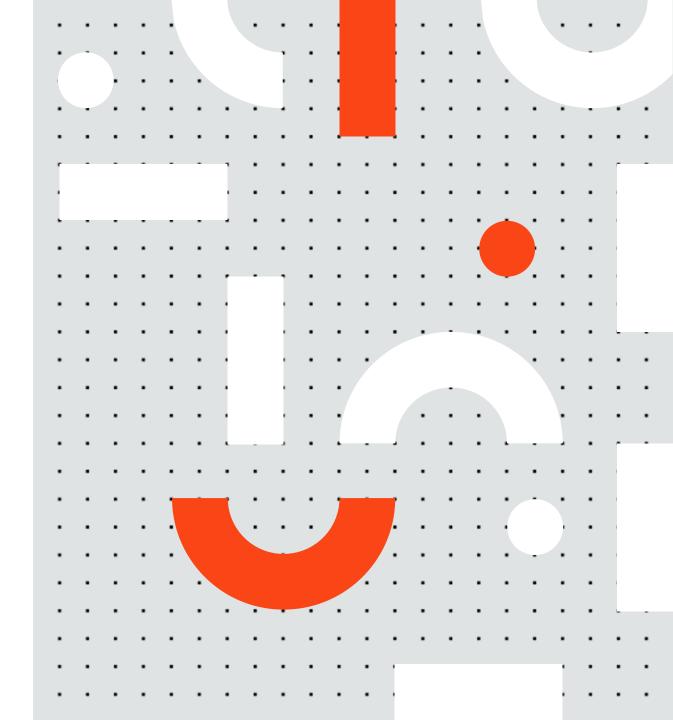
# RPA Design & Development v1.1

# Lesson 8 Data Manipulation







# Agenda



### **Basics of Data Manipulation**

- Introduction to Data Manipulation
- Data Query vs. Data Manipulation



# **Data Types**

Types of variables in UiPath



### **Data Manipulation Operations**

- Initializing
- Updating

Selecting

Deleting

Inserting

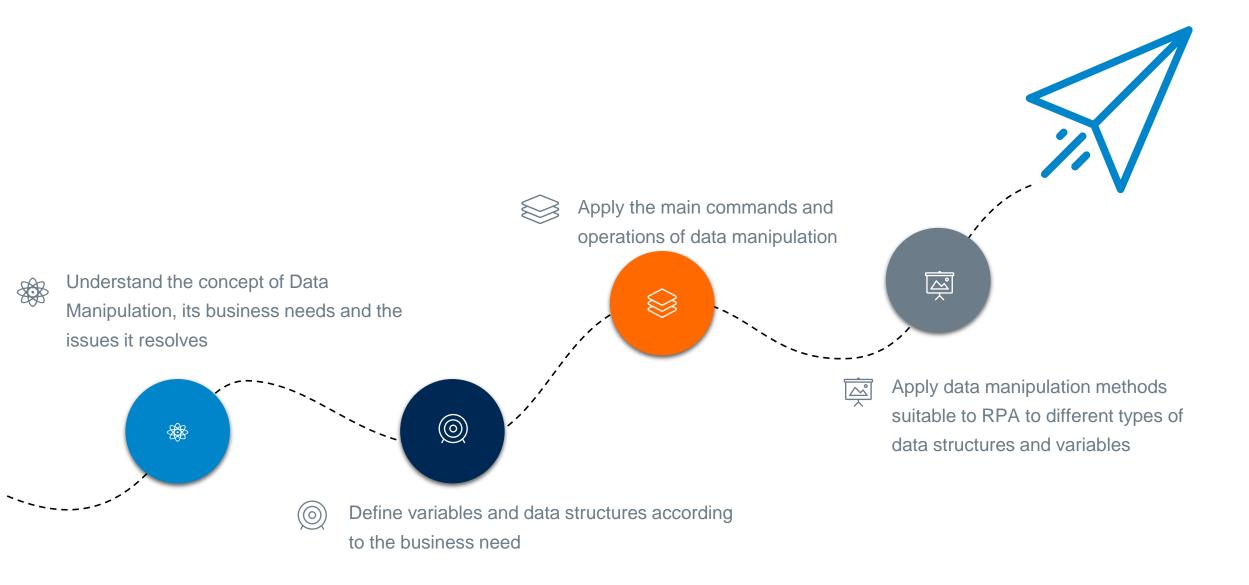


# **Text Manipulation**

String methods

# **Learning Objectives**









# **Basics of Data Manipulation**

- Introduction to Data Manipulation
- Data Query vs. Data Manipulation
- Data Manipulation Basic Operations



# **Data Types**

Types of variables in UiPath



# **Data Manipulation Operations**

- Initializing
- Updating

Selecting

Deleting

Inserting



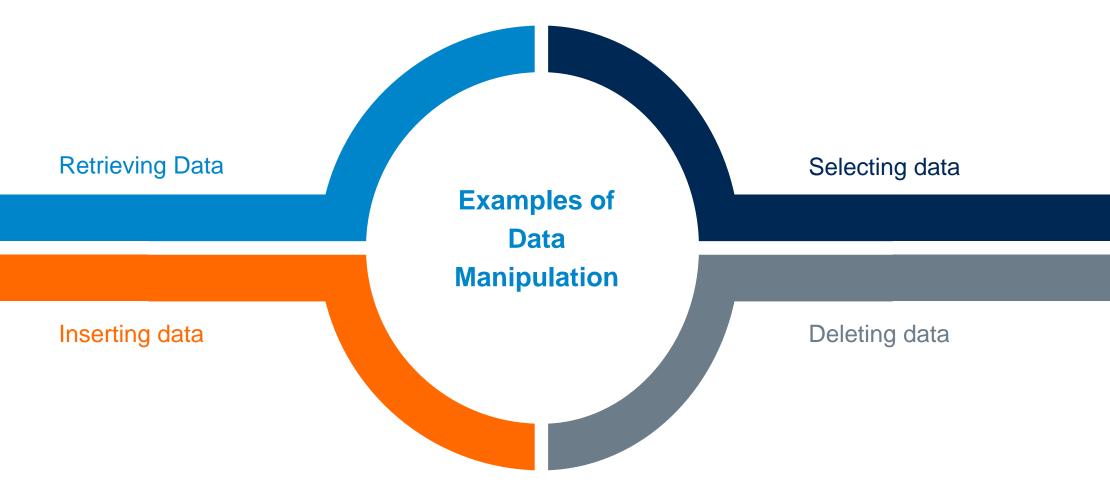
# **Text Manipulation**

String methods

# Ui Path •

# **Data Manipulation**

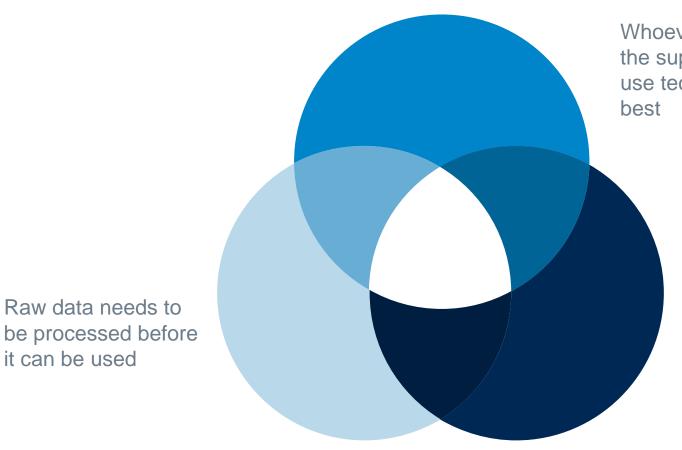
Data Manipulation is defined as the process of altering the original behavior of the data by applying mathematical operations.



it can be used

# Importance of Data Manipulation

Data is defined as collection of information which can be stored and processed.



