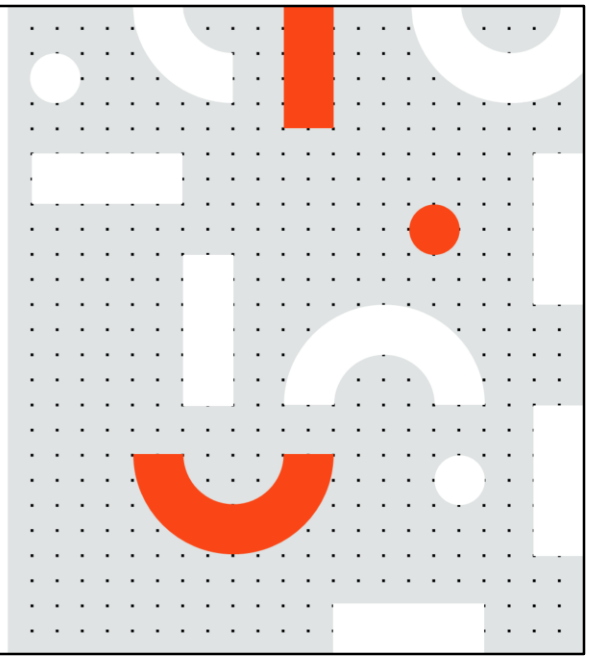
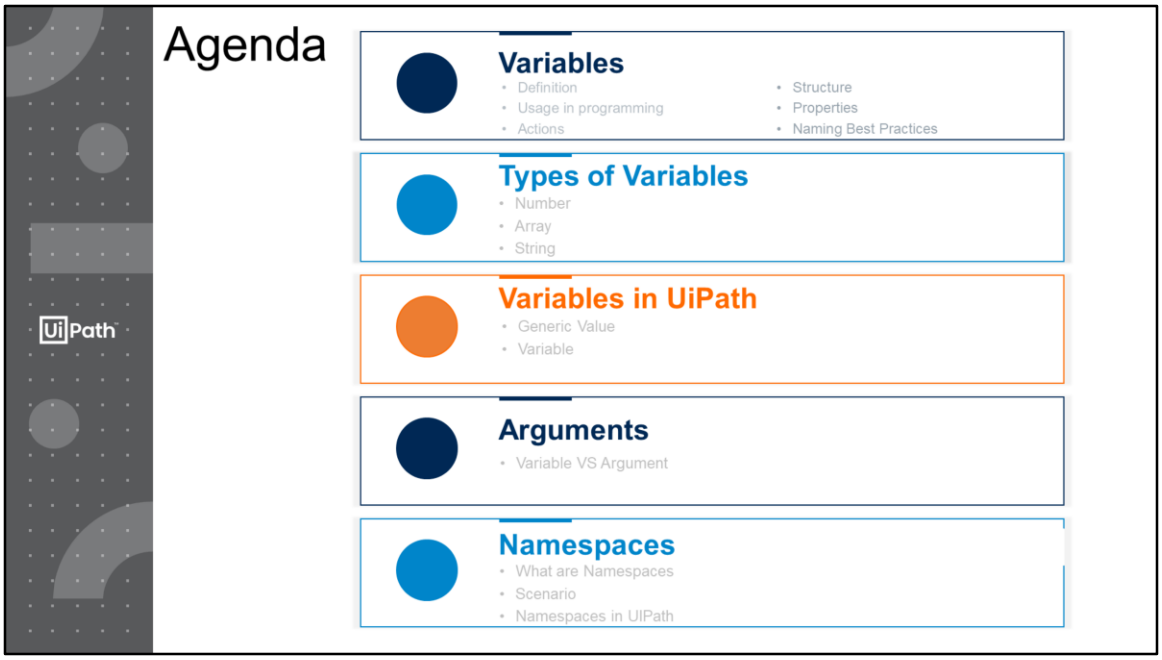
RPA Design & Development v1.1

Lesson 6 Variables



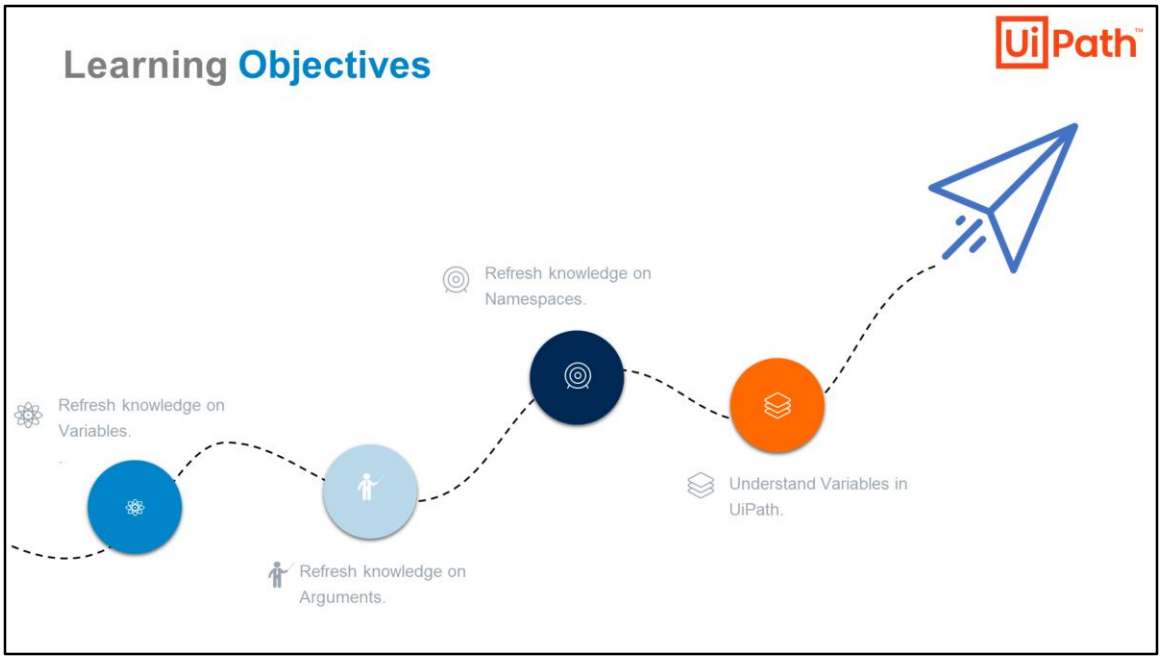
Welcome!
In the last lesson we learnt the basics of UiPath.
In this lesson, we will learn about Variables and how they can be used in UiPath to automate processes.



In this lesson, we will cover the following topics:

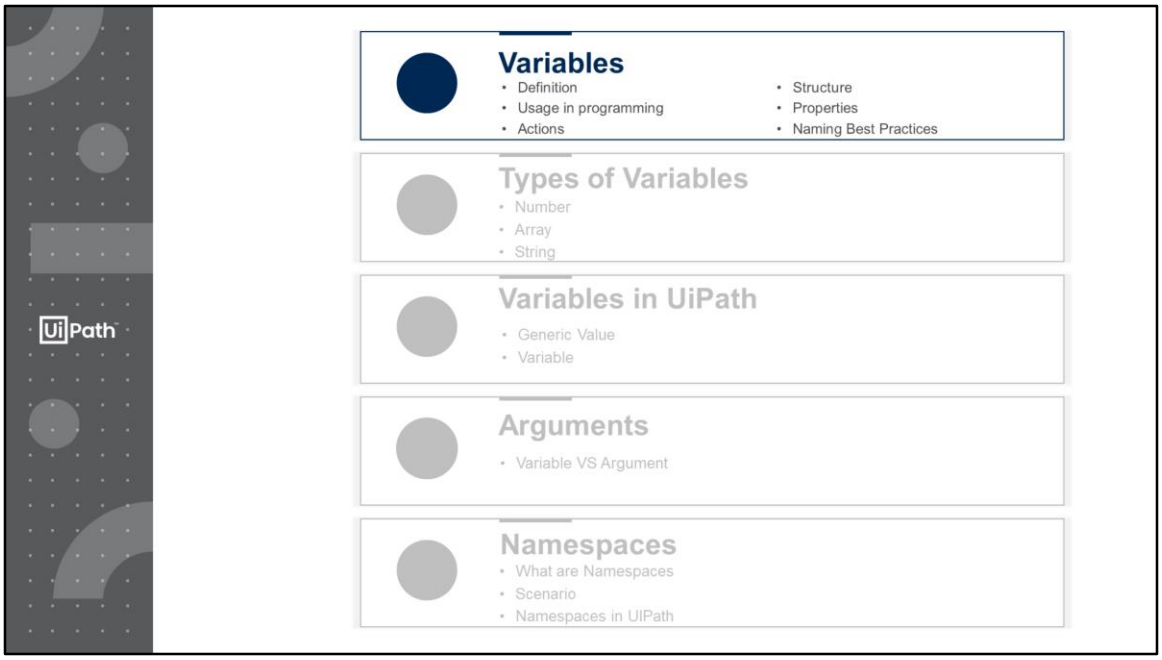
- Variables
  - Definition
  - Usage in programming
  - Actions
  - Structure
  - Properties
  - Naming best practices
- Types of Variables
  - Number Array
  - String
  - Date & Time
  - Boolean
  - DataTable
- Variables in UiPath
  - Generic Value
  - Variable
- Arguments
  - Variable VS Argument

