RPA Design and Development

v4.0

UiPath

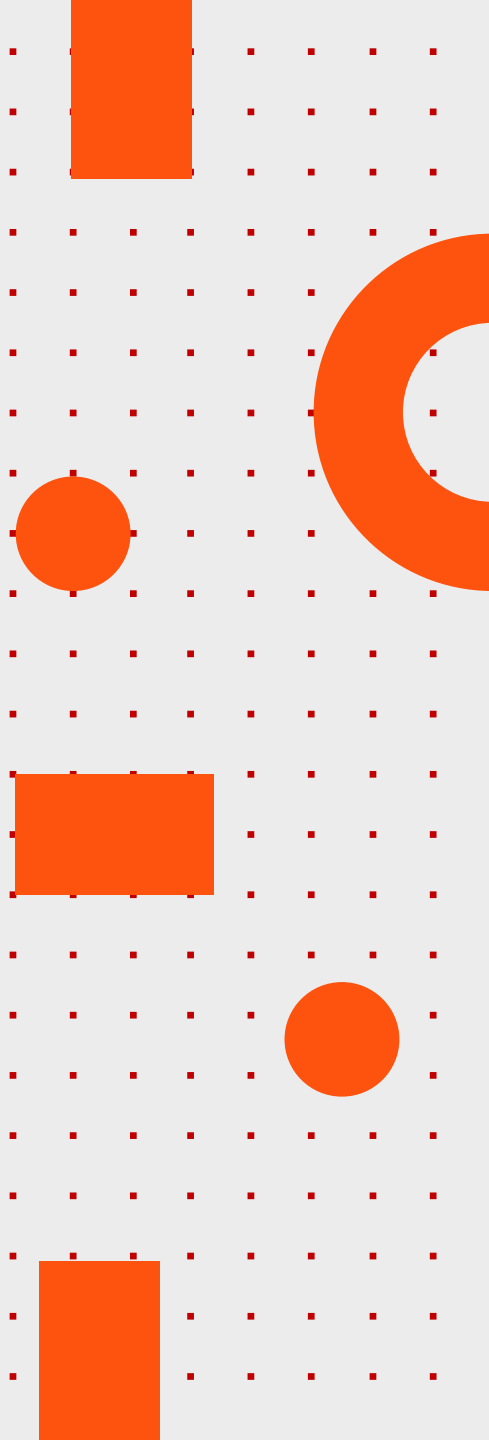
Lesson 15 ¦ Data Manipulation

Data Manipulation – Exam Topics

1. **Strings:** Use VB.Net methods to perform string manipulation: Trim, ToLower, ToUpper, StringConcat, Contains, String, Format, IndexOf, LastIdexOf, String.Join, Replace, Split, Substring.
2. **RegEx Builder:** Use the RegEx Builder for string manipulation.
3. **Arrays:** Initialize an Array variable, iterate through an array, reference elements, replace them and convert arrays of type string to string
4. **Lists:** Initialize a List type, iterate through it and use Studio activities to add, remove and modify the elements of the collection.
5. **Array/List differences:** Explain what the difference is between the Array type and the List type.
6. **Dictionaries:** Initialize a dictionary, add key-value pairs, remove keys and retrieve values.
7. **Datatables:** Use UiPath activities to build, filter, join, merge and iterate through DataTables.
8. **Conversions:** Perform conversions between the data types

Data Manipulation

- Introduction to Data Manipulation
- Operations for Data Manipulation
- Data Conversion



Introduction to Data Manipulation

Data Manipulation is the process through which the data is altered using various operations in order to facilitate its usage.

Why is Data Manipulation important?

Processes and organizes the raw data before it can be used

Makes the data easier to read

Customizes data according to the user's needs

Operations for Data Manipulation

Common operations for data manipulation are:

Retrieving

Specify conditions to extract, filter or summarize data

Adding

Add new data in a data structure

Deleting

Remove entries from a data structure

Modifying

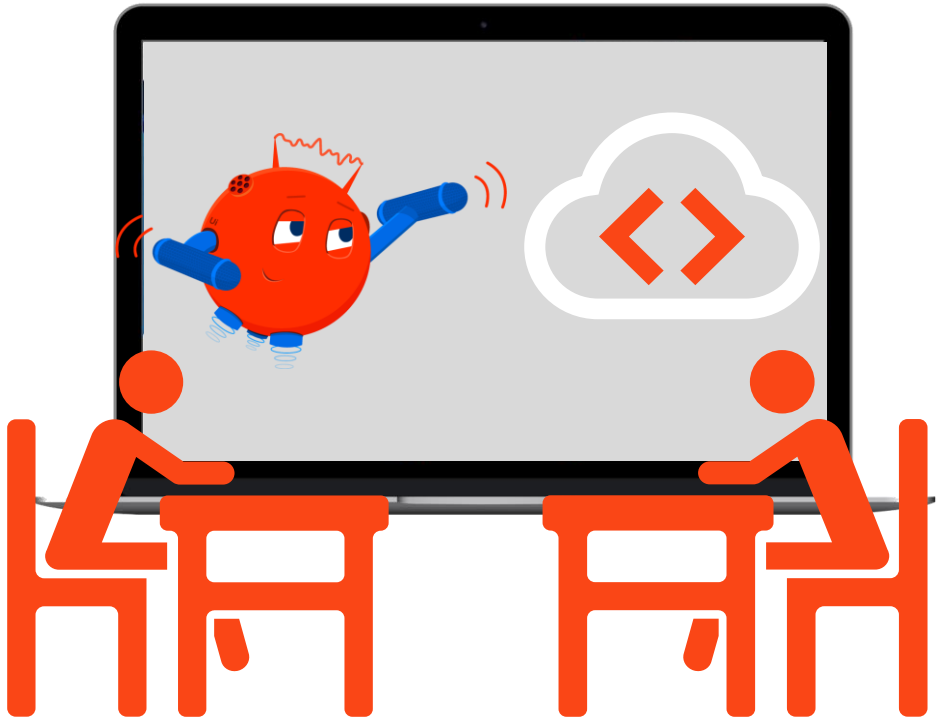
Change the existing entries in a data structure

Classroom Exercise

Demonstrate the process to convert an integer to a string using .**ToString** method.

- Ask the user to enter two numbers
- Add both the numbers
- Display the sum along with the text “The output is:” in a message box