Whoever has data, has the supreme power to use technology at their

> Data needs to be verified, filtered and concatenated to make it reliable



### **Data Query**

- A read-only operation
- Whenever a query is fired, it only works on a static source and returns values accordingly

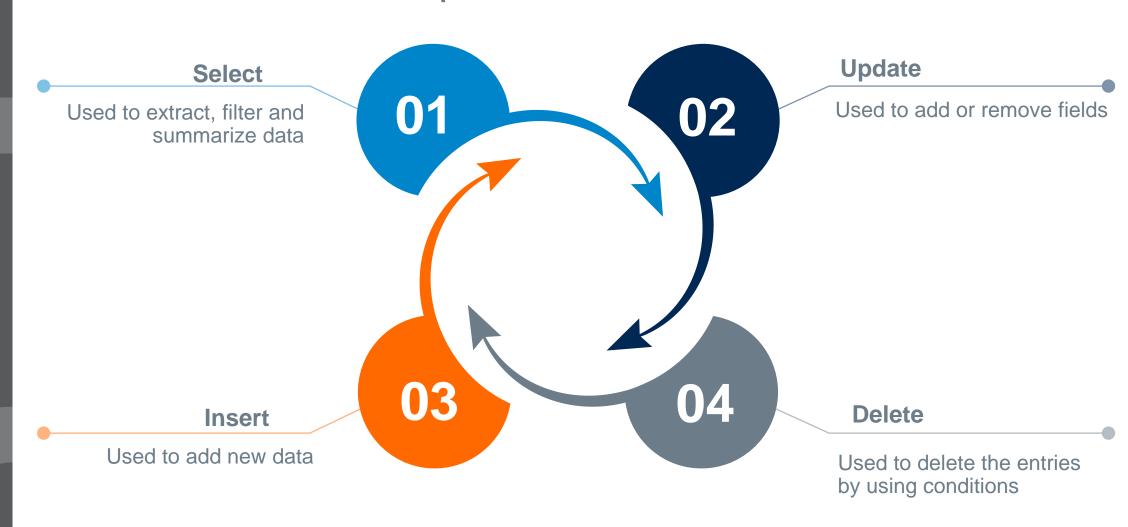
# Data Manipulation

- An operation that alters the data structure
- Operations include inserting, deleting or modifying data



# **Basic Operations of Data Manipulation**

There are four basic data manipulation commands:







# **Basics of Data Manipulation**

- Introduction to Data Manipulation
- Data Query vs. Data Manipulation
- Data Manipulation Basic Operations



# **Data Types**

• Types of variables in UiPath



# **Data Manipulation Operations**

Initializing

Updating

Selecting

Deleting

Inserting



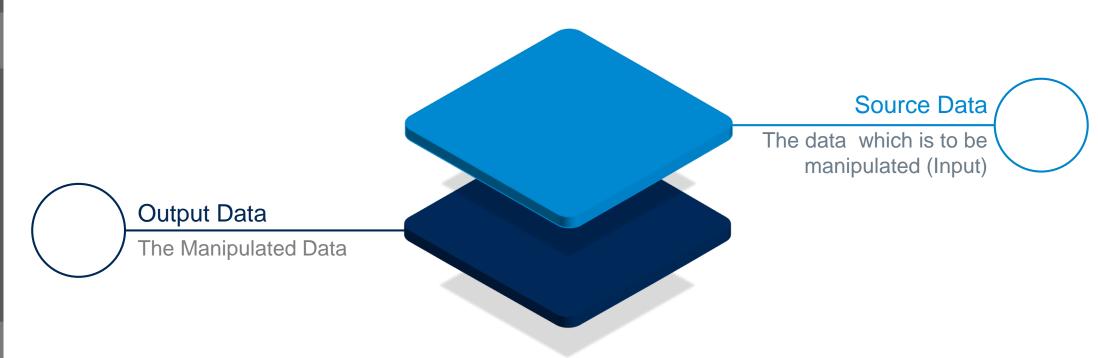
# **Text Manipulation**

String methods

# Ui Path -

# **Data Types**

The types of data determines the type of data operation that can be carried out. Data can be categorized as:



# **Types of Data Storing Variables**

The outcome of the data manipulation depend on the type of variable in which the data is stored. The most important types of variables used in UiPath to store data are:







# **Basics of Data Manipulation**

- Introduction to Data Manipulation
- Data Query vs. Data Manipulation
- Data Manipulation Basic Operations



# **Data Types**

Types of variables in UiPath



### **Data Manipulation Operations**

Initializing

Updating

Selecting

Deleting

Inserting

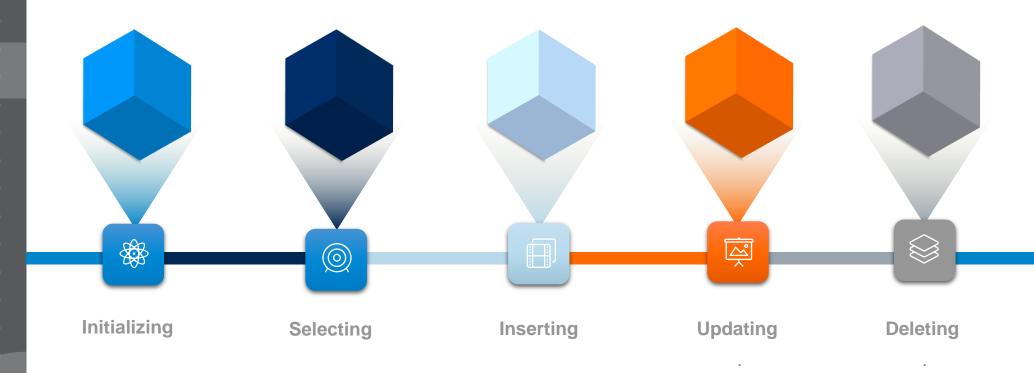


# **Text Manipulation**

String methods

# **Data Manipulation Operations**

There are five types of data manipulation operations in UiPath:



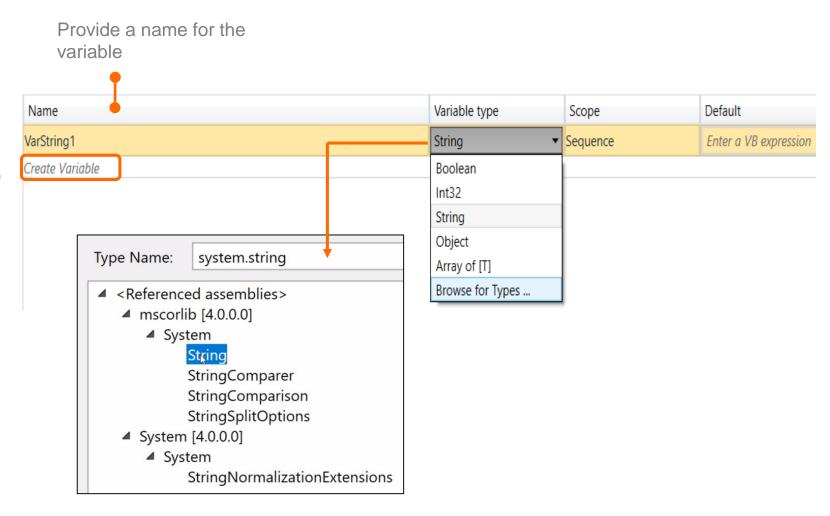


# Variable Initialization

In UiPath, the variable initialization process is known as variable creation. The steps for creating a variable are:

#### Step 1:

- Go to the Variables Panel and Click on "Create variable"
- Choose a name for the variable