- Namespaces
    - What are namespace
    - Scenario
    - Namespace in UiPath

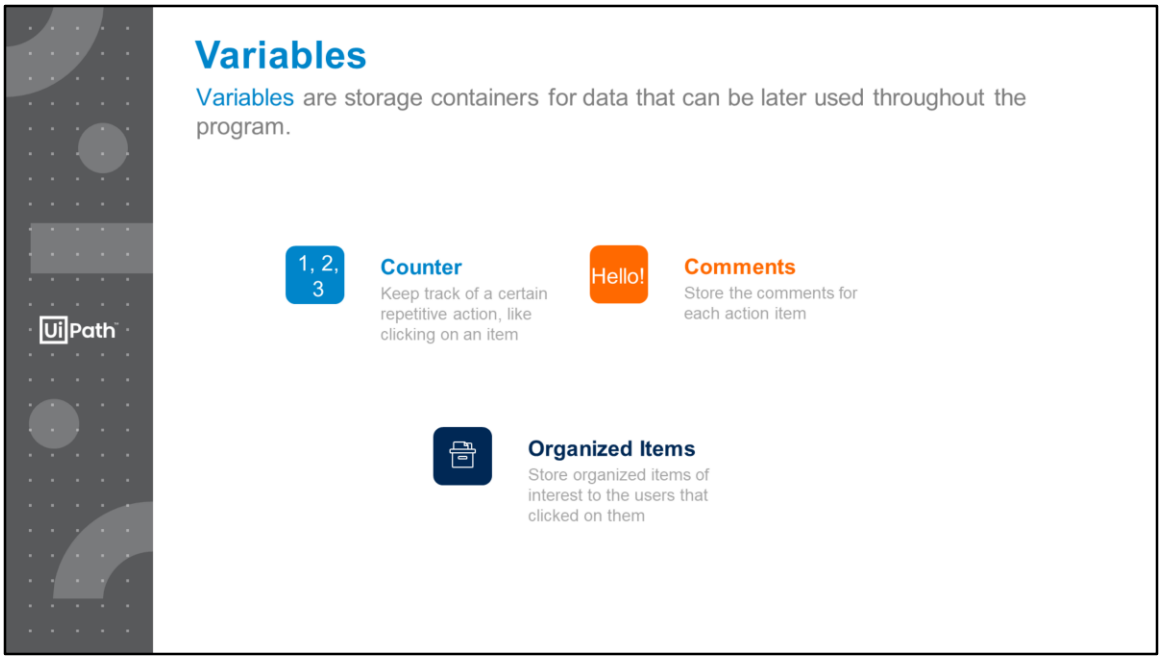


At the end of this lesson, you will be able to:

- Define Variables and understand their usage in UiPath
- Define Arguments and understand their usage in UiPath
- Define Namespaces and understand their usage in UiPath
- Understand the variable and implement in Uipath studio.



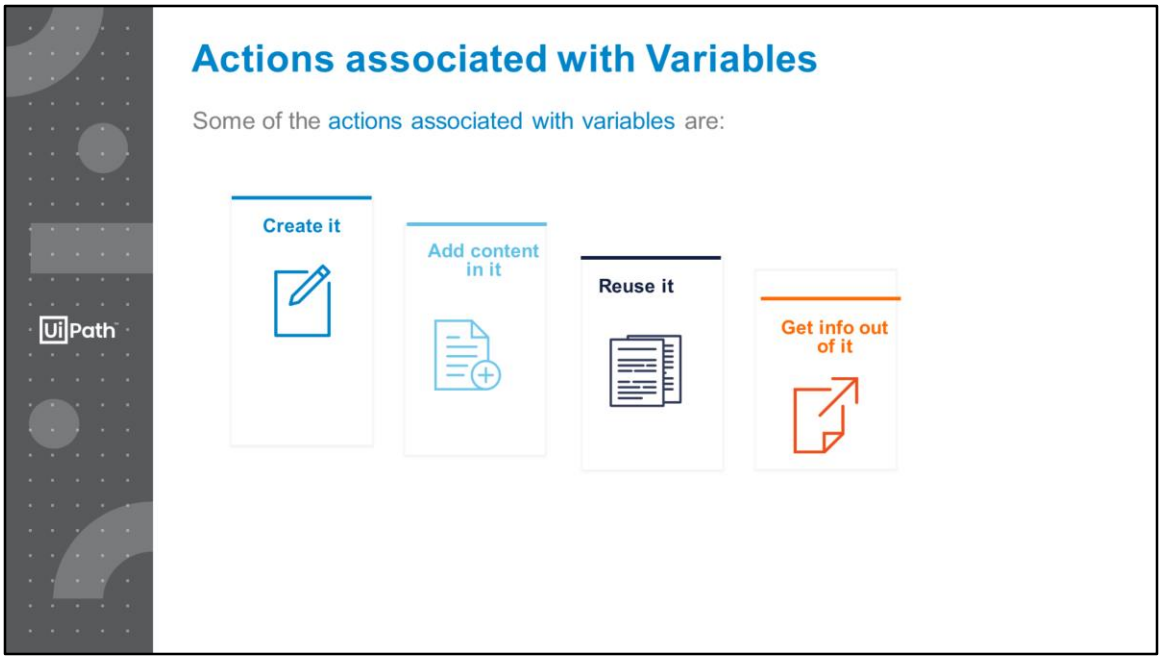
Let’s start by refreshing our knowledge on variables.



Variables are containers that store different types of information. Using variables in computer programs makes it easier to label and store data which can later be used throughout the program.

It's useful to think of variables like boxes that keep different types of data. The data in the box might change, but the box remains the same.

- In one box, you might keep a counter that tracks the number of time users clicked on an item.
- In another box, you could store 'comments (text)' written by the user(s) on that item.
- In another box, you could store a table of items that may be of interest to the user based on the item they click.



Some of the actions associated with variables are:

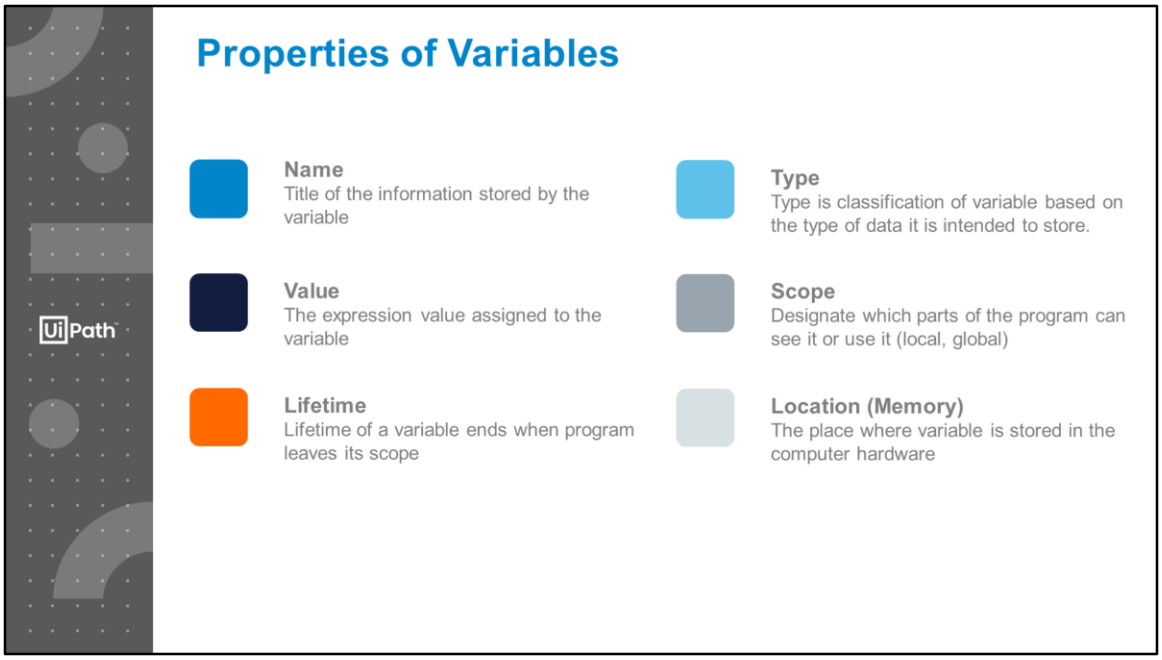
1. **Create it**. The first step is to declare (create) a variable.
   Example (Javascript): var first_name;

2. **Add content in it**. The second step is to store information in it.
   Example: first_name=Helen

3. **Reuse it**. The third step is to reuse the information in a new variable. Depending on what we want to achieve with our code, we can copy information from one variable to another.
   Example: complete_name=first_name+last_name

4. **Get info out of it**. Let us take an example of a variable that stores a large piece of content such as a paragraph. You can apply the required methods in your code to retrieve only a specific piece of content from that entire variable.
   Example:
   text="This is some text I wrote down."

"text.Substring(0,4) = "This" Substring() method extracts the characters from a string, between two specified indices ("start", in our case '0' and "end" in our case '4'), and returns the new sub string.



The variables have following properties:

Name: The name is symbolic. It represents the "title" of the information that is being stored by the variable. A variable should be named to represent all possible values that it might contain. The name is essential in programming, because this is how we "access" the variable. Every variable must have a unique name.

Type: Type is classification of variable based on the type of data it is intended to store. In C, Java, ActionScript, the type of a variable must be explicitly declared when the name is created. The most common (universal) types of a variable are: character, number (integer, floating), array, string, date/time, Boolean and data tables.

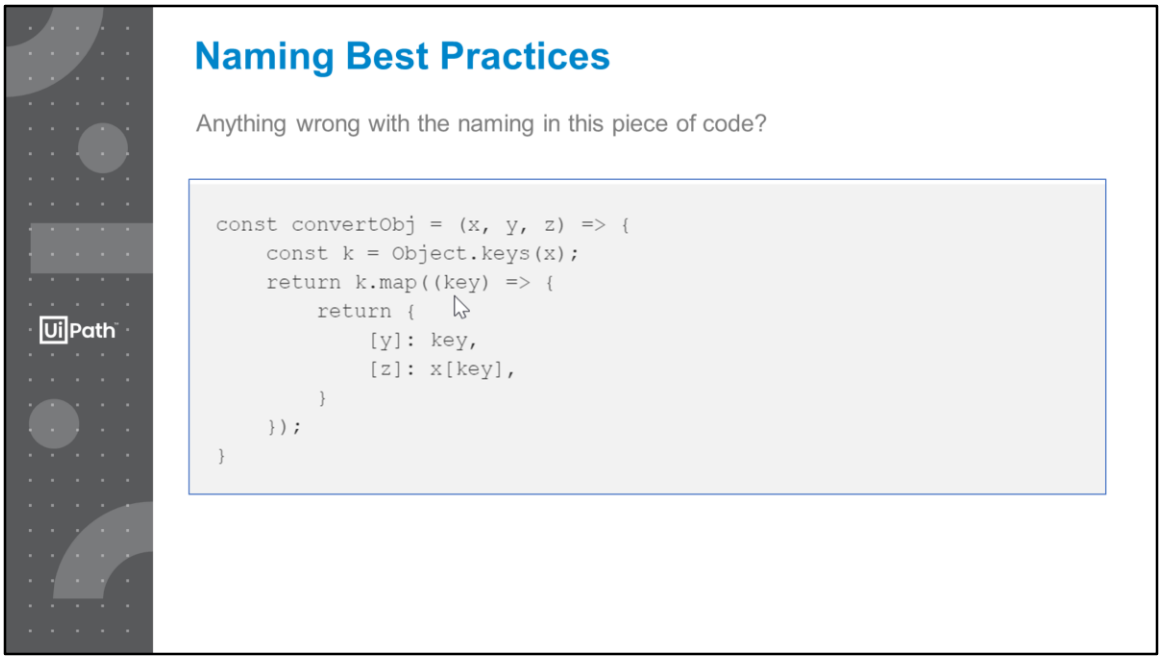
Value: A variable changes over time. Thus if the variable is jims_age and is assigned the value 21. At another point, jims_age may be assigned the value 27. When we "create a variable" we are primarily defining the variables name and type. The best programming practice is to provide an initial value to be associated

with that variable name. If you forget to assign an initial value, then various rules "kick in" depending on the language.

