **row** in **filteredDataTable**.
- Body** (Grey box): Contains a **Write Line** activity with the text **row("first").ToString**.

The **Properties** pane on the right shows the configuration for the **Filter Data Table** activity:

- Common**: **DisplayName** is **Filter Data Table**.
- Input**: **DataTable** is **membersTable**.
- Misc**: **Private** is ☐.
- Options**: **FilterRowsMode** is **Keep**, **SelectColumn...** is **Keep**.
- Output**: **DataTable** is **filteredDataTabl**.

The **Output** pane shows the results of the workflow execution:

- Search**: A search bar.
- Output**: A list of names: **Marcella**, **Nancy**, **Sherry**, **Zachary**, **Kristina**, **Craig**, **Raymond**, **Kent**, and **DataTables execution ended in: 00:00:00**.

The **Variables** pane at the bottom shows the following table:

Name	Variable type	Scope	Default
filteredDataTable	DataTable	Sequence	Enter a VB expression

Data Tables. Example 2B. Filter Data Table - Wizard

Filter Wizard

Input DataTable: membersTable Output DataTable: filteredDataTable

Filter Rows Output Columns

Rows Filtering Mode

☒ Keep ☐ Remove

	Column	Operation	Value		
	first	Does Not Start With	M	x	+
And	club member	=	No	x	+

Filter Wizard

Input DataTable: membersTable Output DataTable: filteredDataTable

Filter Rows Output Columns

Columns Selection Mode

☒ Keep ☐ Remove

Column		
first	x	+
last	x	+
club member	x	+

OK Cancel

Demo 5

- Create a process that performs the following actions:
 - 1. *read* “Students.csv” file with the following structure: Student, Specialisation, Group;
 - *use Read CSV activity;*
 - 2. *print* the .csv file content;
 - *use Output Data Table activity;*
 - 3. *enter* a specialisation;
 - 4. *filter* students by specialisation and order them by group;
 - *use various ways:*
 - **filteredSortedDataTable** = (From row In studentsDataTable.Select("specialisation='"+spec+"'") Order By Convert.ToInt32(row("group")), row("student") Select row).ToArray.CopyToDatatable()
 - **filteredDataTable** = studentsDataTable.Select("specialisation='"+spec+"'")
 - **sortedDataTable** <== **Sort Data Table** activity
 - 5. *save* the resulted data into a .csv file.
 - *use Write CSV activity to write a Data Table object to a .csv file.*

References

- UiPath Academy - <https://academy.uipath.com>
 - Awareness Training;
 - Level 1 – Foundation Training, Lesson 3;
- UiPath Docs - <https://docs.uipath.com/studio>
 - Dictionary - <https://docs.microsoft.com/en-us/dotnet/api/system.collections.generic.dictionary-2?view=netframework-4.8>
 - Data Table variables - <https://docs.uipath.com/studio/docs/data-table-variables>