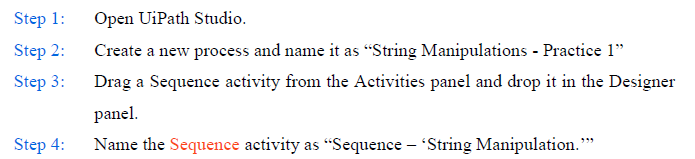
**EXP2:**

**Build a workflow using Format, Join, IndexOf, Split, and Substring methods that extract key information from a text and prints in a different format.   
- Use the text "You always wanted to study Automation Training. The materials are available in the following : UiPath Blog, UiPath Academy.” for extraction.   
- Extract “Automation Training” from the first sentence. - Extract “UiPath Blog” and “UiPath Academy” from the second sentence.   
- Display “study Automation Training from: UiPath Blog; UiPath Academy” in a message box.**



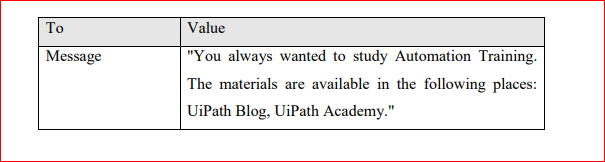
Step 5:

In the Variables panel, create three variables as shown below:

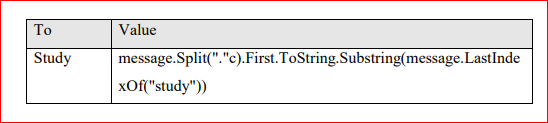
|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Variable type** | **Scope** | **Default** |
| Message | String | Sequence - String Manipulation |  |
| Study | String | Sequence - String Manipulation |  |
| places | List<String> | Sequence - String Manipulation |  |

Note: Select data type List<String> from System.Collections.Generic.List<System.String>

Step 6: Drag and drop an Assign activity in the Sequence activity and name it as “Assign- Message”. Enter values as shown below:



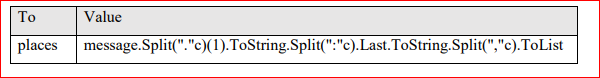
Step 7: Insert another Assign activity and name it as “Assign- Extracting (Automation Training)”. Enter values as shown below:



Note: 1. Split("."c).First.ToString extracts the first sentence of the String and converts it to a String.

2. Substring(message.LastIndexOf("study")) extracts the Substring from "study"

Step 8: Insert another Assign activity and name it as “Assign – Extracting (UiPath Blog and UiPath Academy)”. In the Assign activity, enter values as shown below:



Note : “1. message.Split("."c)(1).ToString extracts the second sentence of the String and converts it to a String.

2. Split(":"c).Last.ToString splits the remaining String and keeps only the last part of it.

3. Split(","c).ToList takes each string separated by a comma and adds it as an element in the List variable”.

Step 9: Insert a Message Box activity. In the text area of the Message Box activity enter the expression:



String.Format: This method formats a string using placeholders like {0}, {1}, etc. Each placeholder is replaced by the corresponding value passed as arguments.

{0} will be replaced by the value of study.

{1} will be replaced by the result of String.Join(";", places).

String.Join(";", places): This method takes an array or collection of strings, places, and joins them into a single string with each element separated by ;.

**EXP 4:**

**Build a workflow using Split and Contains methods that extract sentences containing “RPA” from a paragraph.   
- Store a paragraph in a string variable using an Assign activity.   
- Store all sentences from the text in an array using a Split method.   
- Loop through each sentence and identify sentences containing “RPA” using Contains method. - Store all identified sentences in an MS Word file.**

Step 1: Open a new MS Word file.

Step 2: Open UiPath Studio.