Scope: When it is defined the first time then each variable has a global scope where every part of the program you can access it. We fix the limit of the variable for a single function so that changes inside the function won't affect the main program in unexpected ways. In this, you can see the local scope process inside the sequence where the outside process can not process it. This process is called the Local scope in the Uipath studio**.**

Lifetime: The lifetime of a variable is strongly related to the scope of the variable. Variables "come to life" when the program reaches the line of code where they are "declared". Variables "die" when the program leaves the "Scope" of the variable.
In our initial programs (which do not have functions), the lifetime and scope of the variables will be the "entire program".

Location: We don't have to worry too much about where the variable is stored in the computer hardware. The computer (Matlab/C/Actionscript, etc., via the compiler) does this for us.
But you should be aware that a "Bucket" or "Envelope" exists in the hardware for every variable you declare. In the case of an array, a "bunch of buckets" exist. Every bucket can contain a single value which defines the physical space for storing the variable.



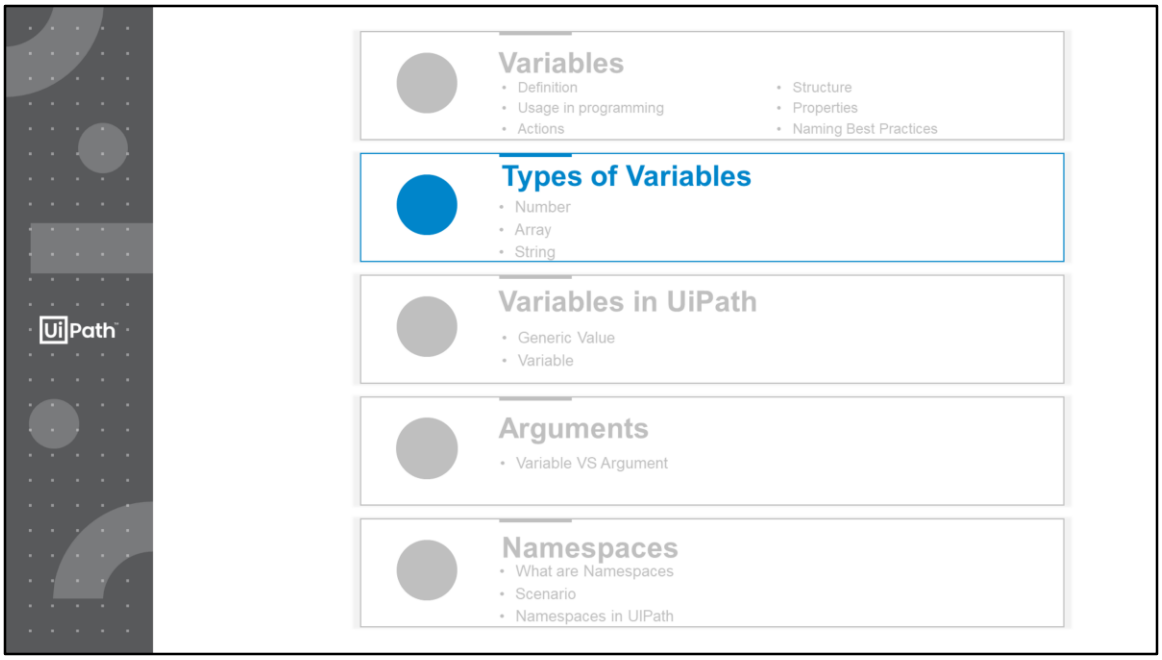
Not variable should be meaningful and describe the information they are storing as the example above XYZ does not describe the nature of variable.

Naming Best Practices: When you start to build a major automation process then it is simple to ignore the variable declaration that what you had done that's why it is crucial to put the naming system good in place. You can always use accurate names, username in the variable which stores the user name.

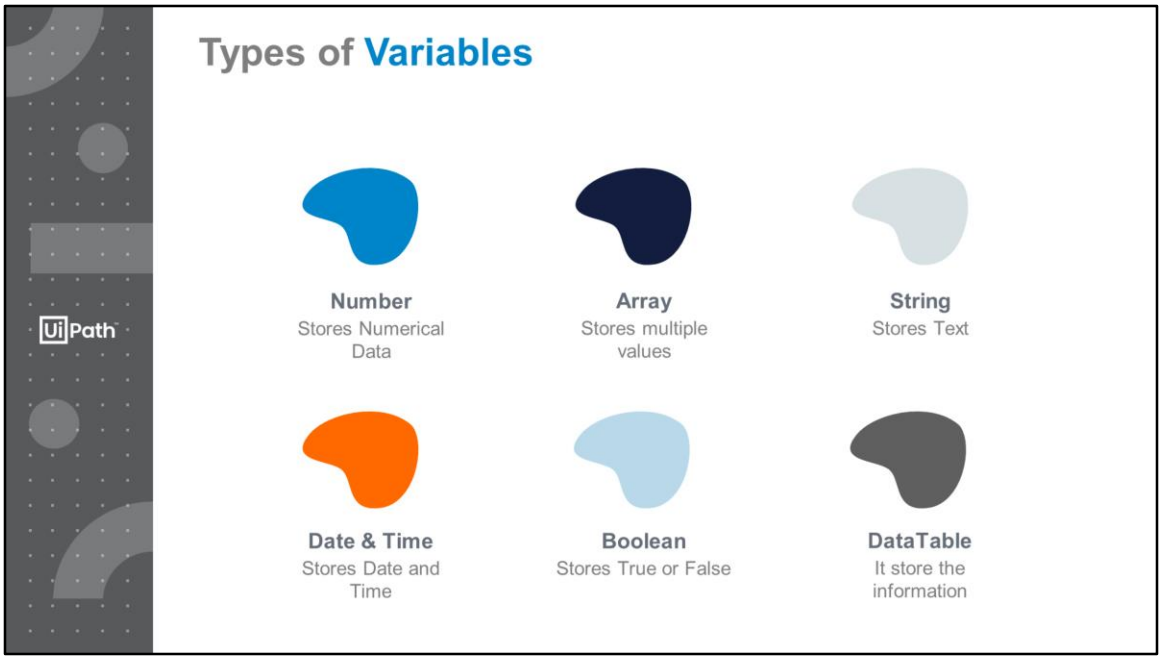
The Naming Best Practices rules that should be applied:

- Be consistent in the way you assign names.
- Use meaningful names: the name of the variable should accurately describe its content.
- Align to company naming standards (if any).
- Align to programming language standards:
  - use Camel Case (aka Upper Camel Case) for classes: VelocityResponseWriter
  - use Lower Case for packages: com.company.project.ui
  - use Mixed Case (aka Lower Camel Case) for variables:

studentName
    - use Upper Case for constants : MAX_PARAMETER_COUNT = 100
    - use Camel Case for enumeration class names and Upper Case for enumeration values.
- Keep your variable names short.
- Try to see if the code can explain itself, it shouldn’t have a lot of comments.

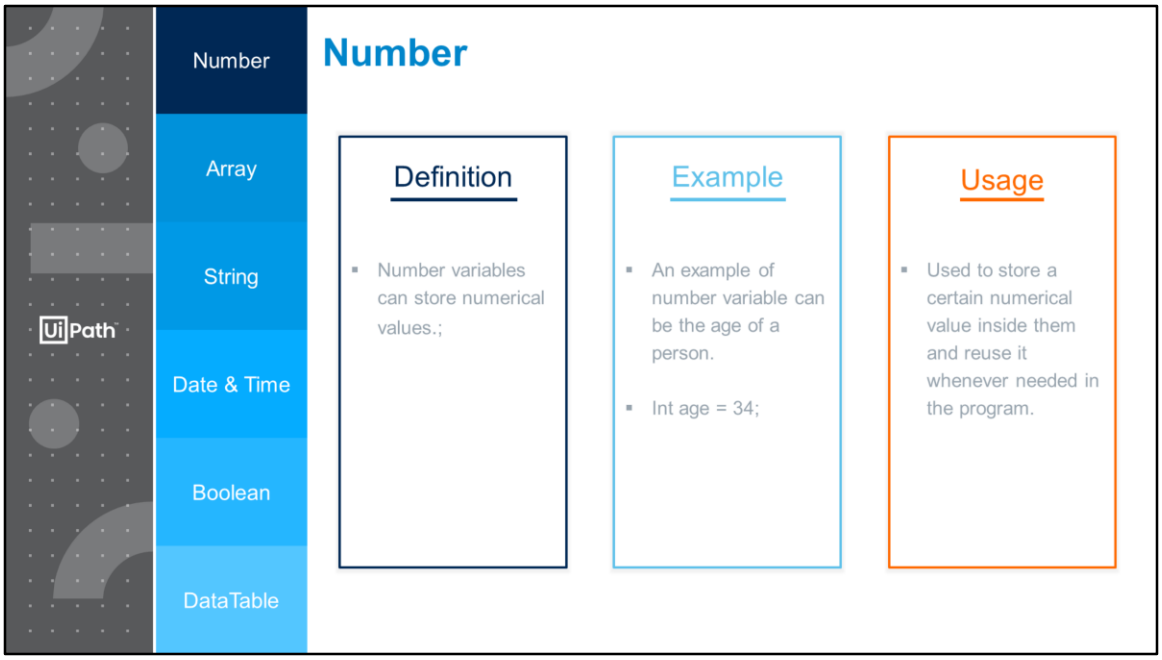


In this topic, we will learn about the different types of variables.