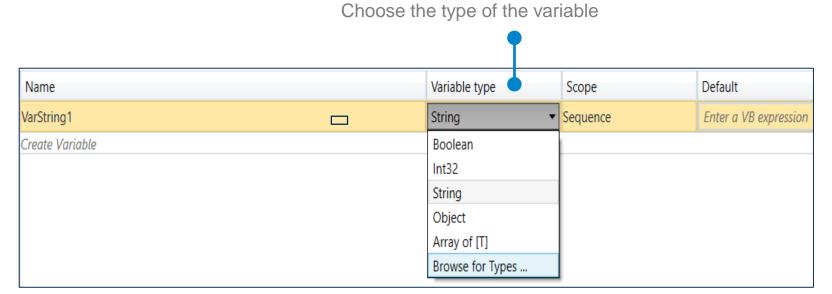




# Variable Initialization (Contd.)

#### Step 2:

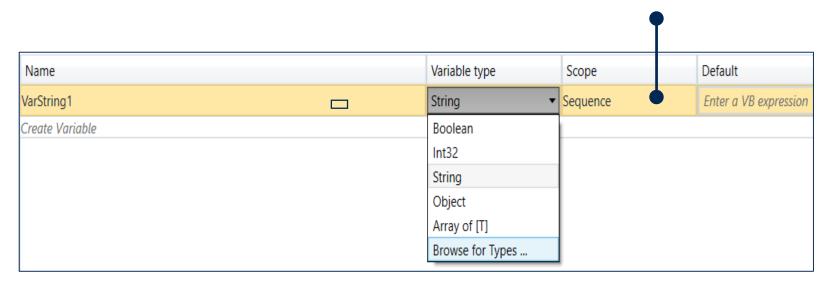
 Choose the type of the variable from the drop-down list



# Variable Initialization (Contd.)

#### Step 3:

 Choose the scope of the variable



Choose the scope of

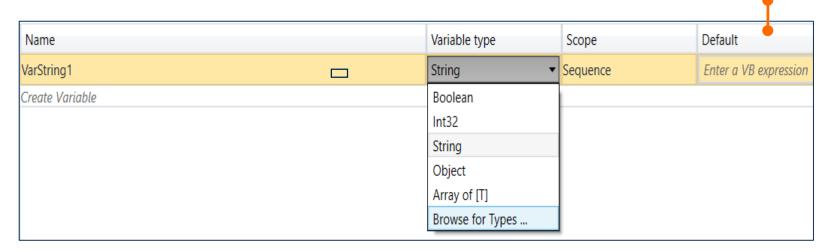
the variable



# Variable Initialization (Contd.)

#### Step 4:

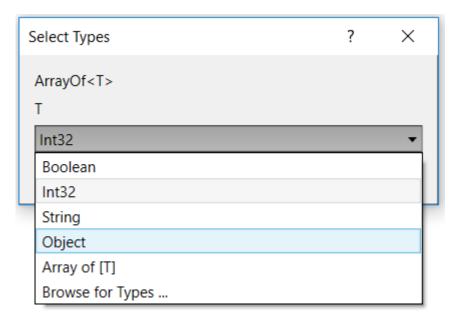
Specify a default value of the variable

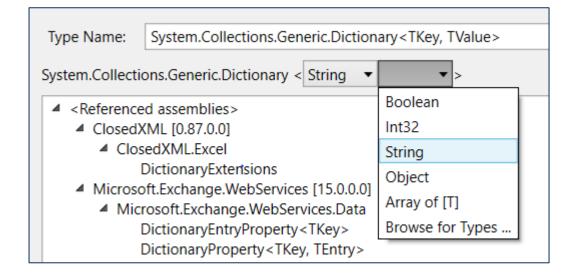


Type a default value (optional)



When we are creating Collection Variables (Arrays, Lists, Dictionaries), the data type of the elements has to be specified.







# **Practice makes Perfect...**





A Calculus 3 class has 20 students: Evalyn Mineau, Altagracia Sentell, Karon Rosamond, Lucas Kowalewski, Tiny Hewlett, Sherlyn Smothers, Carley Chicoine, Charlesetta Carini, Felicia Phillips, Emogene Carrow, Andra Willard, Rachelle Alkire, Carissa Yoshimura, Sheba Holben, Earlean Rames, Kendrick Cornett, Thomasina Mohler, Elly Bottorff, Meaghan Jacquemin, and Craig Brockwell

Create a variable to store the names of the students in the class.

Name	Variable type	Scope	Default
students Array	String[]	Sequence	{"Evalyn Mineau", "Altagracia Sentell", "Karon F
Kowalewsk Chicoine"," "Andra Wil "Earlean Ra		", "Tiny Hewlett Charlesetta Cari ard", "Rachelle mes", "Kendrick	cia Sentell", "Karon Rosamond", "Lucas t", "Sherlyn Smothers", "Carley ini", "Felicia Phillips", "Emogene Carrow", Alkire", "Carissa Yoshimura", "Sheba Holben", to Cornett", "Thomasina Mohler", "Elly Bottorff", raig Brockwell"}

# **Practice** makes Perfect....





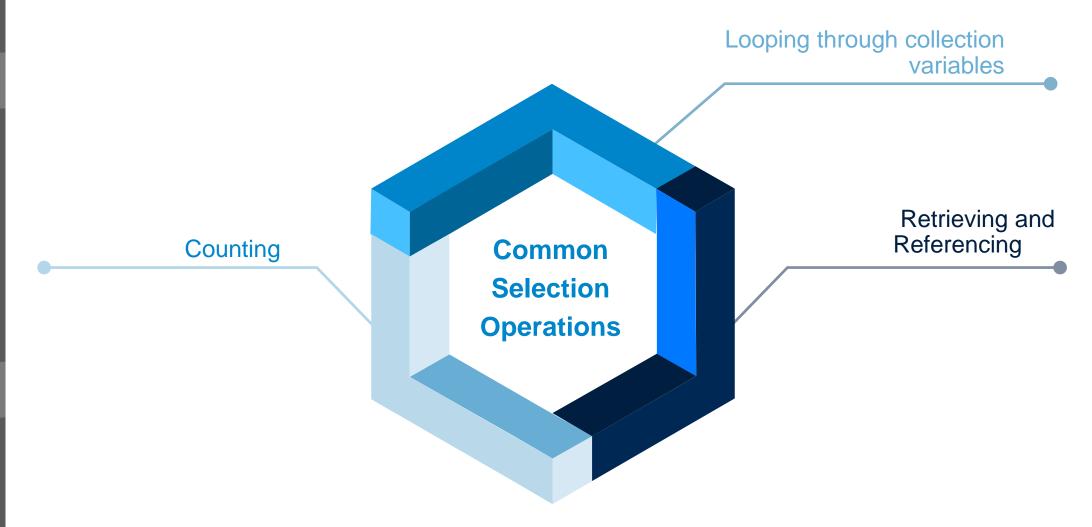
The game character is in the possession of many items. In their wallet they have 500 gold pieces, their pouch has flint, twine, and gemstone, and finally their backpack contains a xylophone, a dagger, a bedroll and a bread loaf.

Create a variable to store the items the character has, mapped to the place where they are located.

Name		Variable type	Scope	Default
Location to Inventory Dictionary		Dictionary < String, String[] >	Sequence	New Dictionary
Create Variable  New Dictionary(Of String, String()) From {{"wallet", New String() {"500 gold pieces"}}, {"pouch", New String() {"flint", "twine", "gemstone"}},  {"backpack", New String() {"xylophone", "dagger", "bedroll", "bread loaf"}}}				

# **Data Selection**

Data selection covers operations that retrieve data from existing data structures without making any changes to the source data.