Practice Exercise

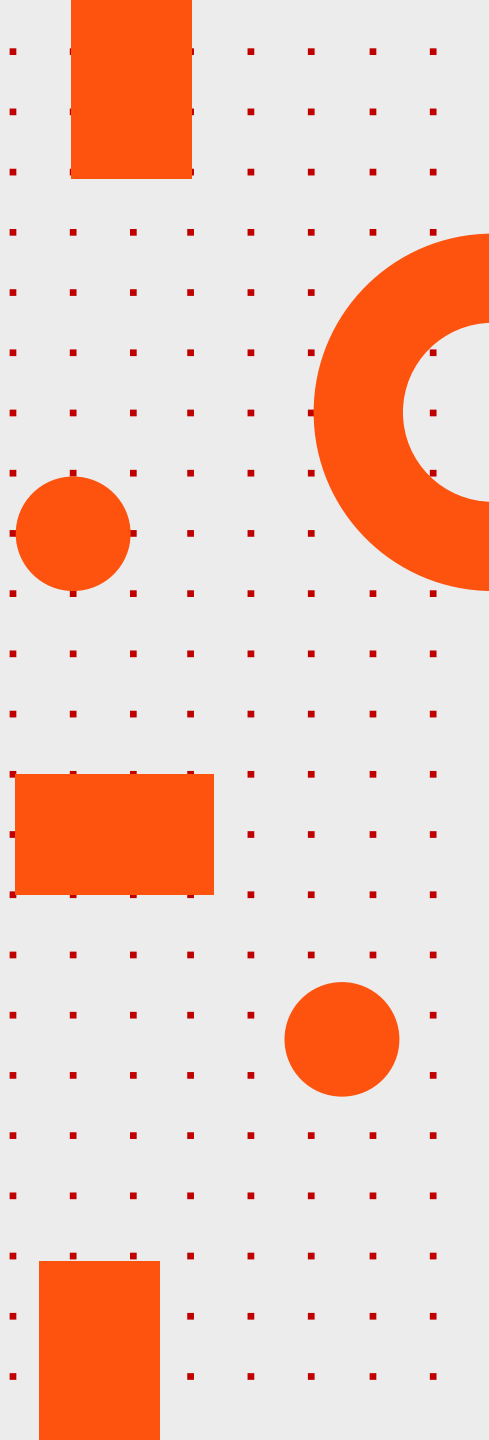


Build a workflow using the .**ToString** method that converts an integer to a string.

1. Ask the user to input their name and age
2. Subtract the age of the user with the current year to get the user's year of birth
3. Display the name and year of birth in a message box in string format

String Manipulation

- Introduction to Strings
- Methods for String Manipulation
- Regular Expression (RegEx)
  - RegEx Builder in Studio
  - RegEx Builder Wizard



Introduction to Strings

A String is the data type corresponding to text that contains a sequence of text. It is used when a text needs to be captured, processed, displayed or sent between applications.

Name	Variable type	Scope	Default
FirstName	String	Sequence	*Enter a VB expression*
LastName	String	Sequence	*Enter a VB expression*
CityName	String	Sequence	*Enter a VB expression*
Create Variable			

Methods for String Manipulation

Some of the operations that can be performed on strings are:

Concat

- Concatenates the string representations of two specified objects
- string.Concat (strVarName1, strVarName2)

Contains

- Checks whether a specified substring occurs within a string. Returns true or false
- Expression: VarName.Contains (“text”)

Format

- Converts an entire expression into a string (and inserts them into another text)
- Expression: String.Format(“{0} is {1}”, VarName1, VarName2)

IndexOf

- Returns the zero-based index of the first occurrence of a character in a string
- Expression: VarName1.IndexOf(“a”)

Methods for String Manipulation (Contd.)

Join

- Concatenates the elements in a collection and displays them as string
- Expression: String.Join("|", CollVarName1)

Replace

- Replaces all the occurrences of a substring in a string
- Expression: VarName.Replace ("original", "replaced")

Split

- Splits a string into substrings using a given separator
- Expression: VarName.Split("|"c)(index)

Substring

- Extracts a substring from a string using the starting index and the length
- Expression: VarName1.Substring(startIndex, length)

Methods for String Manipulation (Contd.)

ToUpper

- Converts lowercase letters to uppercase
- strVarName1.ToUpper

ToLower

- Converts uppercase letters to lowercase
- strVarName1.ToLower

Parse

- Converts a string (text) value to an integer (number) value
- Integer.Parse(strVarName1)

Trim

- Removes blank spaces before and after the string
- strVarName1.Trim

Methods for String Manipulation (Contd.)

Insert

- Adds a string to the specified position
- strVarName1.Insert(1,strVarName2)

Remove

- Removes a specified number of characters from a specified position in a string
- strVarName1.Remove(1,2)

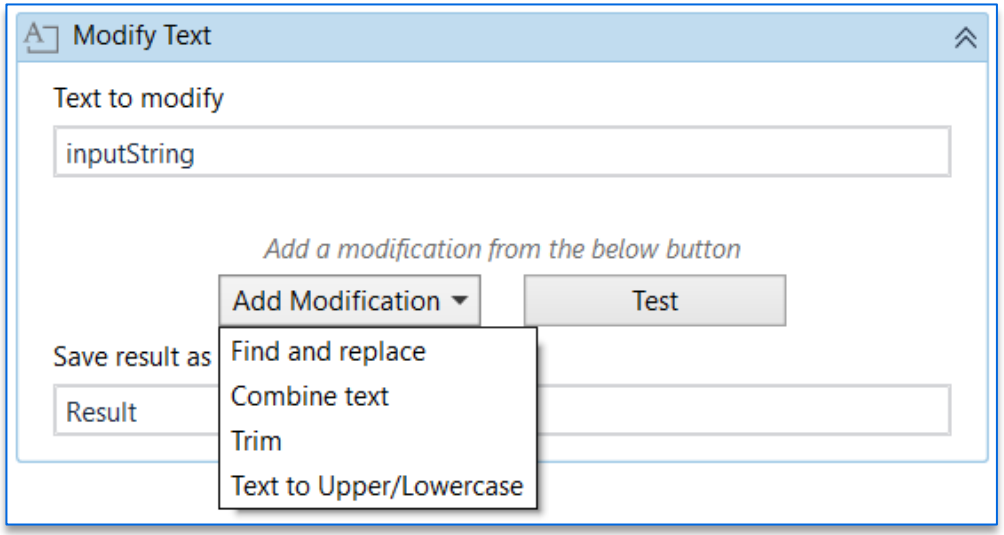
Length

- Returns the length of the string
- strVarName1.Length

Activities for String Manipulation

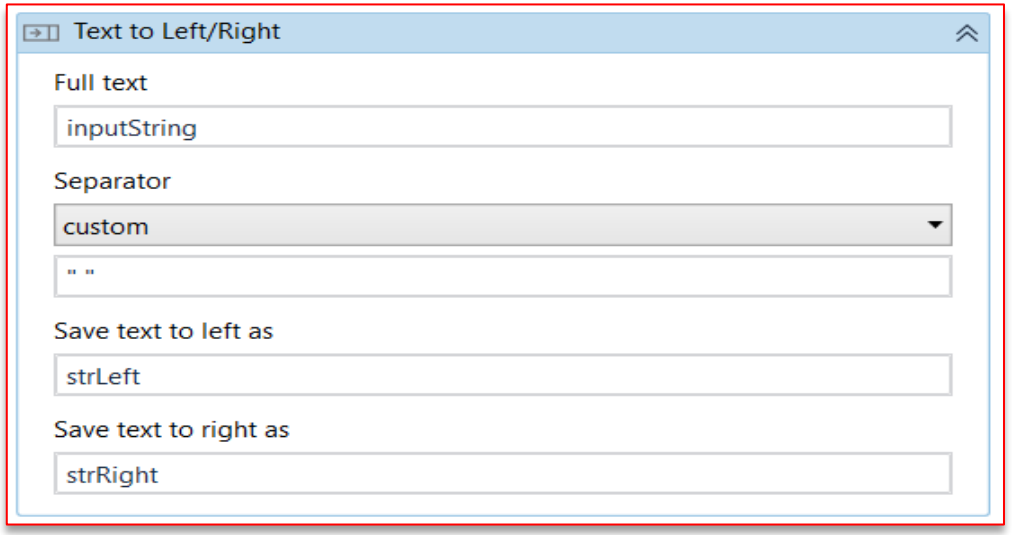
Modify Text

Updates a text value using modifications including find and replace, trim, combine (concatenating) with another text value, and changing to upper/lowercase



Text to Left/Right

Retrieves the text to the left and right of the first occurrence of the indicated subtext