These are the six types of variables which you can use in variable declaration's:

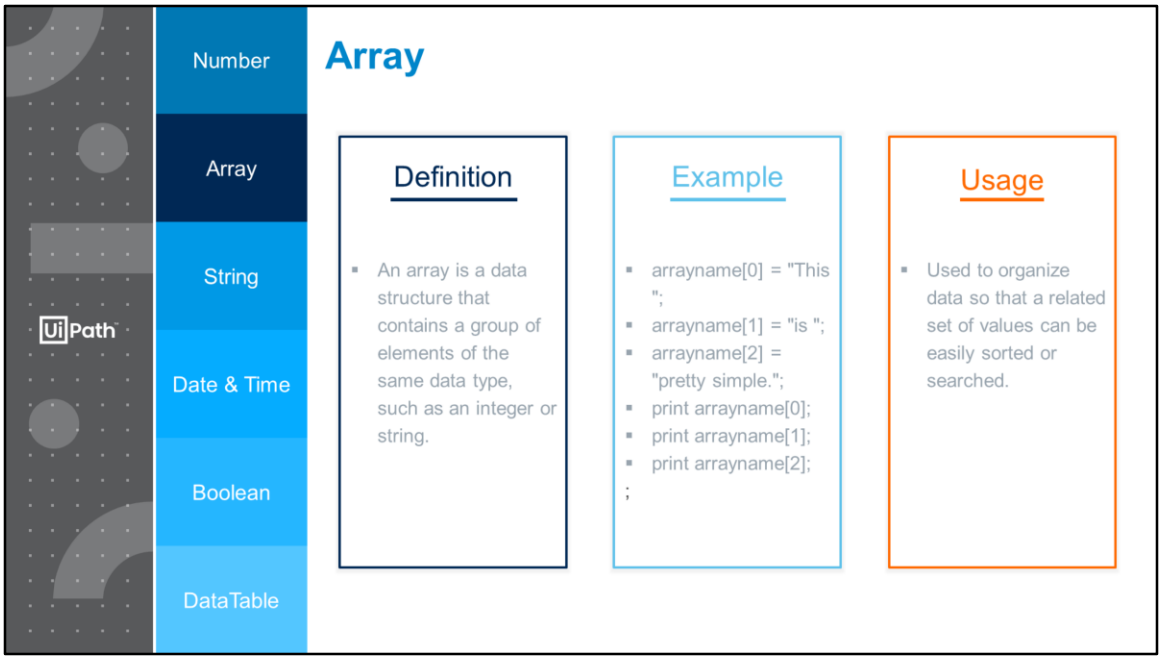
- Number
- Array
- String
- Date & Time
- Boolean
- DataTable



Number variables are types of variables in which you can store numerical values.
Depending on the programming languages, there are different numerical types that can be stored:

- **Int** (signed integers): 10, 299, -100, 0x69
- **Long** (long integers): 5435435343O, -11332424D
- **Float** (floating point real values): 19.35, -46.55
- **Complex** (complex numbers): .874B

An example of number variable can be the age of a person. Int age = 34;
The practical usage of number variables is to store a certain numerical value inside them and reuse it whenever you need in your program.



Array variables are data structures containing groups of elements, usually of the same data type.

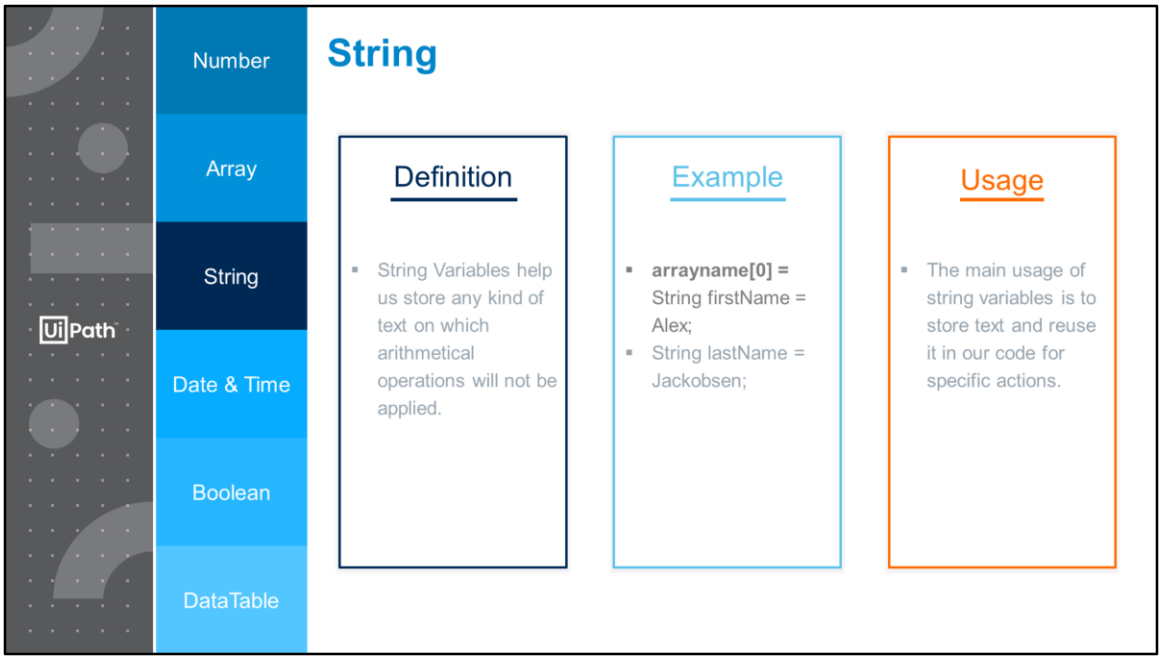
Example
The syntax for storing and displaying the values in an array typically looks something like this:

```
arrayname[0] = "This ";
arrayname[1] = "is ";
arrayname[2] = "pretty simple.";

print arrayname[0];
print arrayname[1];
print arrayname[2];
```

The above commands would print the first three values of the array, or "This is pretty simple." By using a "while" or "for" loop, the programmer can tell the program to output each value in the array until the last value has been reached.

Usage:
A search engine may use an array to store Web pages found in a search performed by the user.
When displaying the results, the program will output one element of the array at a time. This may be done for a specified number of values or until all the values stored in the array have been output.
While the program could create a new variable for each result found, storing the results in an array is more efficient way to manage memory.

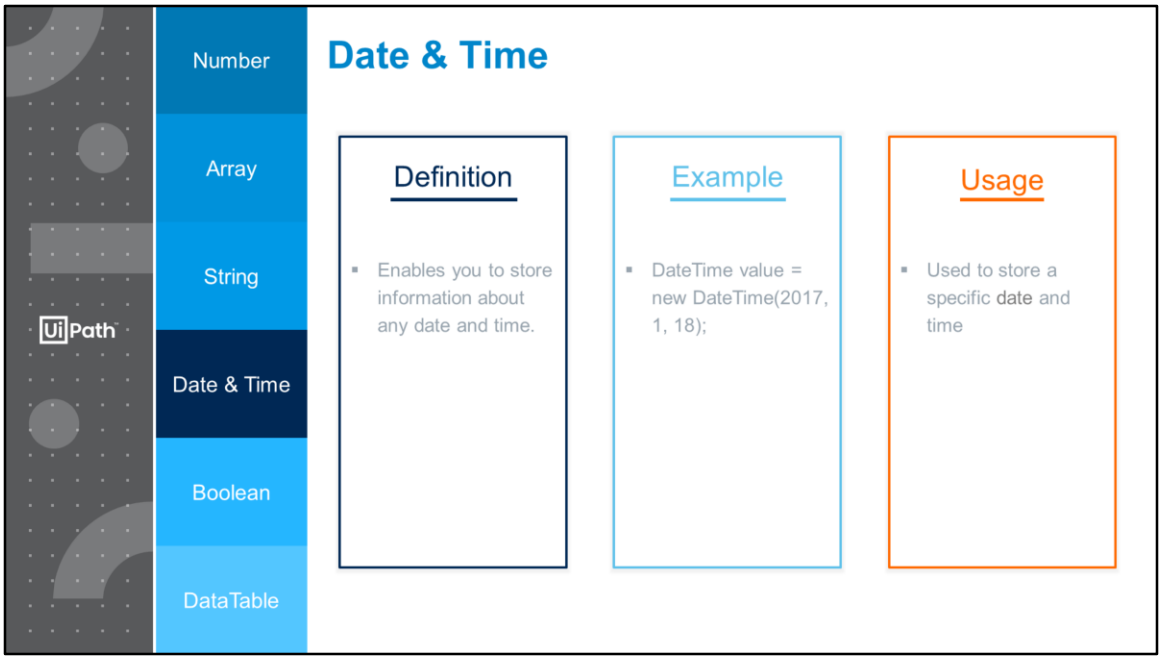


String Variables are variables in which you can store any kind of text and we will not apply any kind of arithmetical operation on them.