Step 3: Create a new process and name it as “String Manipulations”

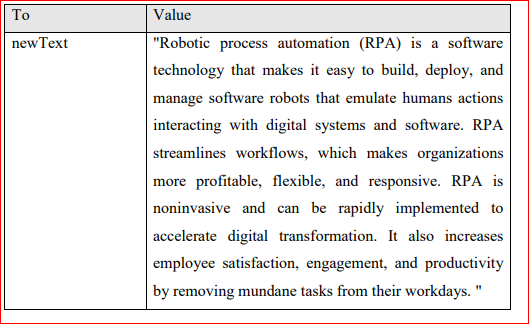
Step 4: Drag a Sequence activity from the Activities panel and drop it in the Designer panel.

Step 5: Name the Sequence activity as “Sequence – ‘This code is to demonstrate the use of Split and Contains method.’”

In the Variables panel, create variables as shown below:

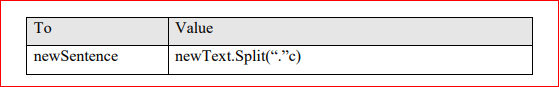
|  |  |  |  |
| --- | --- | --- | --- |
| Name | Variable type | Scope | Default |
| newText | String | Sequence - This code is to Demonstrate the use of Split and Contains. |  |
| newSentence | String[] | Sequence - This code is to Demonstrate the use of Split and Contains. |  |

Step 6: Drag an Assign activity and Drop it into the “Sequence - This code is to Demonstrate the use of Split and Contains.” Name the Assign activity as “Assign - Message”,



Step 7: Drag and drop another Assign activity and place it after the first Assign activity.

Name the Assign activity as “Assign - Split sentences and store it in an array.” and enter the values as shown below:



Step 8: Drag and drop a For Each activity, in the first box enter **item** and in the second box enter **newSentence**. Change the “**Type Argument**” property of For Each activity as “String”.

Step 9: In the Body section of the For Each activity, drag and drop an If activity. Name it as “If - item contains word "RPA"”, and add an annotation: “This activity judges whether the sentence contains word ‘RPA’ or not.” and enter condition as **item.Contains("RPA")**

Step 10: Drag and drop an Attach window activity, place it in the **Then** section of the If activity and name it as “Attach Window - MS Word”.

Step 11: Click on the “Indicate element on screen” link and select the MS Word window.

Step 12: Drag and drop a Type Into activity in the Do container of the Attach Window activity and name it as “Type Into - MS Word”.

Step 13:Click on the “Indicate element on screen” link and select editor area of MS Word.

Step 14: In the text area of the Type Into activity enter expression: **item.ToString+ "[k(enter)]"**

**Save and Run the Workflow.**

**EXP 5 :**

**Build a workflow using Concat and Join method that merges two lists containing the UK and Spain city names, sorts it, capitalizes the first letter of each item, and displays it in a message box.**

**- Create a list containing three UK cities in all capital letters.**

**- Create another list containing three Spain cities in small letters.**

**- Merge both the lists together.**

**- Sort the final list in alphabetical order from A to Z.**

**- Capitalize only the first letter of all the items in the final list.**

**- Display the final list in a message box in string format.**

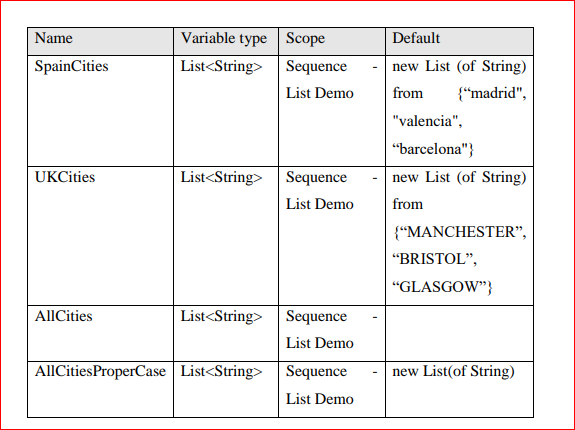
Step 1: Open UiPath Studio.

Step 2: Create a new process and name it as “Lists”

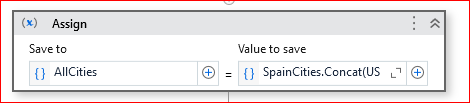
Step 3: Drag a Sequence activity from the Activities panel and drop it in the Designer panel.