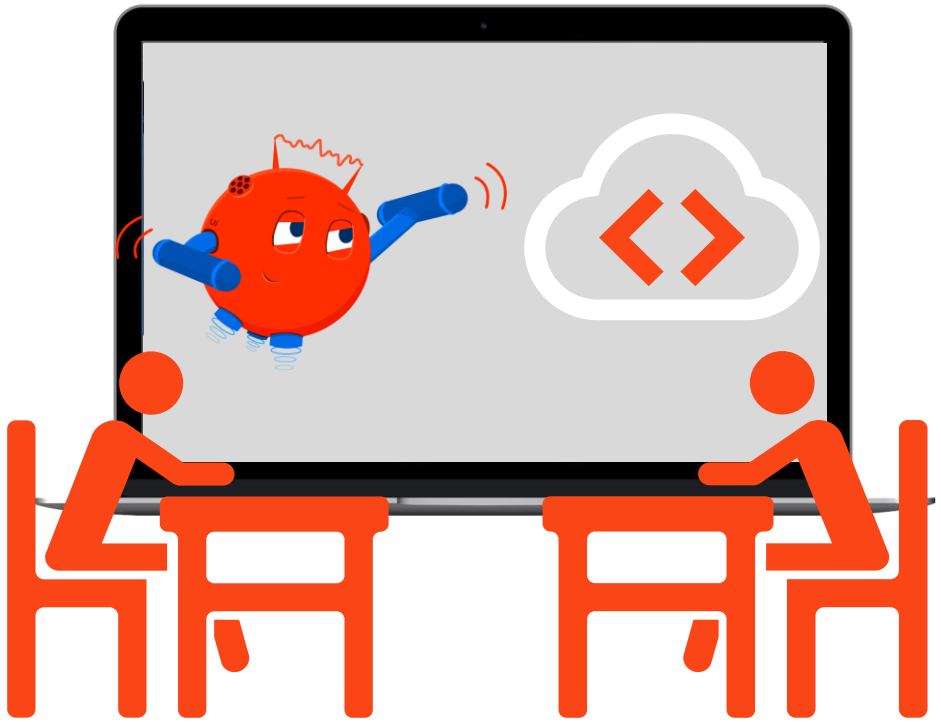


Classroom Exercise

Demonstrate the use of **Substring, Concat, Split, Format, and Replace** methods to manipulate strings.

- Use the initial text “I live in Norfolk county in England. It is a great place to visit.” for extraction
- Extract the text starting from “Norfolk” using the Substring method
- Display “Thank you for showing interest in touring **extracted text**.” in a message box. Replace **extracted text** with the text extracted from the initial text
- Ask for tour budget from the user and display “**budget** is too low to go to **extracted text**.” in a message box. Replace **budget** with users’ entered value and **extracted text** with the text extracted from the initial text
- Finally, display “But, **budget** is too good to go to Cornwall county in England.” in a message box. Replace **budget** with users’ entered value. Use Replace method to replace Norfolk with Cornwall county

Practice Exercise



Build a workflow using **Format, Join, IndexOf, Split, and Substring** methods that extracts key information from a text and prints in a different format.

1. Use the text "You always wanted to study Automation Training. The materials are available in the following places: UiPath Blog, UiPath Academy.” for extraction

2. Extract “Automation Training” from the first sentence

3. Extract “UiPath Blog” and “UiPath Academy” from the second sentence

4. Display “get Automation Training from: UiPath Blog; UiPath Academy” in a message box

Regular Expressions (RegEx)

It is a specific search pattern that can be used to easily match, locate and manage text.

- RegEx builder simplifies the creation of regular expressions

- Using RegEx with selectors, enables the users to identify multiple target element with a single search execution

- RegEx can be used in input validation, string parsing, data scraping, and string manipulation

RegEx Builder in UiPath

There are three activities in Studio that use the RegEx Builder. They are:

Searches an input string for all occurrences and returns all the successful matches

Specified regular expression finds a match in the specified input string

Replaces strings that match a regular expression pattern with a specified replacement string

RegEx Builder Wizard

Eases the process of building and testing Regular Expression search criteria. It can be opened from the body of any of the three activities (Matches, Is Match, and Replace).

Test Text

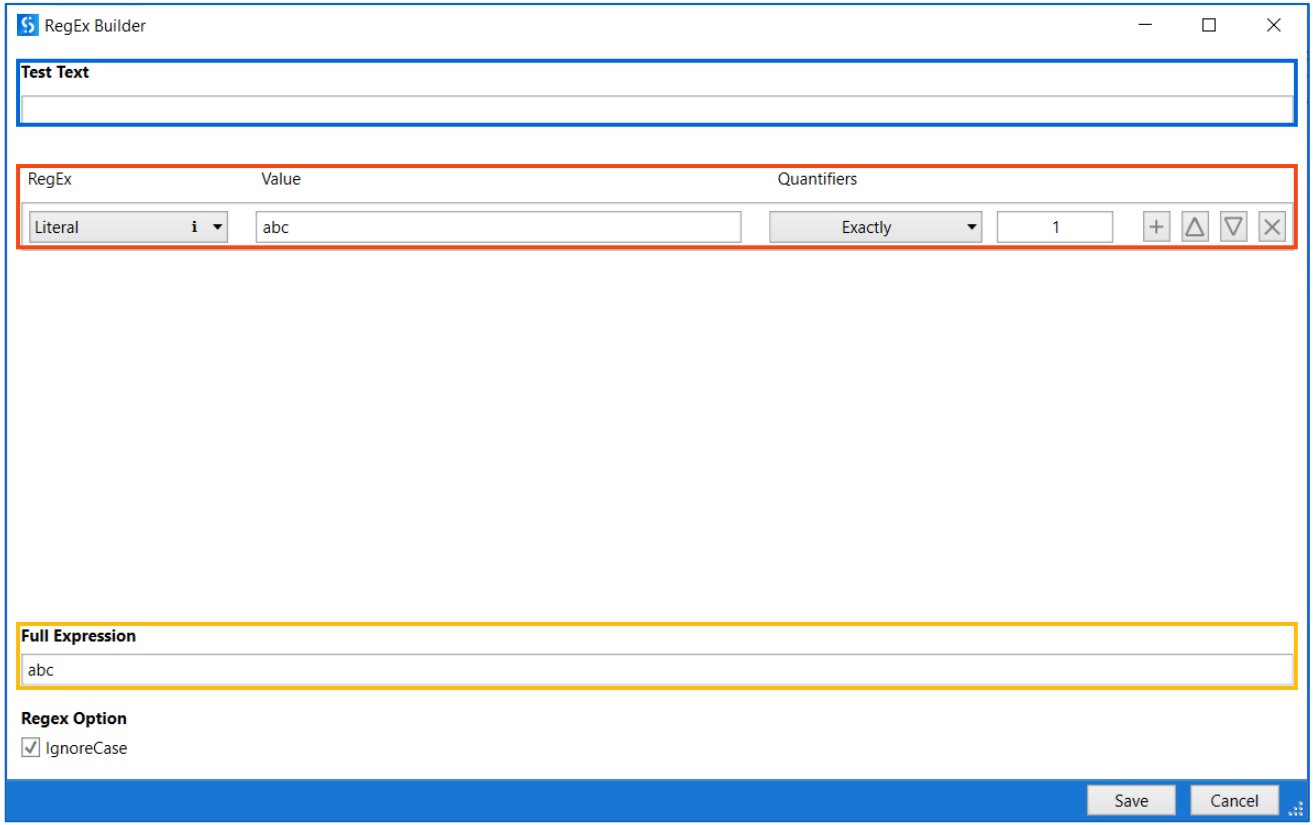
A text editor where the user can test the chosen search criteria against the text on which RegEx is applied