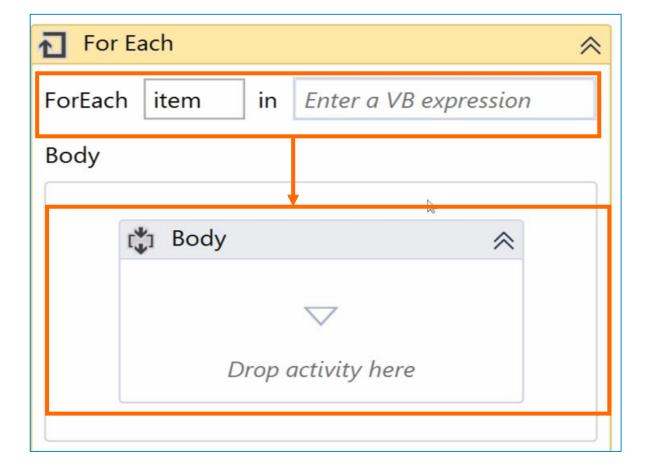


# **Variable Selection**

Steps for looping through arrays:

#### Step 1:

Drag a 'For Each' activity into the workflow



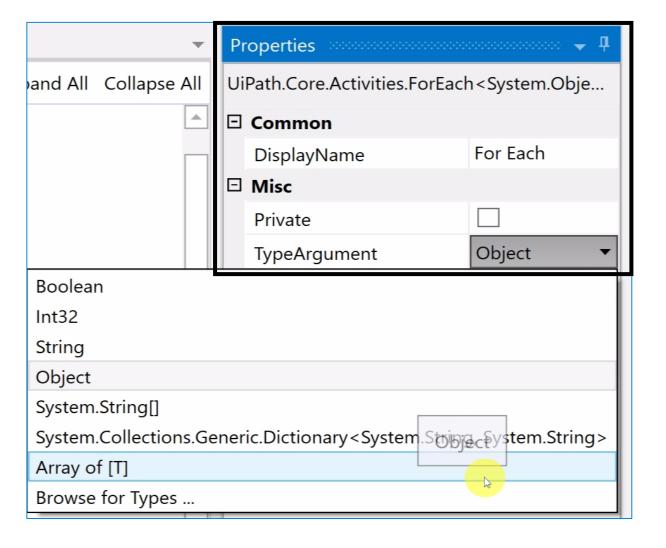


# Variable Selection (Contd.)

Steps for looping through arrays:

#### Step 2:

- Go to the properties panel of the activity
- Change the 'TypeArgument' to the type of the variable you wish to iterate through



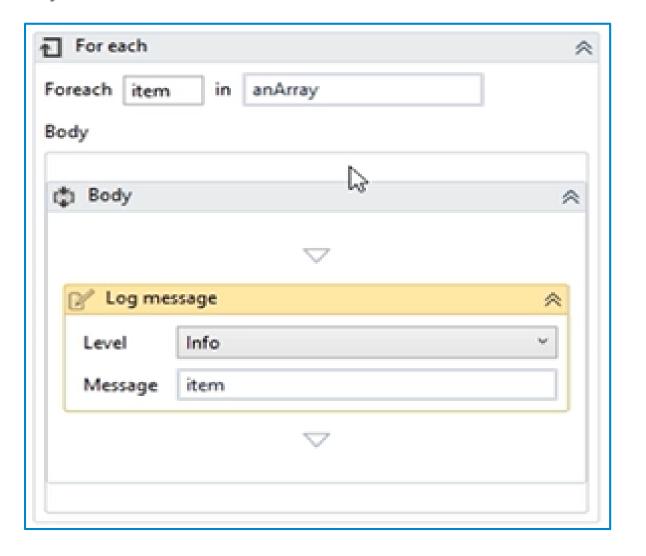
# Ui Path •

# Variable Selection (Contd.)

Steps for looping through arrays:

#### Step 3:

Loop through the collection by inserting its name in the 'For Each' activity to display items



# **Practice** makes Perfect....





Let's go back to the Array with the name of the Calculus 3 students: Evalyn Mineau, Altagracia Sentell, Karon Rosamond, Lucas Kowalewski, Tiny Hewlett, Sherlyn Smothers, Carley Chicoine, Charlesetta Carini, Felicia Phillips, Emogene Carrow, Andra Willard, Rachelle Alkire, Carissa Yoshimura, Sheba Holben, Earlean Rames, Kendrick Cornett, Thomasina Mohler, Elly Bottorff, Meaghan Jacquemin, and Craig Brockwell

For Each

For Each

item in students Array

Body

Write Line

Text item. To Upper

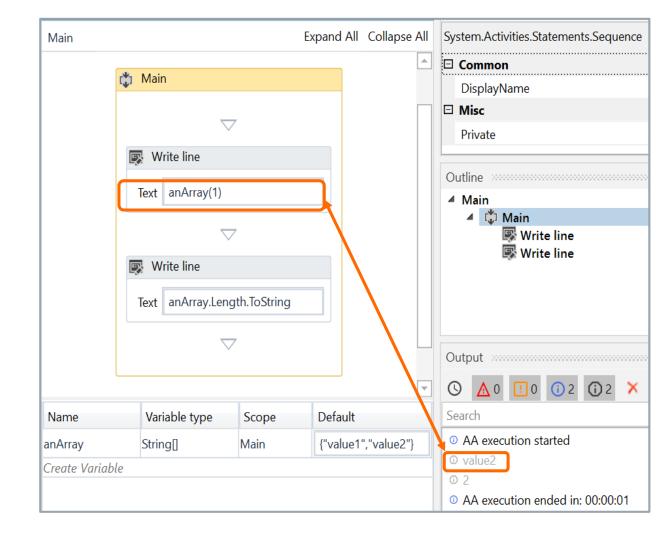
Create a simple workflow to loop through it.

BlankProcess execution started
EVALYN MINEAU
ALTAGRACIA SENTELL
KARON ROSAMOND
LUCAS KOWALEWSKI
TINY HEWLETT
SHERLYN SMOTHERS
CARLEY CHICOINE
CHARLESETTA CARINI
FELICIA PHILLIPS
EMOGENE CARROW
ANDRA WILLARD
RACHELLE ALKIRE
CARISSA YOSHIMURA
SHEBA HOLBEN
EARLEAN RAMES
KENDRICK CORNETT
THOMASINA MOHLER
ELLY BOTTORFF
MEAGHAN JACQUEMIN
CRAIG BROCKWELL
BlankProcess execution ended in: 00:00:00



# Referencing an Item

Referencing an item in a collection variable by its index
VariableName(index of item as integer).toString



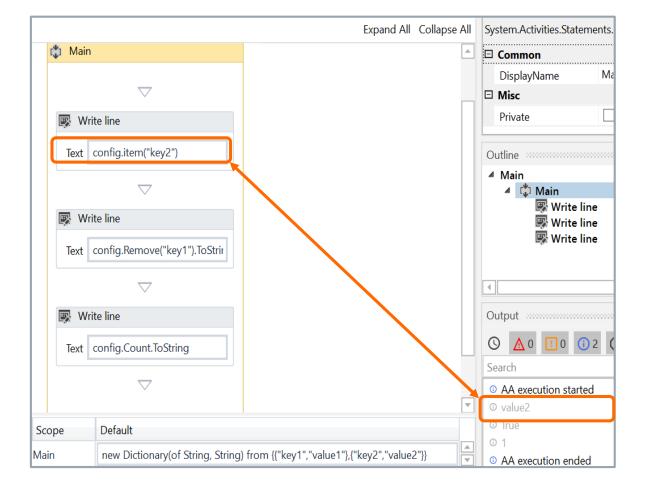


# Retrieving an Item

An item from a Dictionary variable can be retrieved by using the Item method.