Step 4: Name the Sequence activity as “Sequence – ‘List Demo’”

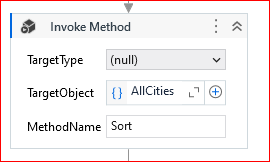
Step 5: Create three variables through the Variables panel as shown below:



Step 6: Drag and drop an Assign activity and name it as “Assign – Merge”, Add an annotation: “Merging Lists”. Enter the values as shown below:



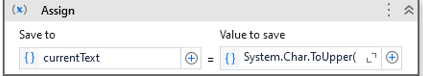
Step 7: Drag and drop an Invoke Method activity, and enter values as shown below:

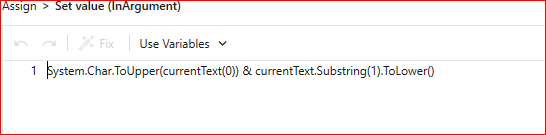


Step 8: Drag and drop a For Each activity and name it as “For Each City”. In the first box of the For Each activity enter “item” and in the second box enter the variable **AllCities .**

In the properties panel of the For Each activity, Change the Type Argument to ‘String’.

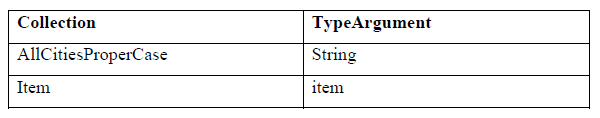
Step 9: In the Body section, drag and drop an Assign activity and name it as “Assign ProperCase” . In the Assign activity, enter values as shown below:



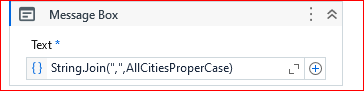


Step 10: Drag and drop an Add to Collection activity below the Assign activity.

In the properties panel of the Add to Collection activity enter values as shown below:



Step 11: Drag and drop a Message Box activity and name it as “Message Box – AllCitiesProperCase” . In the text area of the Message Box activity, enter the expression: **String.Join(",",AllCitiesProperCase)** as shown below.

****

**Save and run the workflow**

**EXP 3: Build a workflow using data table activities to join two library databases using matching**

**student ID and display the output in a message box.**

**- Create a data table variable and populate it with student ID and name of students.**

**- Create another data table variable, and populate it with student ID and book names**

**- Join both the data tables based on matching student ID.**

**- Remove the student ID column and sort the final data table as per student names in**

**alphabetical order from A to Z.**

**- Display the final data table containing the student and book names in a message box as**

**a string.**

Step 1: Open UiPath Studio.

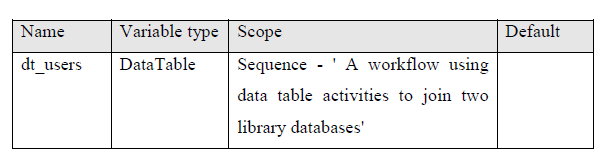
Step 2: Create a new process and name it as “Data Table Manipulations”.

Step 3: Drag a Sequence activity from the Activities panel and drop it in the Designer panel.

Step 4: Name the Sequence activity as “Sequence – ‘A workflow using data table activities to join two library databases’”.

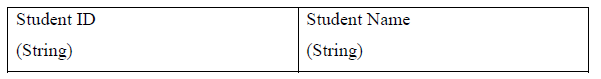
Step 5: Drag and drop a Build Data Table activity and name it as “Build Data Table – Users”. Add an annotation: “Data table contains Columns Student ID, Student Name”.

Step 6: Create a variable through Variables panel for the Build Data Table activity as shown below:



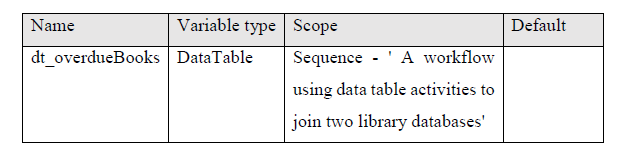
Step 7: In the Properties panel of the Build Data Table activity, enter **dt\_users** in the Output property.