Example:

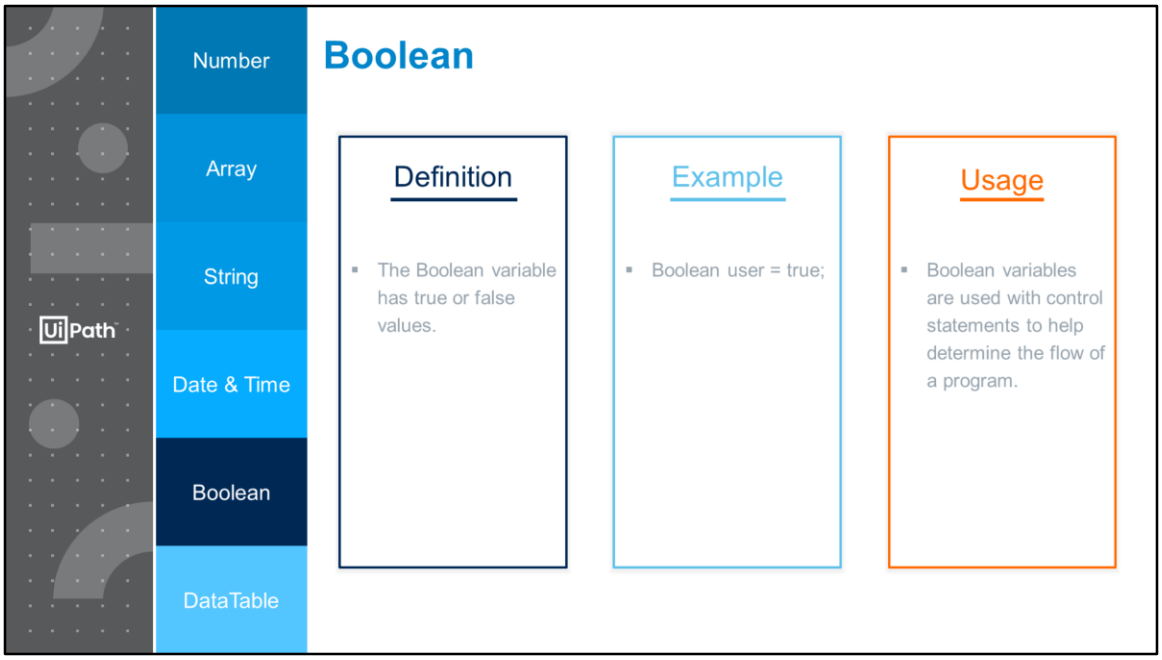
- String firstName = Alex;
- String lastName = Jackobsen;

The main usage of string variables is to store text and reuse it in our code for specific actions.

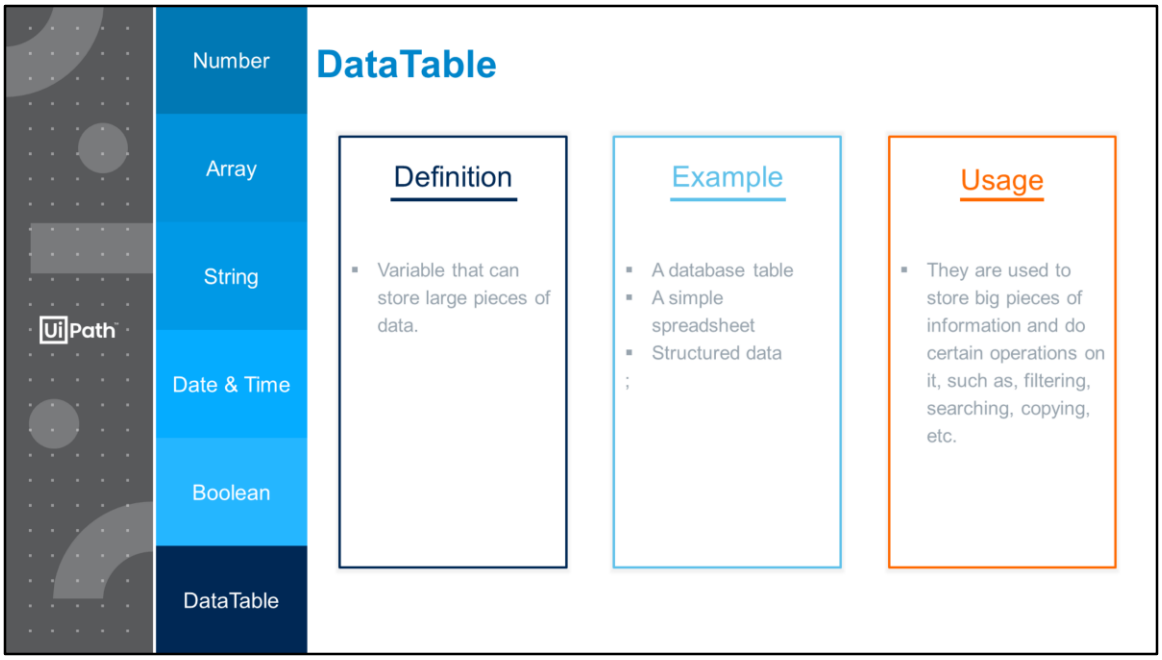


The date and time variable is a type of variable that enables you to store information about any date and time.

Example
DateTime value = new DateTime(2017, 1, 18);
The main usage of Date & Time variable is to store a specific date and time inside or to perform other operations with them. (Ex. Calculate how many days are left until the end of the month).

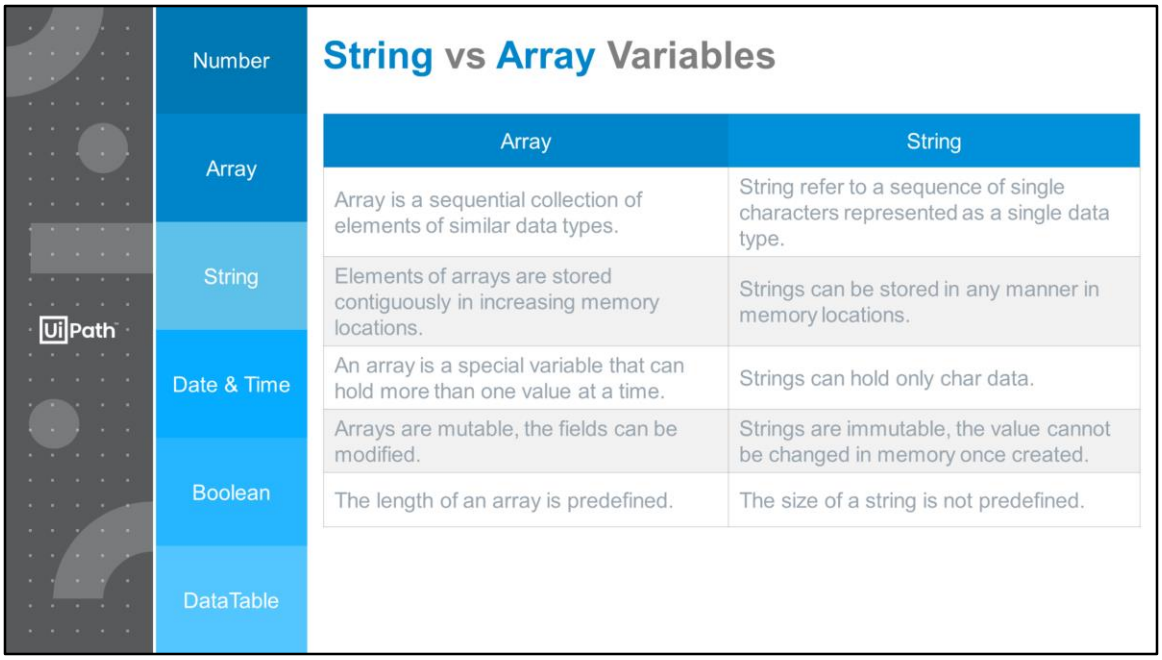


A Boolean variable has only two possible values: true or false. It is common to use Boolean variable with control statements to determine the flow of a program.
The Boolean data type is primarily associated with conditional statements, which allow different actions by changing control flow depending on whether a programmer-specified Boolean condition evaluates to true or false.



DataTable variables represent a type of variable that can store the information and act as a database, or a simple spreadsheet with rows and columns. In UiPath Studio, DataTable variables can be found in the Browse and Select a .Net Type window, under the System.Data namespace.

The practical usage of data tables is to store big pieces of information and do certain operations on it, such as, filtering, searching, copying, etc. They are often used to migrate data from a database to another, extract information from a website and store it locally in a spreadsheet.



An **array** is a fixed-size sequenced collection of elements of the same base types that share a single name and can be used to represent a list of names or numbers.

A **string** is similar to an array with a few exceptions. It is a sequence of characters that are represented as a single data item.

Webform Exercise

Think about the variables you need in a webform

Your client is a company that collects recycled oil.

They need to have a webform created for them with the purpose of scheduling the time for collecting the oil from the customers.

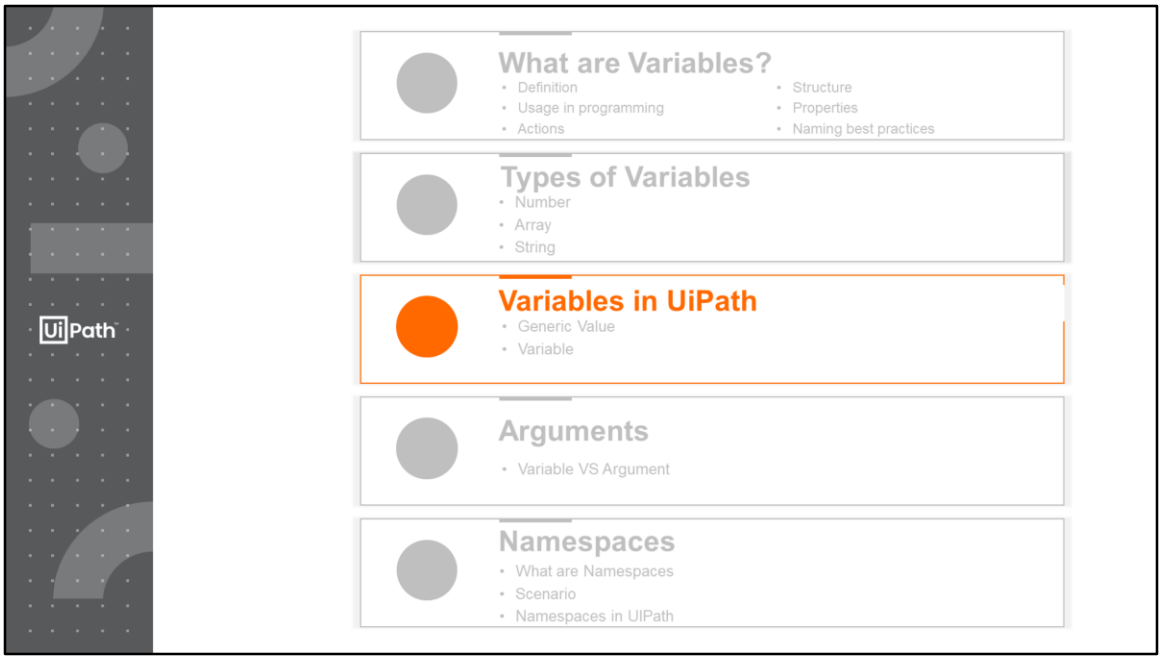
They would need to collect data about customers:

- First and last name
- Age
- Address
- Date when the oil can be collected
- Time range in which the oil can be collected
- Phone number
- Comments

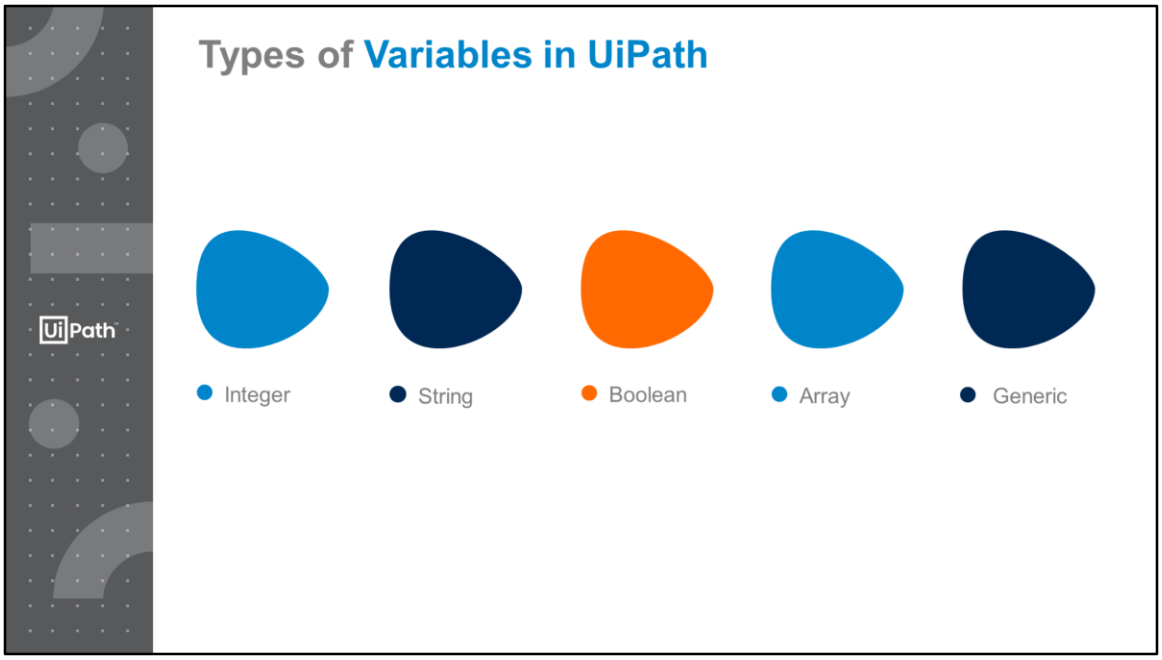
What variable types would be used?

The following variables will be needed in the webform:

- First and last name: String
- Age: Integer
- Address: Array
- Date when the oil can be collected: Date & Time
- Time range in which the oil can be collected: Date & Time
- Phone number: Integer
- Comments: String

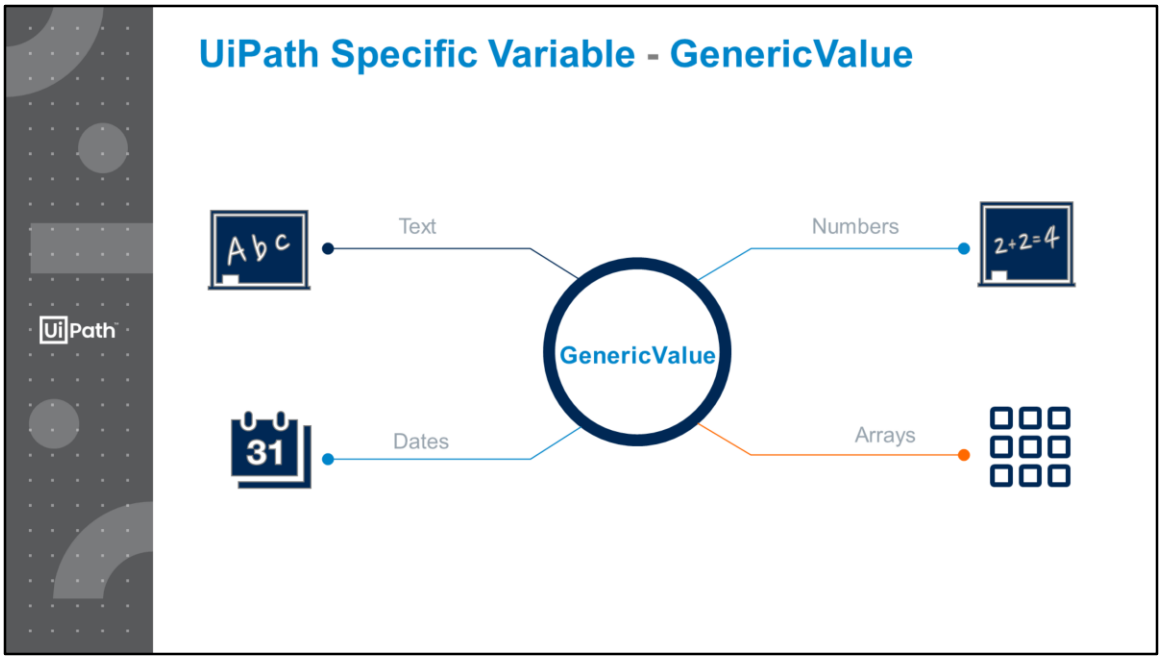


In this topic, we will learn about variables in UiPath.



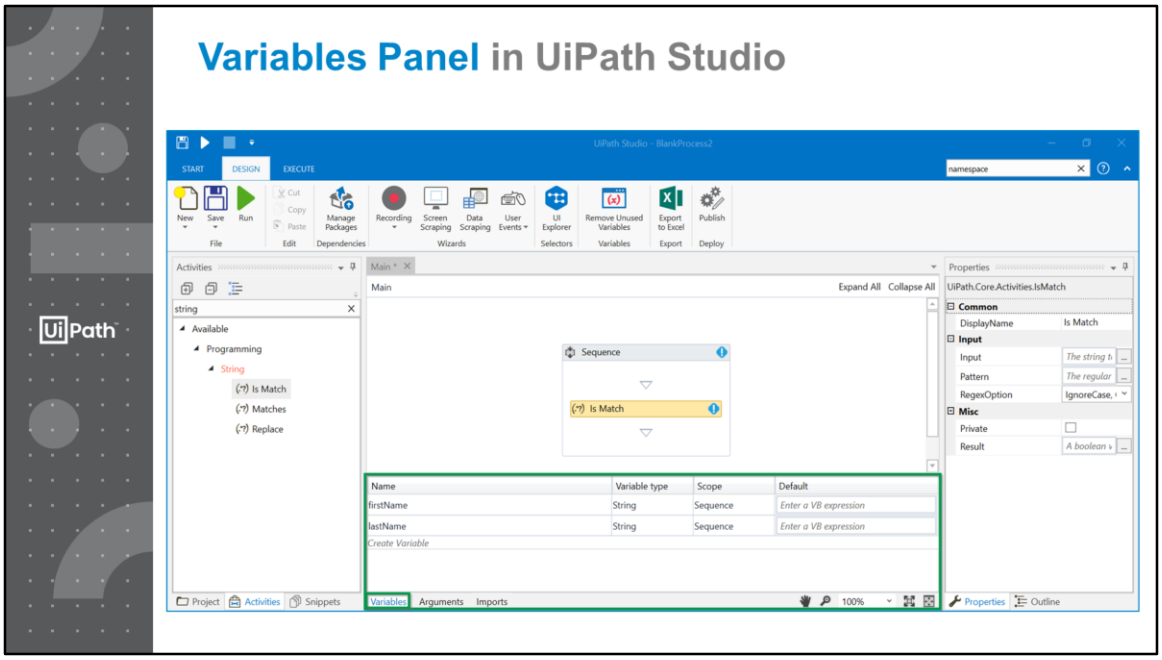
UiPath works with various types of variable: numbers, text, images, files, color and many others
Thee most common types of variables are:

Integer: Whole number (1,2,3, 4353)
String: Text of any kind (abc, 123, !, @#)
Boolean: True or False
Array: A list of any type of data



The GenericValue is a type of variable that is particular to UiPath Studio and can store any kind of data, including text, numbers, dates, and arrays. GenericValue variables are automatically converted to other types, in order to perform certain actions. However, it is important to use these types of variables carefully, as the conversion may not always be the correct one for the project.

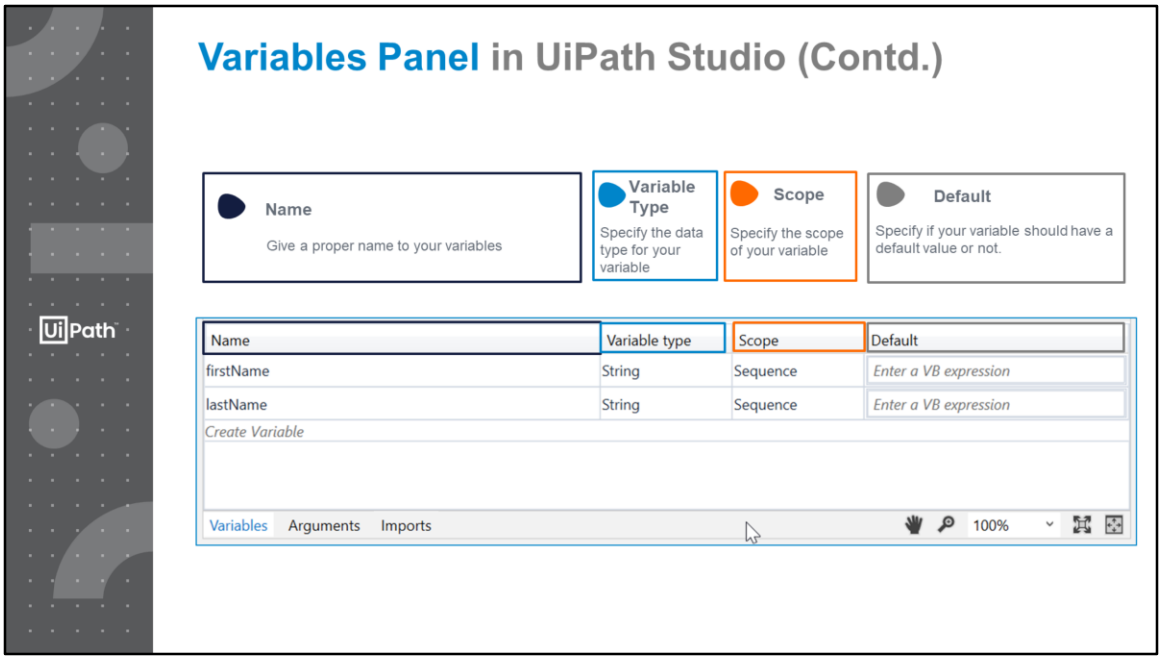
UiPath Studio has an automatic conversion mechanism of GenericValue variables. First element in your expression takes precedence while deciding the outcome. For example, when you try to add two GenericValue variables, if the first one in the expression is defined as a string, the result is the concatenation of the two. If it is defined as an Integer, the result is their sum.