Type, Value, and Quantifiers

- **Type:** Allows searching for a given text or one expression from many.
- **Value:** Contains exactly the text that needs to be retrieved.
- **Quantifiers:** A dropdown list that enables the user to select the type of results that should be displayed

Full Expression

Displays the current RegEx expression in its raw form

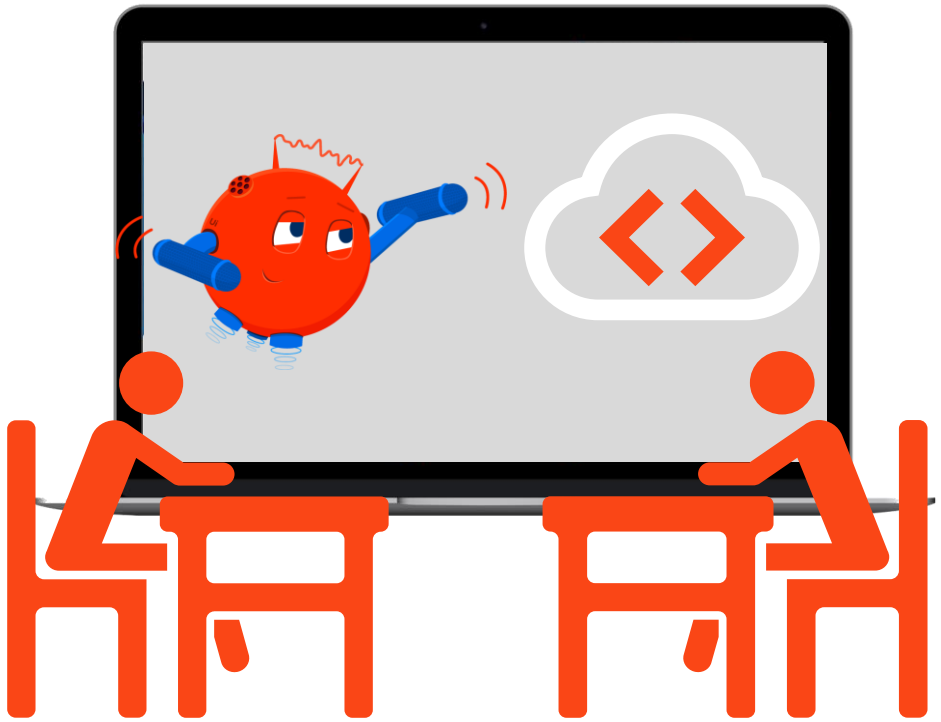


Classroom Exercise

Demonstrate the use of **Regex Builder** in order to extract email IDs from a text.

- Use the text “My name is Joe. My email is joe@mail.com, and my father’s email is jack@mail.com, and my uncle’s email is jay@mail.com” for extraction
- Extract email ids from the text using Regex Builder wizard
- Store the email ids in an MS Word file

Practice Exercise



Build a workflow using **Split** and **Contains** methods that extract sentences containing “RPA” from a paragraph.

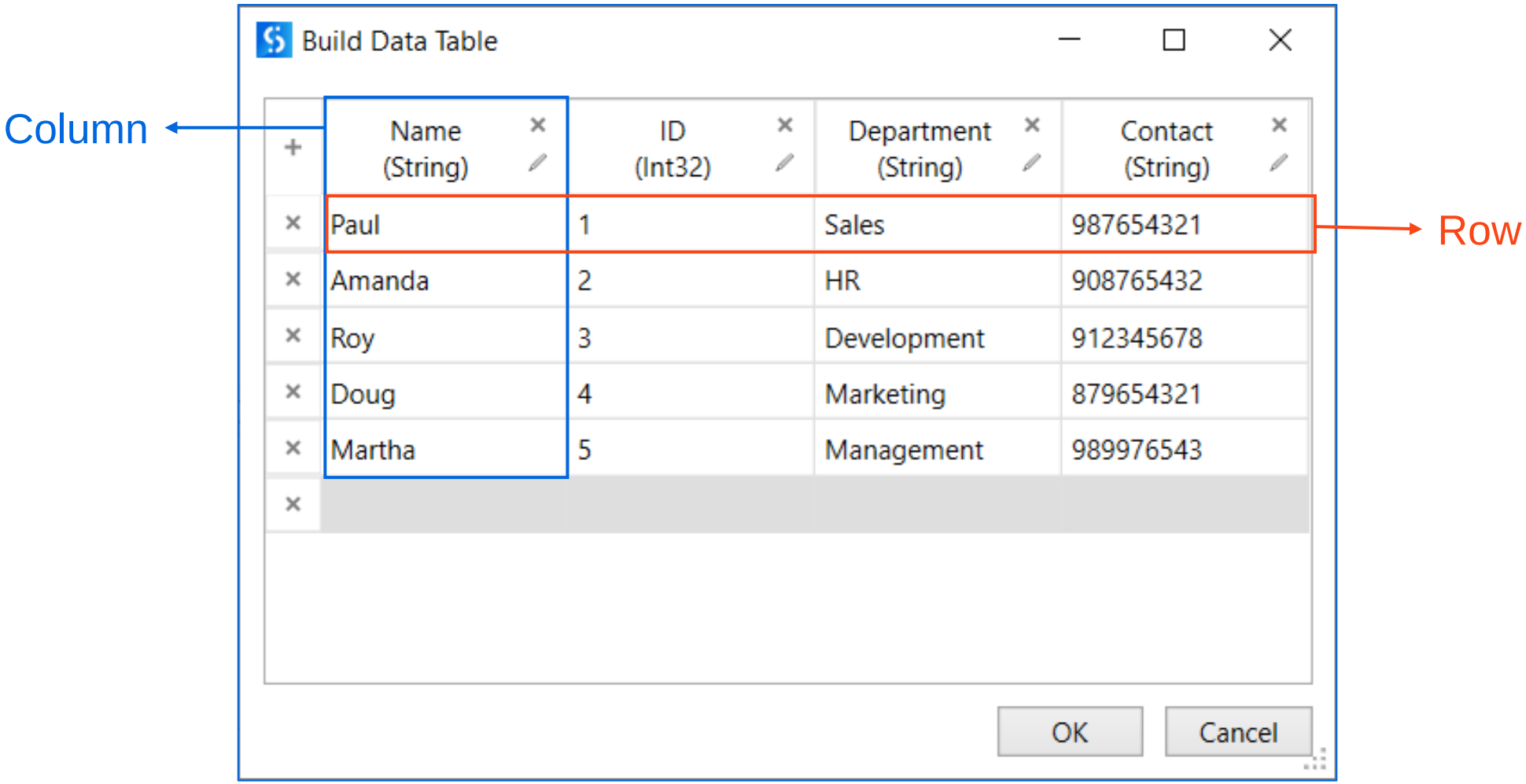
1. Store a paragraph in a string variable using an Assign activity
2. Store all sentences from the text in an array using a Split method
3. Loop through each sentence and identify sentences containing “RPA” using the Contains method
4. Store all identified sentences in a Notepad file

DataTable Manipulation

- Introduction to DataTable
- Creating DataTables
- Activities for DataTables Manipulation

Introduction to DataTable

A DataTable is an in-memory representation of a single database table that has a collection of rows and columns.



Creating DataTable

There are multiple ways to create DataTable. Some of the most common are:

Classroom Exercise

Demonstrate the use of **Build Data Table** and **Output Data Table** activities to store phone numbers in a data table and print it in the output panel in string format.