Step 8: Click on the “Data Table” button and create a Data Table as shown below:



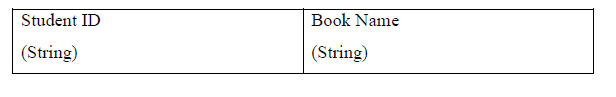
Step 9: Drag and drop a Build Data Table activity and name it as “Build Data Table – Overdue Books”. Add an annotation: “Datatable contains columns Student ID, Book Name”.

Step 10: Create a variable through Variables panel for the Build Data Table activity as shown below:



Step 11: In the Properties panel of the Build Data Table activity, enter **dt\_overdueBooks** in the Output property.

Click on the “DataTable...” button and create a Data Table as shown below:



Step 12: Drag and drop Join Data Table activity and name it as “Join Data Tables - Join dt\_Users and dt\_overdueBooks Datatable”. Add an annotation: “This activity joins both the data tables resulting in a single datatable”.

Step 13: Click on the “Join Wizard” button.

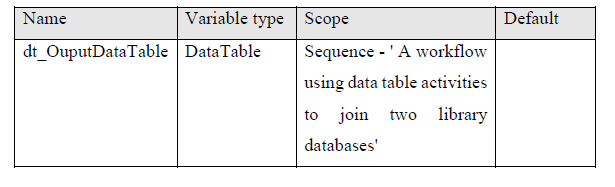
Step 14: Enter **dt\_users** in Input Data Table 1, and enter **dt\_overdueBooks** in Input Data Table 2 text box.

Step 15: Select the Join Type as Inner, and in the Output Data Table text box, press **Ctrl + K** to create a new variable called **dt\_borrowedBooks**.

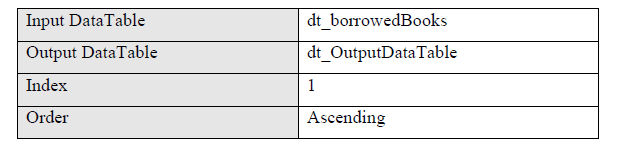
Step 16: In the Column Table 1, enter the column index as ‘0’ (column index of Data Table ‘dt\_users’), in the Column Table 2 enter the column index as ‘0’ (column index of Data Table ‘dt\_overdueBooks’) and press OK.

Step 17: Drag and drop a Remove Data Column activity. Name it as “Remove Data Column of dt\_borrowedBooks”. Add an annotation: “This activity removes the duplicate column of Student ID”. In its Properties panel, enter 2 in ColumnIndex property and enter **dt\_borrowedBooks** in DataTable property.

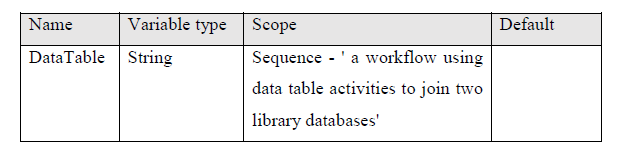
Step 18: Create a variable through the Variables panel as shown below



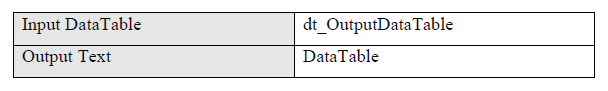
Step 19: Drag and drop a Sort Data Table activity. Name it as “Sort Data Column of dt\_borrowedBooks”. Add an annotation: “This activity sorts the DataTable in an ascending order.” and pass the values in the Properties panel as shown below:



Step 20: Create a variable through the Variables panel as shown below:



Step 21: Drag and drop an Output Data Table activity. Name it as “Output Data Column of dt\_OutputDataTable”., add an annotation: “This activity coverts DataTable's variable type from the datatable to string” and pass the values in the Properties panel as shown below:



Step 22: Drag and drop a Message Box activity, name it as “Message Box - Text”, add an annotation: “Prints the String form of DataTable” and in its text area enter the variable **DataTable**.

Save and run the workflow.