Variables are easy to find in UiPath.

The Variables Panel is located at the bottom right side of the Activities and it is hidden by default.
When clicked, the Panel opens and shows the currently defined variables.

To create a new Variable, click on the first row in the panel and type a name for it.
Let’s move on the further look at the properties of a Variable in UiPath.



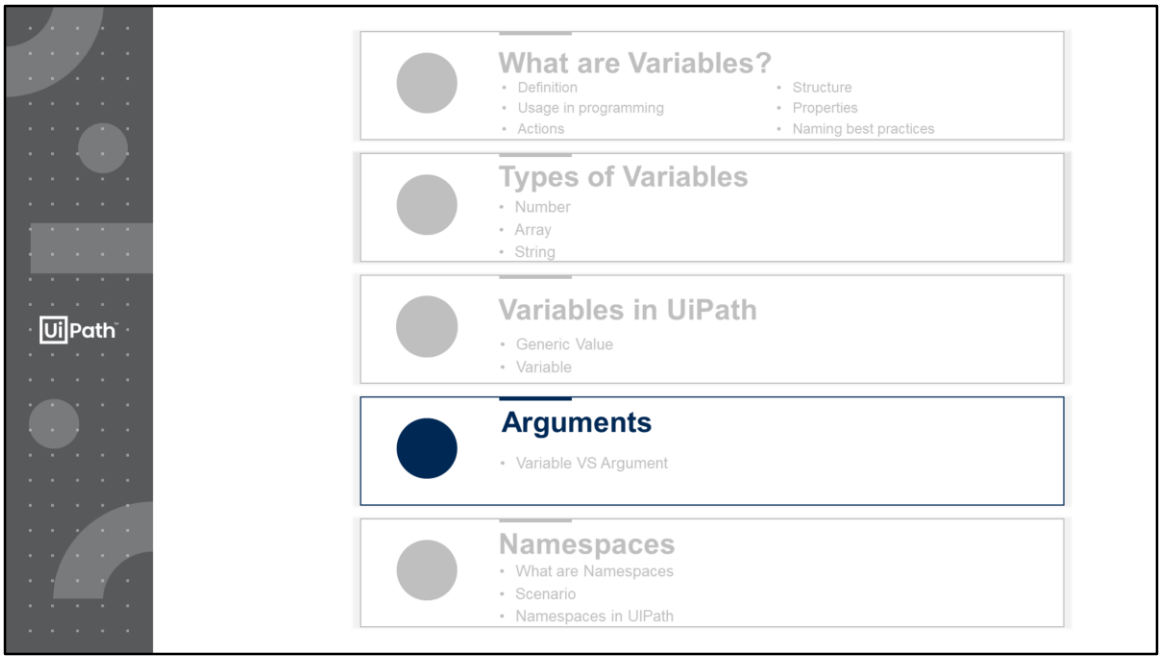
The properties of Variables in UiPath are as follows:

Name: Here, You can give a name to your variables. If you rename a variable in the Variables pane, it will be renamed in all instances where it's used. This is useful in complex workflows.

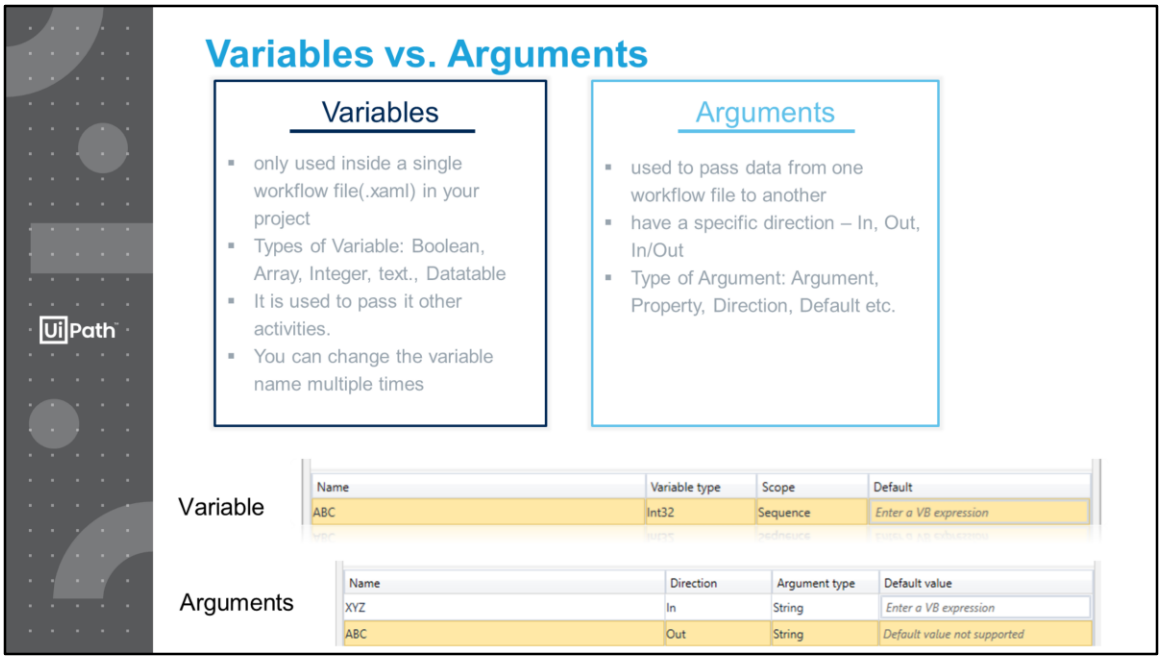
Variable Type: Here you can specify the data type that you will store. These variables can be in the string, Array, GenericValue, Int32, and many more forms. Also you can change insert the variable type library by clicking browse types and search any variable library.

Scope: You can specify the scope of your variable in which it can be visible. It define the scope by public and private method, Where if you define any variable scope in the container then it is applicable only for those containers where you define it that is private scope. But if you want to publicly define the scope then you can define in the main class which is applicable for all containers that are called public scope.

Default: This is where you specify Variable’s default value.

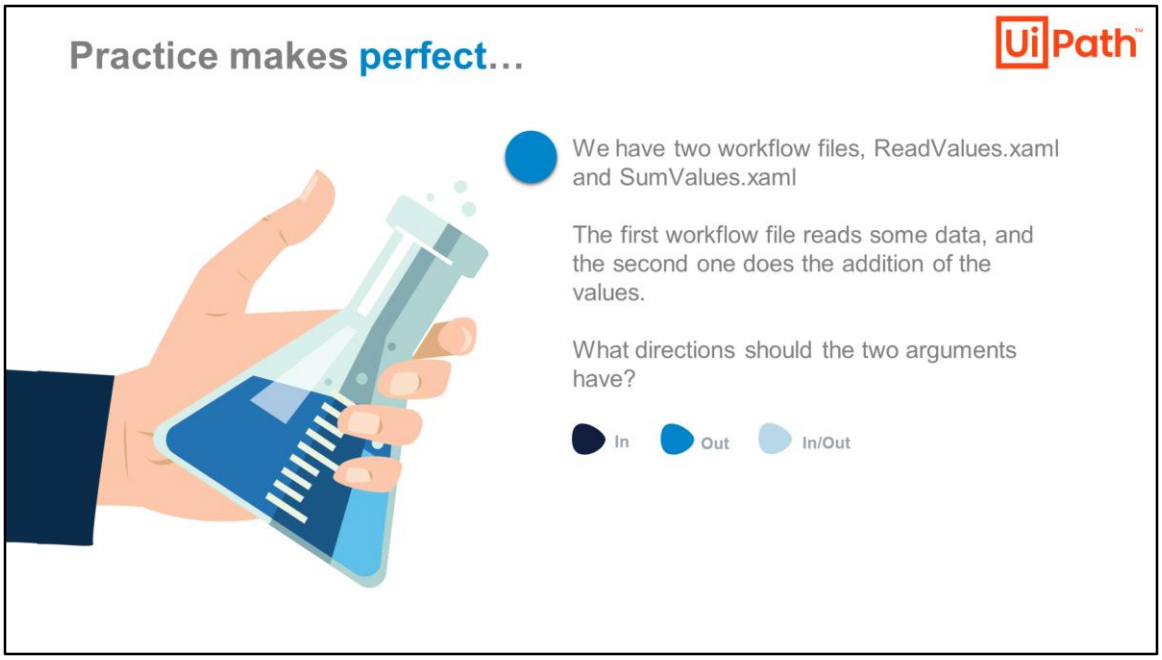


In this topic, we will learn about Arguments in UiPath.