Retrieve a value using the item method [DictionaryName].item("[key]")

In our example: config.ltem("key2")



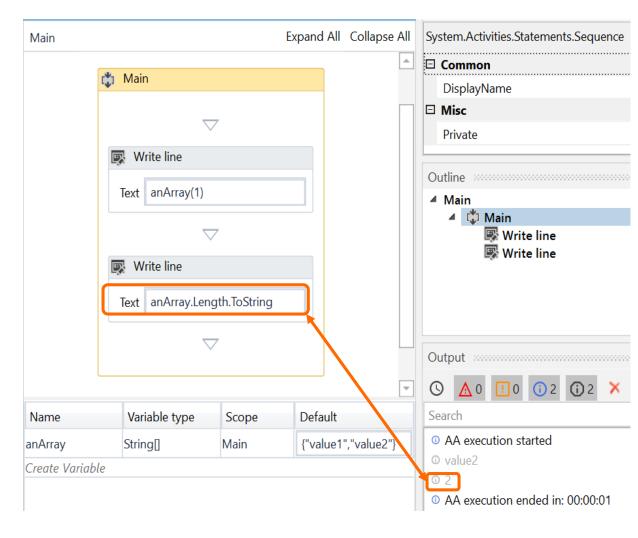


# **Counting an Item**

We can use an activity to count the elements in a collection variable and display the outcome (Length).

Displaying the length of a collection variable through the command 'Length'

ReadOnly Property Array.Length As Integer





We can use the assign activity to insert data in the workflow area.

Name	Variable type	Scope	Default
config	Dictionary <string,string></string,string>	Main	new Dictionary(of String, String) from {{"key1", "value1"}, {"key2", "value2"}}
A			

Use an Assign activity to add a **Key/Value** pair to an existing dictionary. In the To section insert **[DictionaryName]("[Key]")** and insert the value you wish to be stored in the opposite entry.





# **Practice** makes Perfect....





Let's go back to a previous example, in which the game character is in the possession of many items. In their wallet they have 500 gold pieces, their pouch has flint, twine, and gemstone, and finally their backpack contains a xylophone, a dagger, a bedroll and a bread loaf.

Add a pocket location to the inventory of the game character. This pocket contains a seashell, a strange berry, keys and lint

LocationtoInventor = new String(){"seash

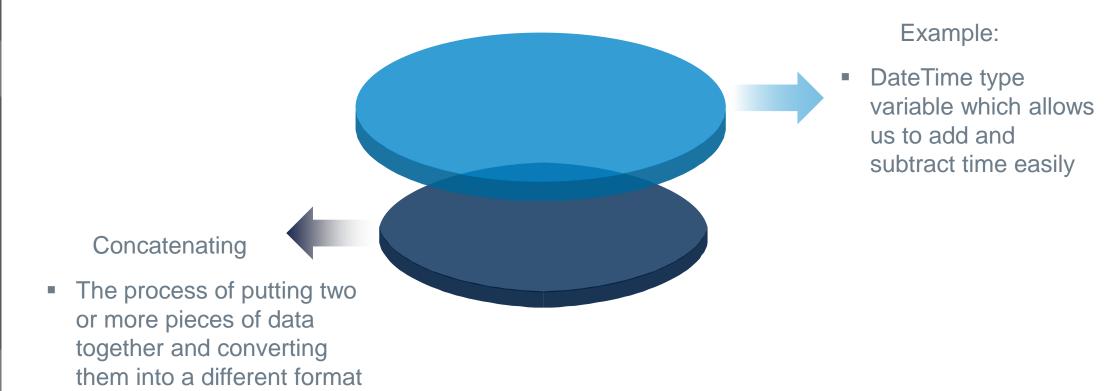
LocationtoInventoryDictionary("pocket")

new String(){"seashell","strange berry", "keys","lint"}

# Ui Path -

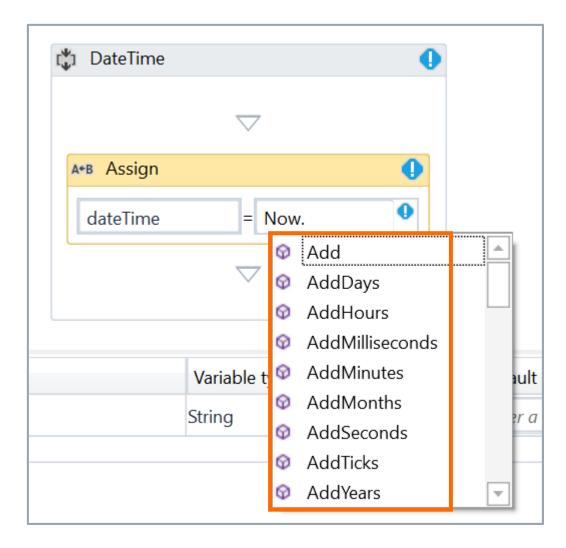
# **Updating**

Data updating is a simple task in which we perform certain operations to modify the current data.



# Manipulating DateTime Variables

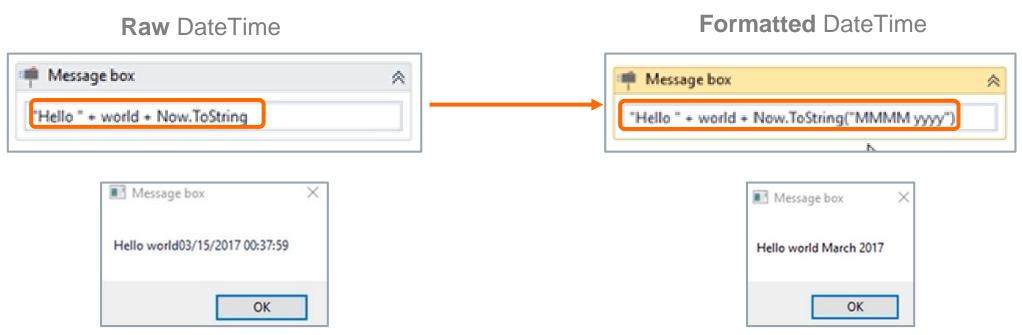
The built-in methods in UiPath can be used to add any period of time to the given value of the DateTime type variable, when defining it.



# Ui Path •

# **Concatenating and Converting**

In UiPath, the "+" symbol is used to concatenate same type of data. To concatenate different data types, they are first converted to string type using the "ToString" method.





# **Formatting DateTime**

Filter by title

Base Types

Common Type System

- > Type Conversion in .NET
- → Formatting Types

Standard Numeric Format Strings

Custom Numeric Format Strings

Standard Date and Time Format Strings

#### **Custom Date and Time Format Strings**

Standard TimeSpan Format Strings

Custom TimeSpan Format Strings

**Enumeration Format Strings** 

Composite Formatting

- > Performing Formatting Operations
- > Manipulating Strings
- > Parsing Strings

The following table describes the custom date and time format specifiers and displays a result string produced by each format specifier. By default, result strings reflect the formatting conventions of the en-US culture. If a particular format specifier produces a localized result string, the example also notes the culture to which the result string applies. See the Notes section for additional information about using custom date and time format strings.

Format specifier	Description	Examples
"d"	The day of the month, from 1 through 31.	2009-06-01T13:45:30 -> 1
	More information: The "d" Custom Format Specifier.	2009-06-15T13:45:30 -> 15
"dd"	The day of the month, from 01 through 31.	2009-06-01T13:45:30 -> 01
	More information: The "dd" Custom Format Specifier.	2009-06-15T13:45:30 -> 15
"ddd"	The abbreviated name of the day of the week.	2009-06-15T13:45:30 -> Mon (en-US)
	More information: The "ddd" Custom Format Specifier.	2009-06-15T13:45:30 -> Пн (ru-RU)
		2009-06-15T13:45:30 -> lun. (fr-FR)
"dddd"	The full name of the day of the week.	2009-06-15T13:45:30 -> Monday (en-US)
	More information: The "dddd" Custom Format Specifier.	2009-06-15T13:45:30 -> понедельник (ru-RU)
		2009-06-15T13:45:30 -> lundi (fr-FR)

**Source**: https://docs.microsoft.com/en-us/dotnet/standard/base-types/custom-date-and-time-format-strings

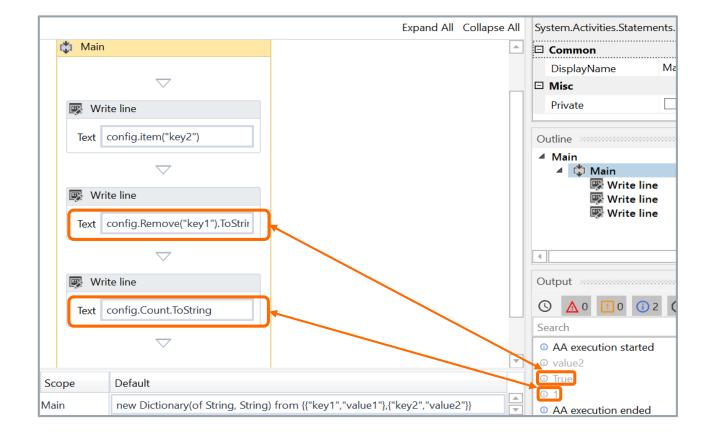


# **Deleting**

Deletion refers to a complex process of selection and deletion of data.