1. Create a data table and store five phone numbers in it

2. Print the phone numbers in string format in the Output panel

Activities for DataTable Manipulation

Some of the manipulations that can be done on DataTable are:

Add Data Column

- Adds a column to an existing DataTable variable
- The input data can be of DataColumn type, or the column can be added empty by specifying the data type and configuring the options

Add Data Row

- Adds a new row to an existing DataTable variable
- The input data can be of DataRow type or can be entered as an Array Row by matching each object with the data type of each column

Merge Data Table

- Appends a specified DataTable to the current DataTable
- The operation is simpler than the Join Data Type activity, as it has 4 predefined actions to perform over the missing schema

Activities for DataTable Manipulation (Contd.)

Some of the manipulations that can be done on DataTable are:

Lookup Data Table

- Allows searching for a provided value in a specified DataTable
- It returns the RowIndex at which it was found or can be configured to return the value from a cell with given coordinates

Filter Data Table

- Allows filtering a DataTable through a Filter Wizard, using various conditions
- It can be configured to create a new DataTable for the output of the activity or to keep the existing one and filter out entries

Join Data Tables

- Combines rows from two tables by using values common to each other
- It is one of the most useful activities in business scenarios, where working with more than one Data Table is very common

Activities for DataTable Manipulation (Contd.)

Generate Data Table From Text

Creates a DataTable from unstructured data by letting the user indicate the row and column separators.

For Each Row in Data Table

Used to perform a certain activity for each row of a DataTable (similar to a For Each activity).

Output Data Table

Writes a DataTable to a string using the CSV format.

Clear Data Table

Clears all the data in an existing DataTable.

Sort Data Table

Sorts a DataTable ascending or descending based on the values in a specific column

Activities for DataTable Manipulation (Contd.)

Remove Data Column

Removes a certain column from a specified DataTable. The input may consist of:

- Column index
- Column name, or
- Data Column variable

Remove Data Row

Removes a row from a specified DataTable. The input may consist of:

- Row index, or
- Data Row variable

Remove Duplicate Rows

Removes the duplicate rows from a specified DataTable, keeping only the first occurrence

Get Row Item

Retrieves a value from a row in a DataTable according to a specified column

Update Row Item

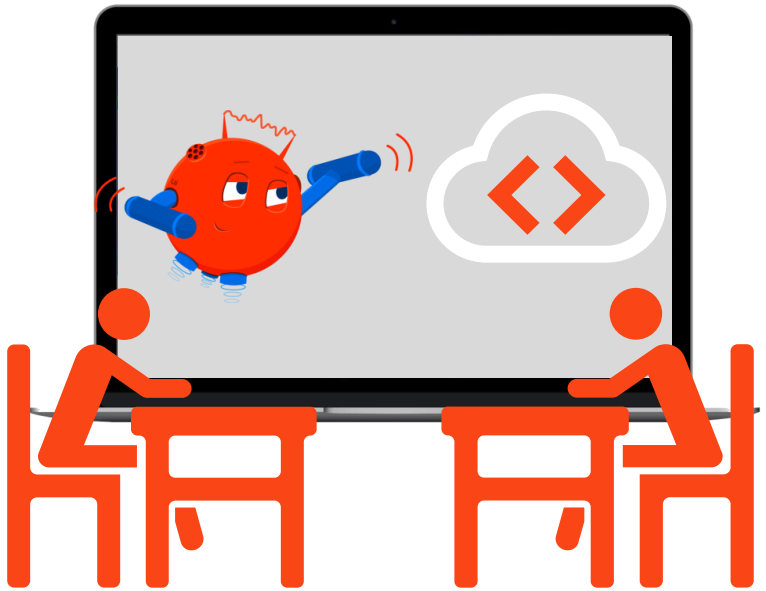
Assigns a specified value to the indicated column of a DataTable row

Classroom Exercise

Demonstrate the use of the **Sort Data Table** activity to sort a table.

- Create a data table to store names of five students and their ages
- Sort the data table based on students' names in alphabetical order from A to Z
- Print students' name and their age in the Output panel in a string format

Practice Exercise



Build a workflow using **DataTable** activities to join two library databases using matching student ID and display output in a message box.

1. Create a data table variable and populate it with student ID and the name of students

2. Create another data table variable, and populate it with student ID and book names

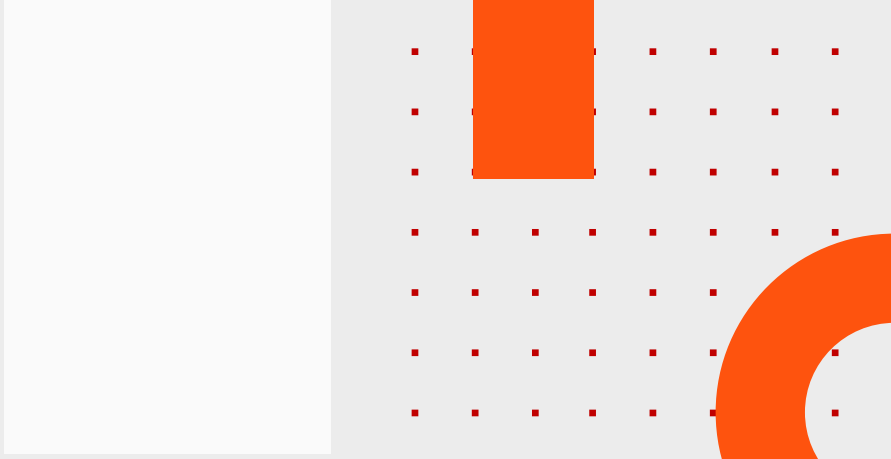
3. Join both the data tables based on matching student ID

4. Remove student ID column and sort the final data table as per student names in alphabetical order from A to Z

5. Display the final data table containing student and book names in a message box as a string

Collection, Its Types and Manipulation

- Collection and Its Types
- Collection Manipulation

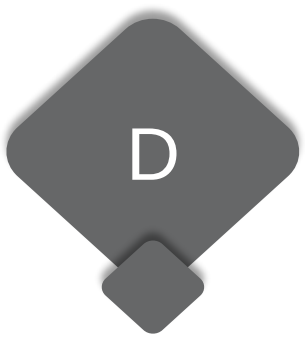


Collection and Its Types

A collection is used to represent a set of similar data type items in a single unit which is used for grouping and managing the objects. Different types of collections are:

Arrays

Lists



Dictionaries

Collection Manipulation

Studio offers the following methods for manipulating collections:

Add to Collection

- Adds an item to a specified collection
- Example: It can be used to add a new name to a list of company names

Remove From Collection

- Removes an item from a specified collection and can output a Boolean variable that confirms the success of the removal operation
- Example: It can be used to remove an invoice number from a list of invoices to be processed

Exists In Collection

- Indicates whether a given item is present in a given collection by outputting a Boolean as the result
- Example: It can be used to check whether a list of clients contains a specific name

Clear Collection

- Clears a specified collection of all items
- Example: It can be used to empty a collection before starting a new phase of a process that will populate it again

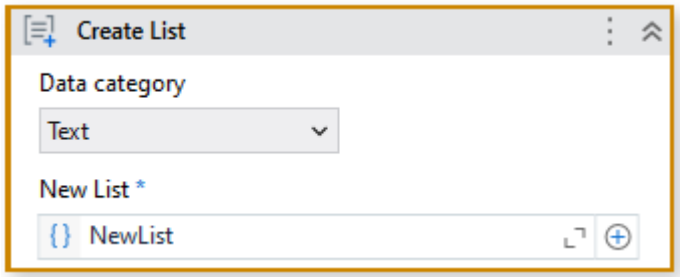
Lists

Lists are data structures consisting of objects of the same data type. Each object has a fixed position in the list. Lists provide specific methods of manipulation, such as:

Lists

Initializing a list

Activity : Create List – Empty List

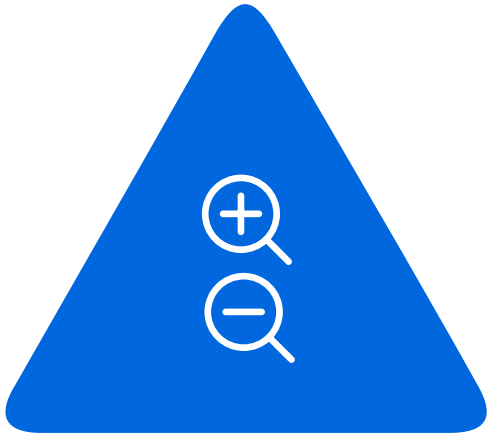


E.g. New List(Of String)

E.g. New List(Of String) ({“Apple”,”Orange”, “Pear”})

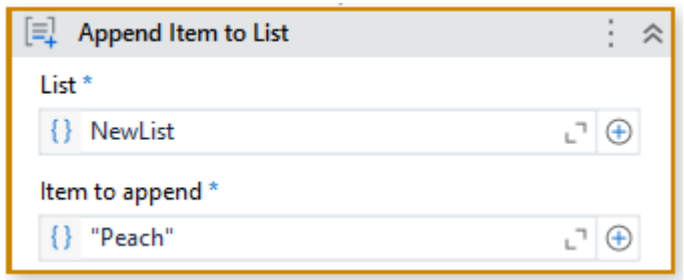
E.g. New List(Of String) from {“Apple”,”Orange”, “Pear”}