Function Dictionary(Of String, String).Remove(key As String) As Boolean

Removing a Key/Value pair







# **Basics of Data Manipulation**

- Introduction to Data Manipulation
- · Data Query vs. Data Manipulation
- Data Manipulation Basic Operations



# **Data Types**

Types of variables in UiPath



# **Data Manipulation Operations**

Initializing

Updating

Selecting

Deleting

Inserting



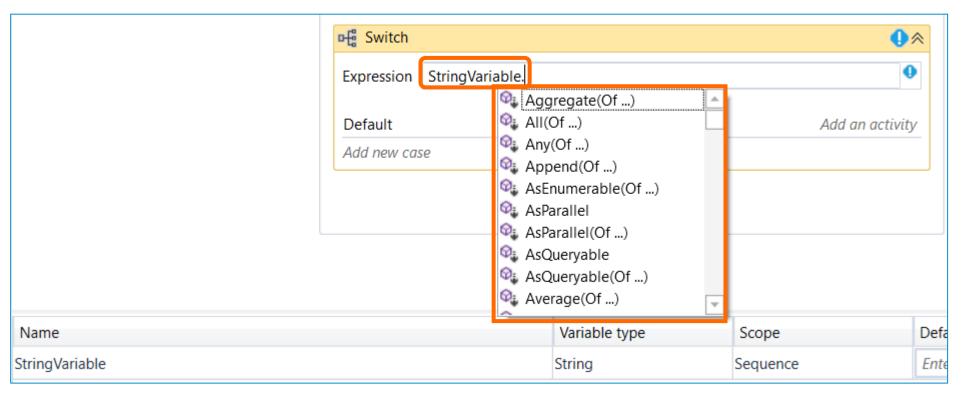
# **Text Manipulation**

String methods

### **Text Manipulation**

Text manipulation refers to the operations performed on text. Since text is stored in string variables, text manipulation operations are known as String Methods.

In UiPath, all the string methods are displayed as a drop-down list after typing the name of a string variable, followed by a dot character.



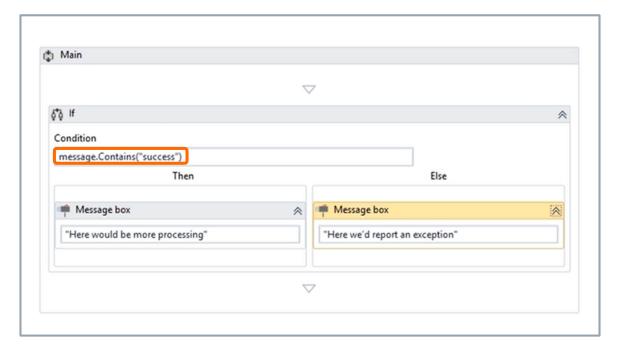
# **Main String Methods**

Method	Syntax				
Contains	Function String.Contains(value As String) As Boolean				
EndsWith	Function String.StartsWith(value As String, ignoreCase As Boolean, culture As CultureInfo) As Boolean				
StartsWith	Function String.EndsWith(value As String, ignoreCase As Boolean, culture As CultureInfo) As Boolean				
Format	Function String.Format(provider As IFormatProvider, format As String, arg0 As Object, arg1 As Object, arg2 As Object) As String				
Replace	Function String.Replace(oldChar As Char, newChar As Char) As String				
Split	Function String.Split(separator As Char(), count As Integer, options As StringSplitOptions) As String()				
Substring	Function String.Substring(startIndex As Integer, length As Integer) As String				
ToLower	Function String.ToLower(culture As CultureInfo) As String				
ToUpper	Function String.ToUpper(culture As CultureInfo) As String				
Trim	Function String.Trim(ParamArray trimChars As Char()) As String				



### **String.Contains**

The String. Contains method is used to verify if a string contains a particular string of characters.



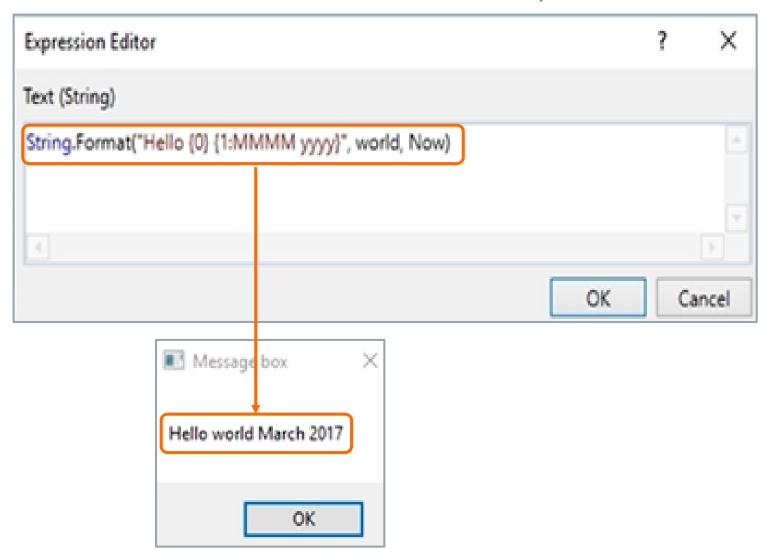
It returns a Boolean value.



# Ui Path •

### **String.Format**

The String. Format method is used to convert an entire expression into a string.





### String.Replace

The String.Replace method is used to identify a sequence of characters (string) in a text and replace it with a given string.

**Syntax** 



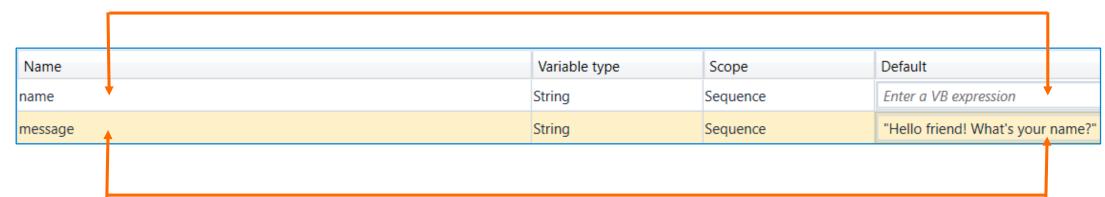
Function String.Replace(oldValue As String, newValue As String) As String



### String.Replace Example

A sequence replaces a part of the greeting with the name that the user inputs.

The variable "name" stores the name that the user inputs.

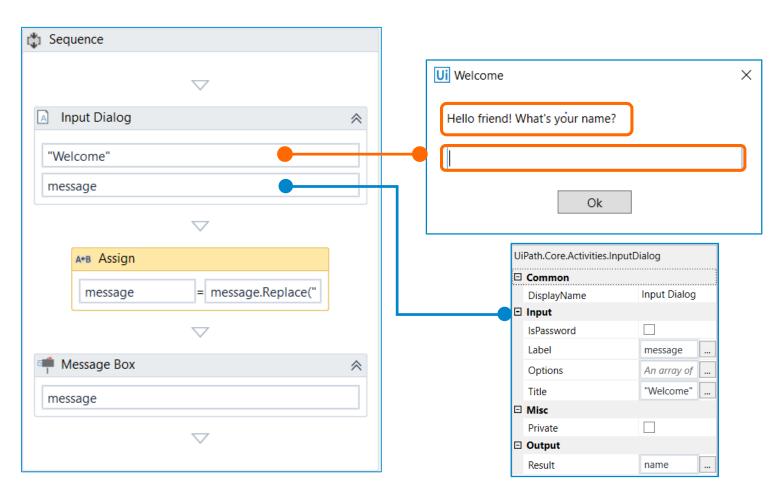


The variable "message" has an initial value: "Hello friend! What's your name?"

### String.Replace Example (Contd.)

### Step 1:

- Greet the user with the initial value of the "message" variable.
- Get the name that the user inputs.

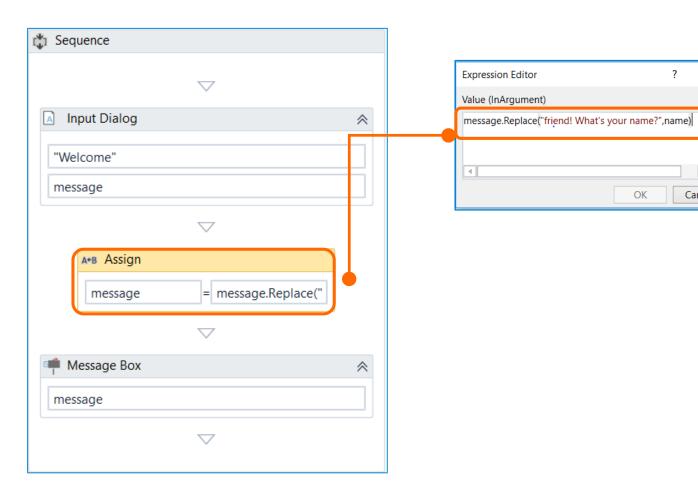




## String.Replace Example (Contd.)

### Step 2:

- Perform the "String.Replace" method inside an Assign activity.
- Use the expression: "message.Replace ("friend! What's your name?" name)."



 $\times$ 

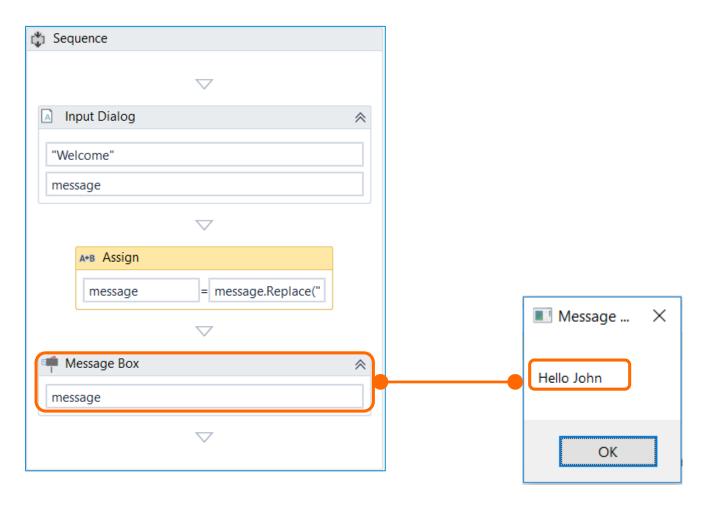
Cancel

OK

### String.Replace Example (Contd.)

### Step 3:

 Show the new value of the message variable ("Hello" + the name that the user has input.





## String.Split

String. Split is a command that is used to separate strings based on certain criteria. The below example shows how to extract and view an automatically generated record number.

The variable "status" stores the status.

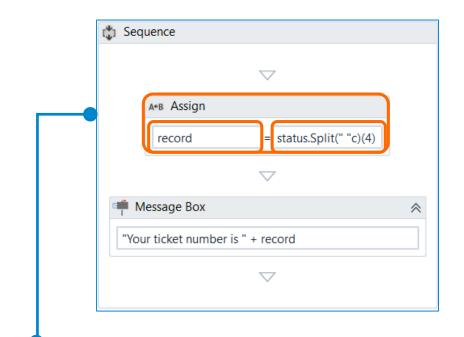
Name	Variable type	Scope	Default	
status	String	Sequence	"Operation completed successfully. Record	d 1234 has
record	String	Sequence	Enter a VB expression	

The variable "record" assigns the record number.

# String.Split (Contd.)

### Step 1:

- Use an Assign activity with a "String.Split" method.
- Use space ("") as a separator.
- Isolate the substring "Record".
- Use the expression status.split (" "C)(4)



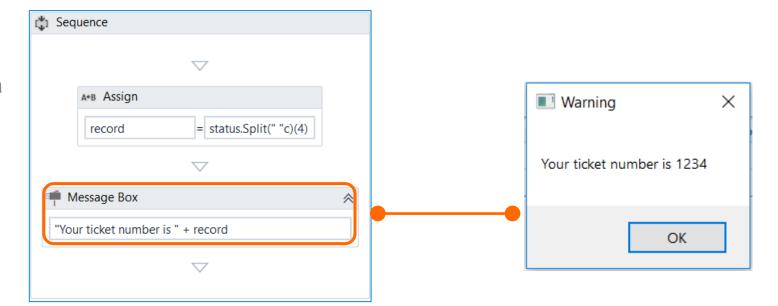




### String.Split (Contd.)

### Step 2:

Display the result in a message box.





### String.Trim and String.Substring

String. Trim is used to remove a part of a string.

**Syntax** 

Function String.Trim(ParamArray trimChars As Char()) As String

String. Substring is used to isolate a part of a string for further use.

**Syntax** 

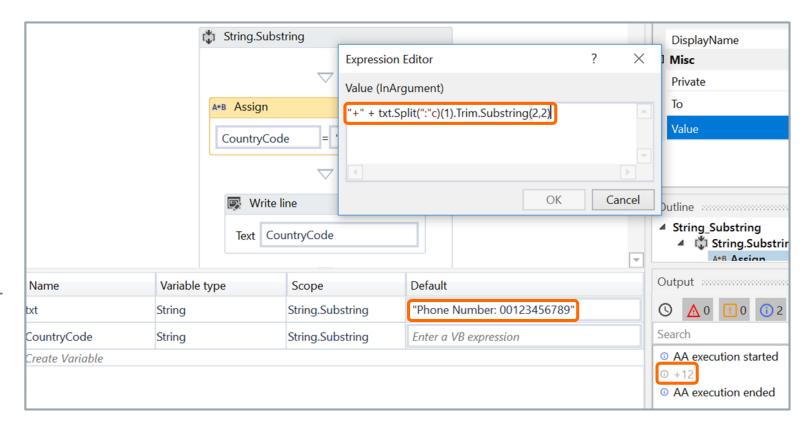
Function String.Substring(startIndex As Integer, length As Integer) As String

# String.Trim and String.Substring Example

In order to find out the location from where a person called based on their phone number kept in string variable "Phone Number 00123456789"

### Steps:

- 1. Use "String.Trim" to remove everything before ":".
- 2. Use "txt.Split(":"c)(1).Trim" to split the part before the ":".
- 3. Use (Substring(2,2):) to isolate the 2 figures after the first 2 figures.