Lists

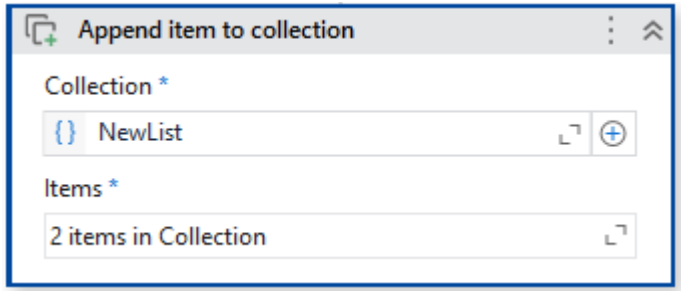


Adding and removing items

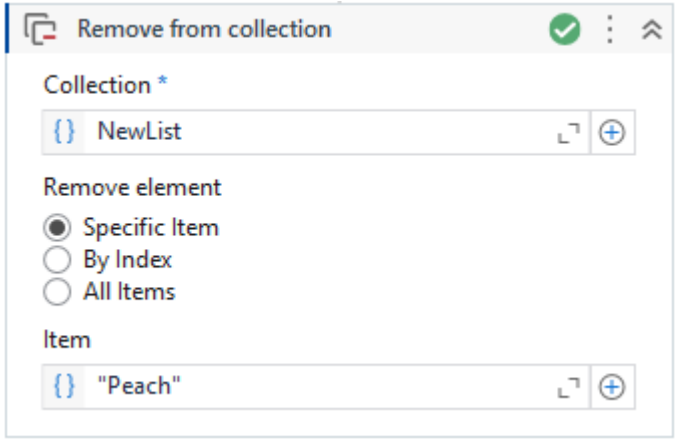
Activity : Append Item to List



Activity : Append Item to Collection



Activity : Remove from Collection

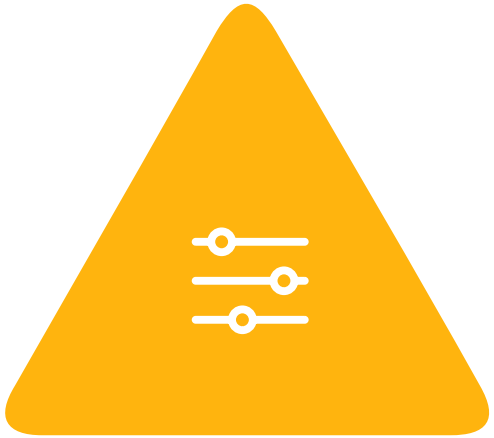


Lists

Searching for an element

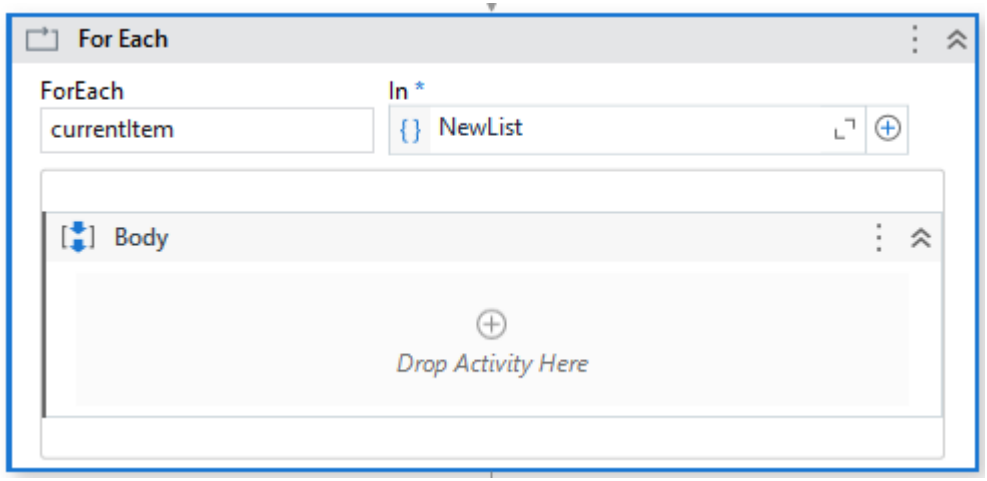
Activity : Exists in Collection

Lists



Looping through the items

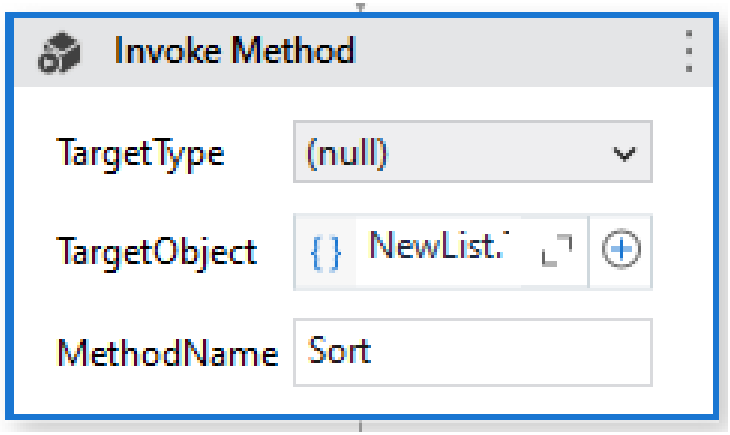
Activity : For Each

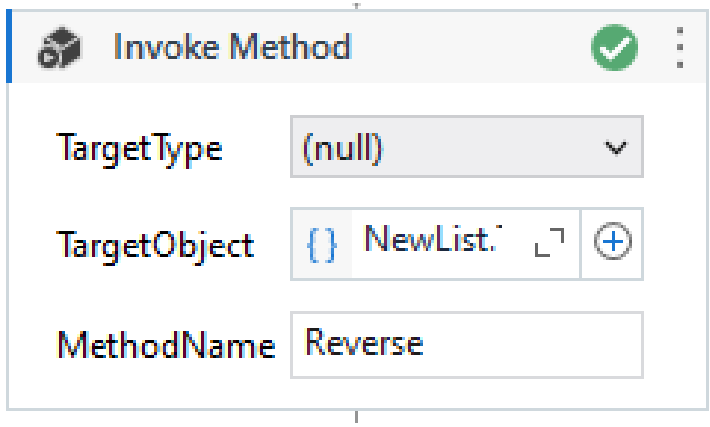


Lists

Sorting the objects

Activity : Invoke Method





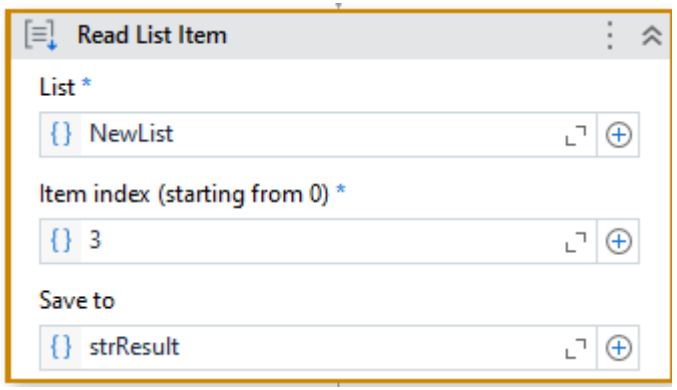
Lists

Extracting items & converting them to other data types

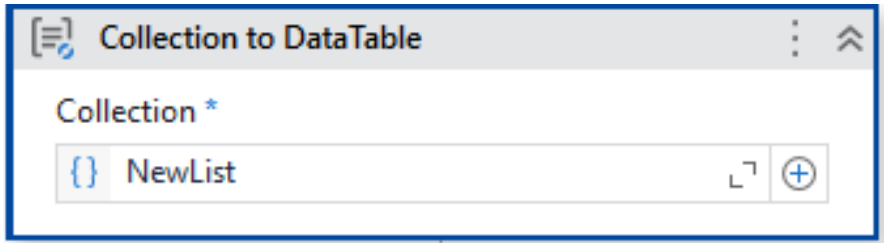
Collection

- Append Item to Collection
- Build Collection
- Collection to DataTable
- Exists In Collection
- Filter Collection
- Merge Collections
- Remove From Collection

Activity : Read List Item



Activity : Collection to Datatable

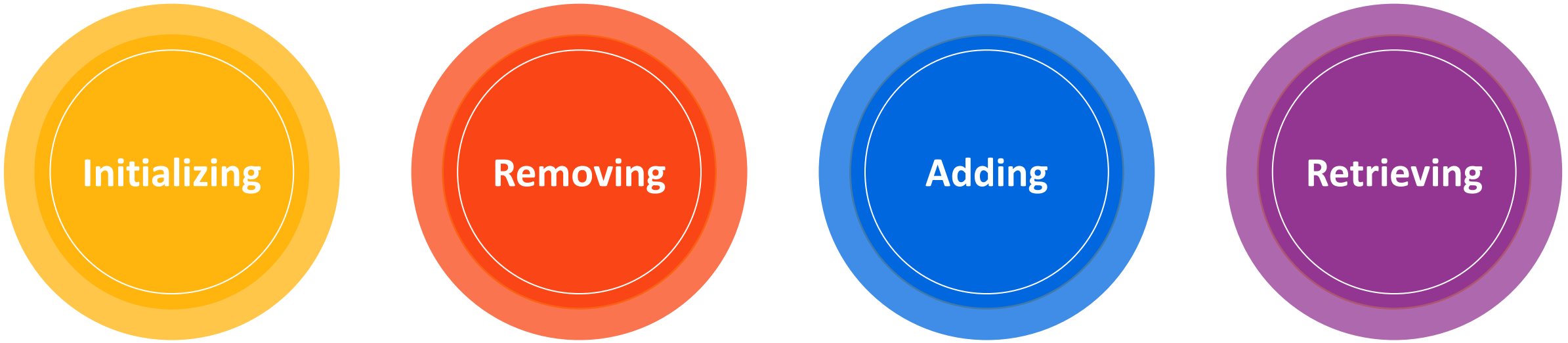


Dictionaries

Dictionaries are collections of (key, value) pairs in which the keys are unique. The data types for both keys and values are chosen when the variable is initialized. Operations on dictionaries include:

Working with Dictionaries

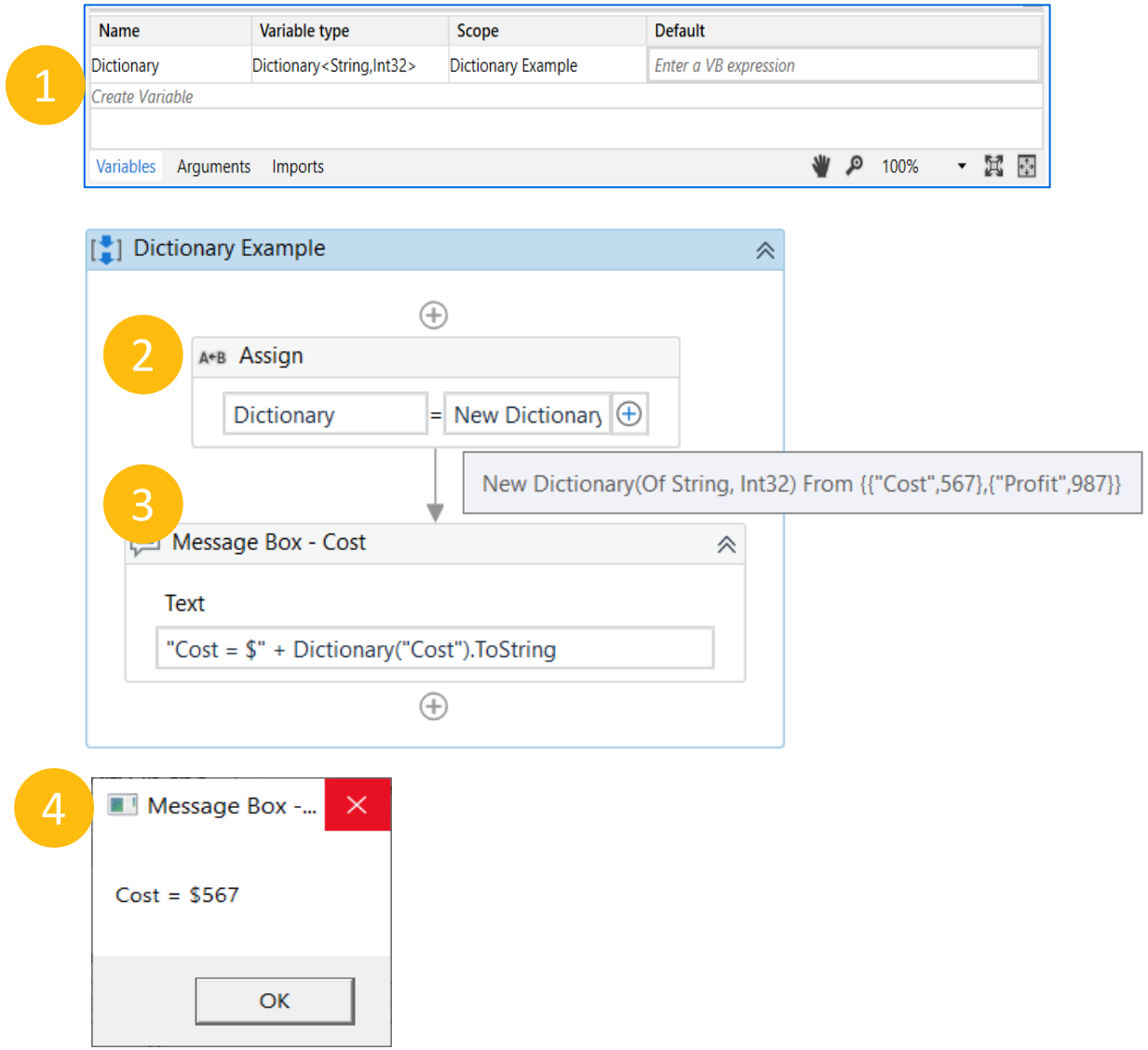
Dictionaries in Studio can be used in the following ways:



Methods for Working with Dictionaries

Initializing

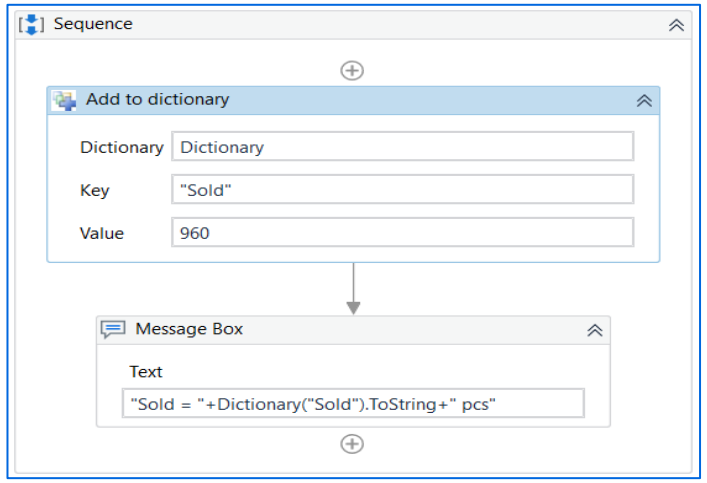
1. Create a variable of type Dictionary in the Variables panel
2. Drag and drop an Assign activity in the Designer panel
   - Enter Dictionary in the *To* textbox.
   - Initialize the dictionary variable in the *Value* textbox using **new Dictionary (Of String, Int32) From {{key, value}, {key, value}}**, if the key is String and the value is Integer
3. Insert a Message Box activity to display the added key and value pair
4. Execute the workflow. The Value of the first Key displays in the message box



Methods for Working with Dictionaries (Contd.)

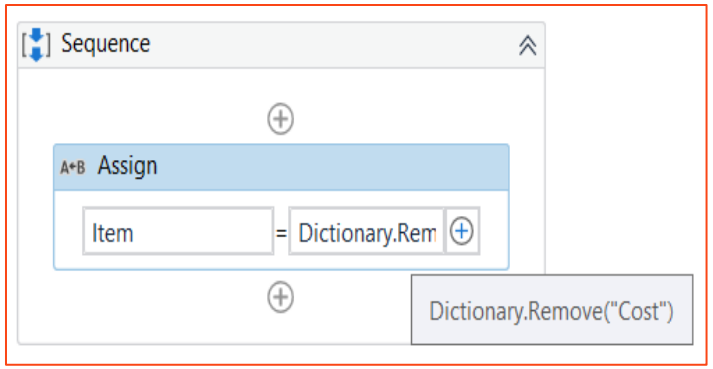
Adding

- Install the package Microsoft.Activities.Extension.
- Drag and drop the Add to Dictionary activity
- Enter the dictionary variable, key, and value in their respective fields



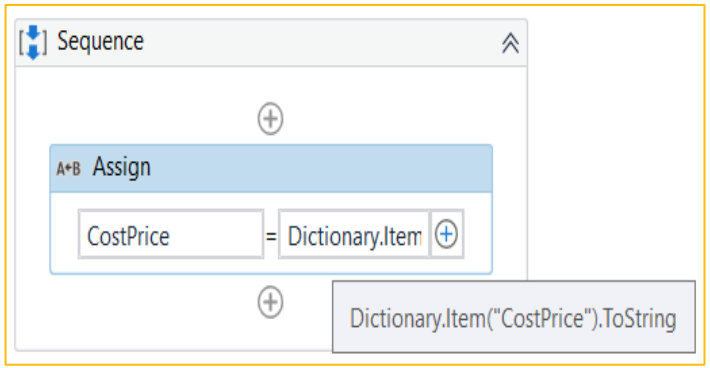
Removing

- VarName.Remove(Key) – removes an item from the Dictionary. It can be used in an 'Assign' activity



Retrieving

- VarName.Item(Key) – returns the Dictionary item by its key

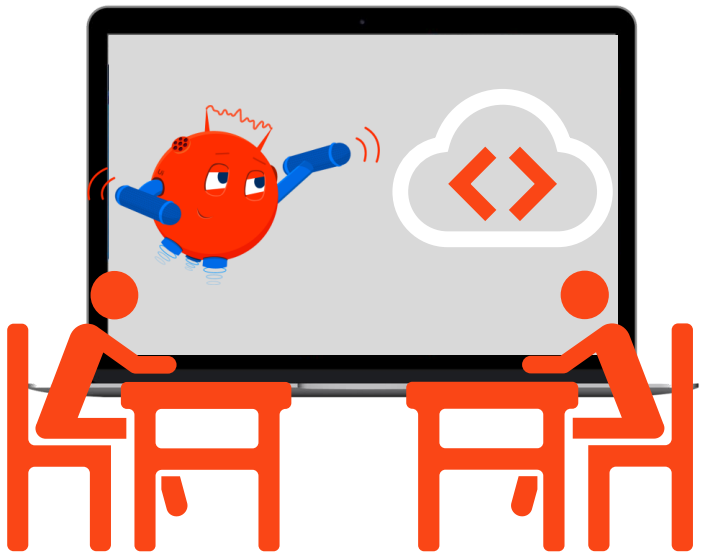


Classroom Exercise

Demonstrate the steps to sort a list in reverse order and print the first five items from the list in a message box.

1. Create a list of ten names of people
2. Sort the names in reverse alphabetical order from Z to A.
3. Extract the first five names from the list
4. Display the extracted names in a message box

Practice Exercise



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

1. Create a list containing three cities of the UK in Uppercase
2. Create another list containing three cities of Spain in Lowercase
3. Merge both the lists together
4. Sort the final list in alphabetical order from A to Z
5. Capitalize only the first letter of all the items in the final list
6. Display the final list in a message box in string format

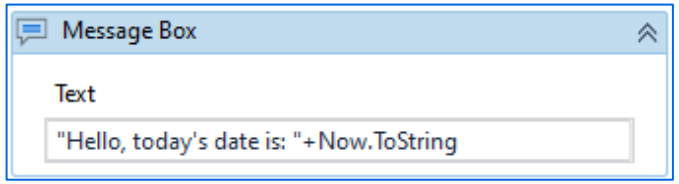
Data Conversion

Data Conversion is the process of converting one type of data to another. The methods for data conversion include:

Assign
Simply assigning the data value to the desired data type

.ToString Method
Convert any datatype to string

Raw DateTime



Formatted DateTime