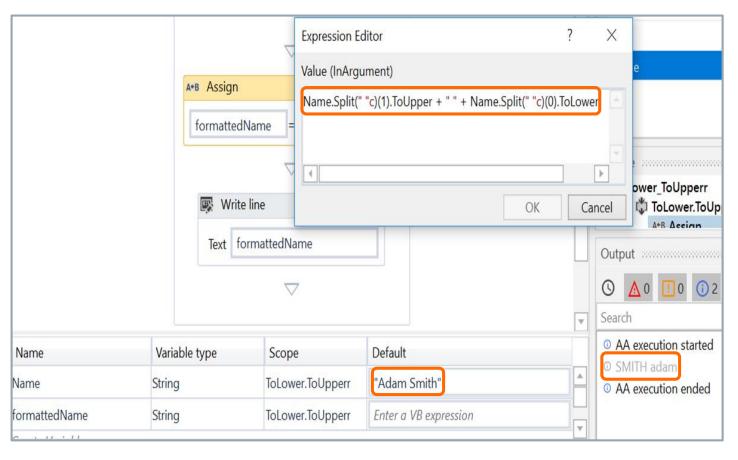




### String.ToLower and String.ToUpper

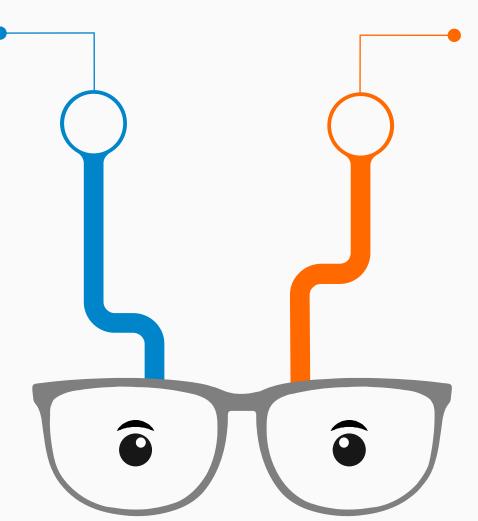
String.ToLower is used for converting a string to lowercase, while String.ToUpper converts a string to uppercase.

- Use "Name.Split(" "c)(1).ToUpper + " " + Name.Split("
   "c)(0).ToLower" to convert a string to lowercase, and uppercase respectively.
- "Name.Split(" "c)(1).ToUpper" splits the "Name" string and converts the part after space to upper case.
- "Name.Split(" "c)(0).ToLower" splits the Name string and converts the part before the space to lower case.



### **Basics of Data Manipulation**

Data Manipulation is defined as the process of altering the original behavior of the data by applying mathematical operations.



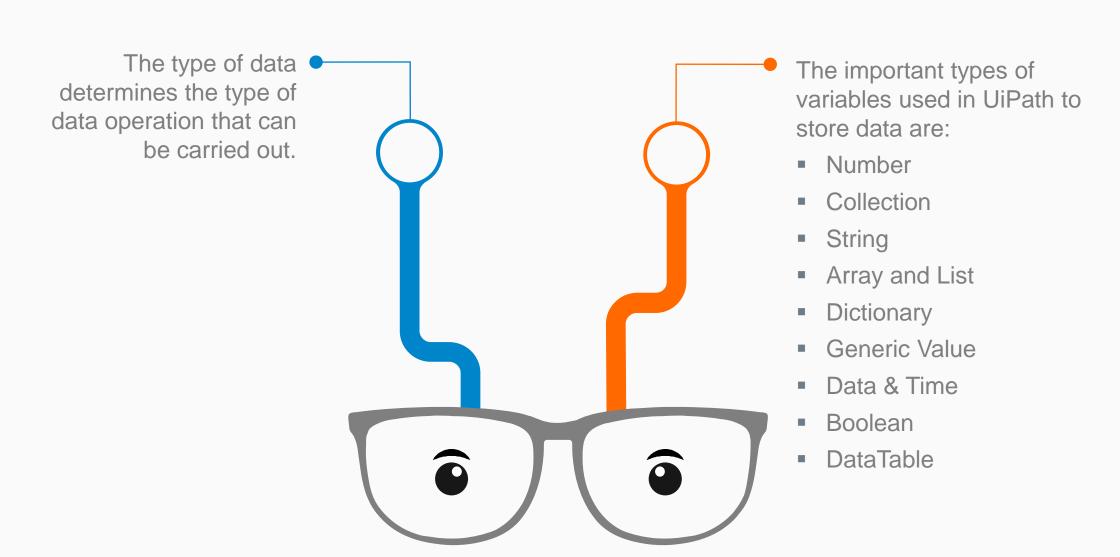


There are four basic data manipulation commands:

- Select
- Update
- Insert
- Delete

# **Ui** Path<sup>™</sup>

### **Data Types**



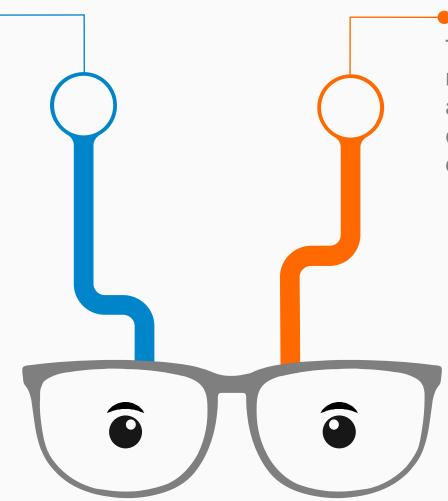
# **Ui** Path<sup>™</sup>

### **Data Manipulation Operations**

There are five types of data manipulation operations in UiPath.

The data manipulation operations are:

- Initializing
  - Selecting
  - Inserting
  - Updating
  - Deleting

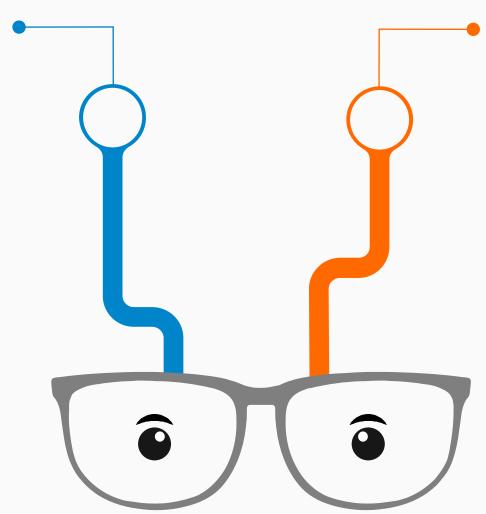


The importance of data manipulation cannot be ruled as it helps in achieving the end output by making the desired changes

# **Ui** Path<sup>™</sup>

### **Text Manipulation**

Text manipulation refers to the operations performed on text. They are also called string methods.



The main string methods are:

- String.Contains
- String.Format
- String.Replace
- String.Split
- String.Trim
- String.ToLower
- String.ToUpper

### **Questions & Answers**





# How can we display today's date in the following format: March, 07 of 2019?

a) Now.ToString("mm, dd 'of' yyyy")

Ui Path

b) Now.ToString("MMMM, dd 'of' yyyy")

c) Now.ToString("mmmm, dd 'of' yy")



### Spot the error in the following:

String.Format("First Name: (1) & Last Name: (2)","Adam","Smith")

Intended Output: "First Name: Adam & Last Name: Smith"

a) Use {} instead of ().

b) The indices start at zero and not at one.

c) Both

# What is the best collection type to store a group of variables with their names?

a) Dictionary

b) String

c) List



In the UiPath Studio, the shortcut "Ctrl + K" is used to automatically create a variable with the appropriate data-type for the property used with.

a) True

b) False



# Write two different ways to extract the State and Zip Code "TX 75248" from the following addresses. Each address is a separate string of name "Addr".

- 3159 Charla LaneDallas, TX 75248
- 3042 Desert Broom CourtWayne, NJ 07477
- 2159 Hewes AvenueBaltimore, MD 21201
- 4994 Corpening DriveTroy, MI 48084

- a) Addr.Split(","c)(1).Trim.
- b) Regex.Match(Addr,"[A-Z]{2} \d{5}").
- c) Both

# **Next Steps**



Module 3 - Lesson 5: Recording & Advanced UI Interaction