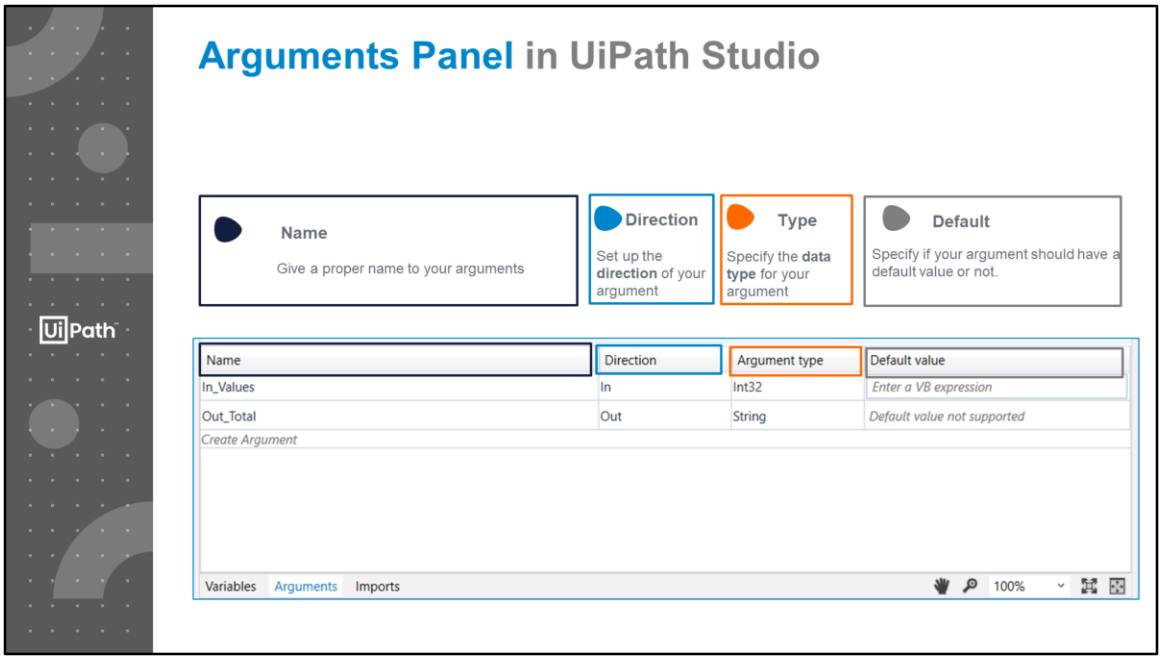


While variables pass data between activities, arguments pass data between automation, enabling you to use automation time and time again. Managing arguments is similar to the variables.

The key difference is the differentiator. For example, if the direction is an argument then you need to ensure the direction is a correct one. Apart from this, in the In/Out properties tell the application that where the information stored in them is supposed to go.

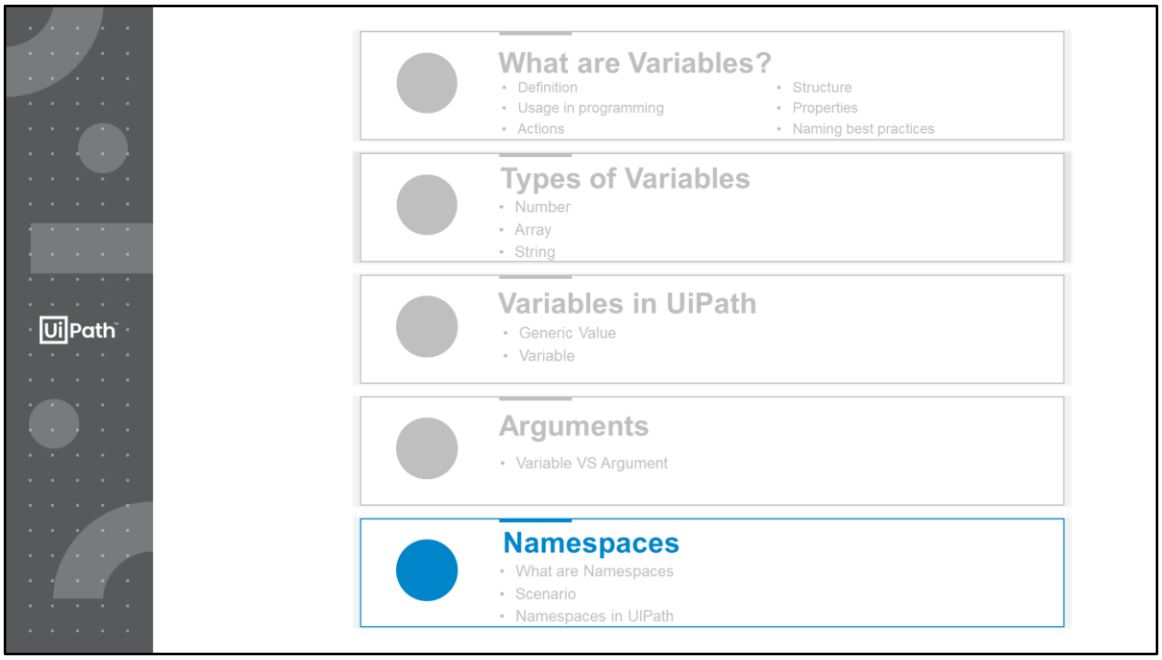


The first workflow file reads some data, and the second one does the addition of the values.
The first workflow(ReadValues.xaml) should have the arguments with the direction Out, as we are sending out data.
The second workflow(SumValues.xaml) should have the same arguments, but with the direction In, as we are receiving data.

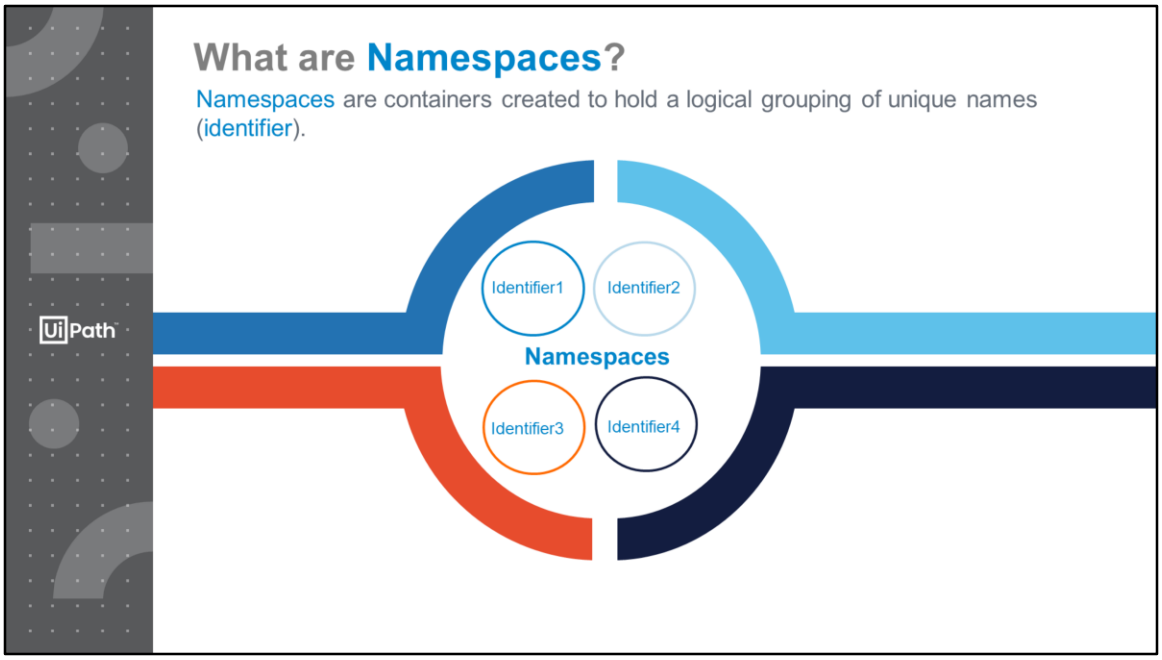


Similar to how the variables panel works, the Arguments panel is the main area where you can make changes to your arguments. This is the argument panel where you can create the argument and see the declared argument including with these:

1. Name (it define the argument name),
2. Direction (it is a set up or process that your argument is IN or OUT form)
3. Type (In it you can set your argument data type in form of Int32, Array, String and many more)
4. Default (It specify your argument value that is right or not)

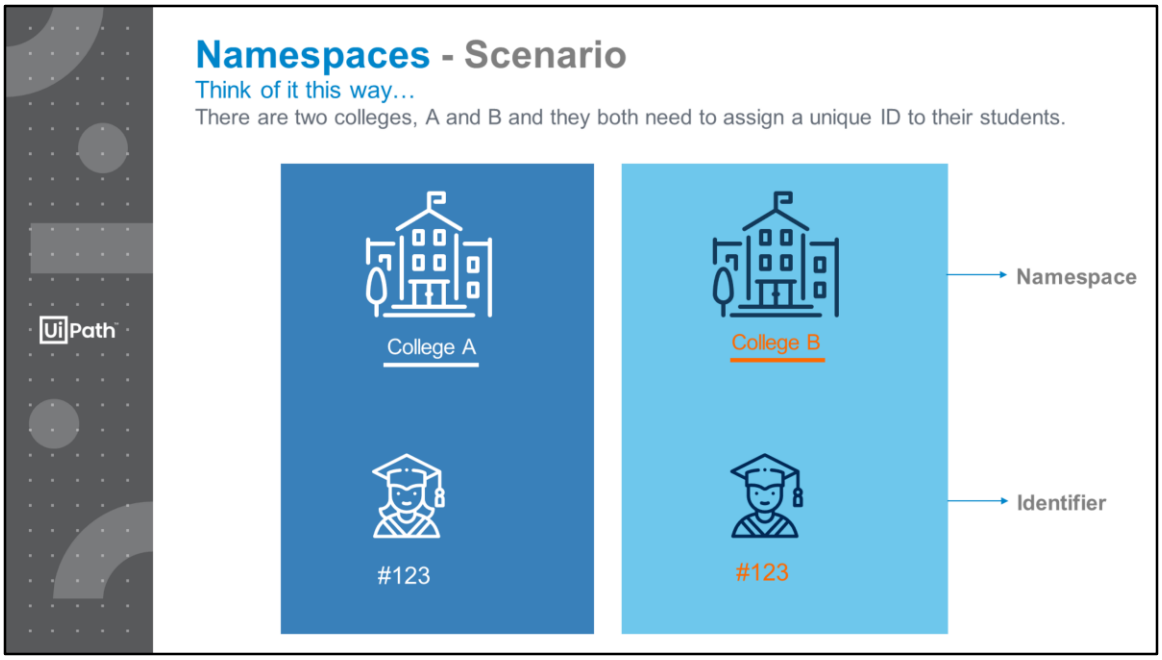


In this topic, we will learn about Namespaces in UiPath.

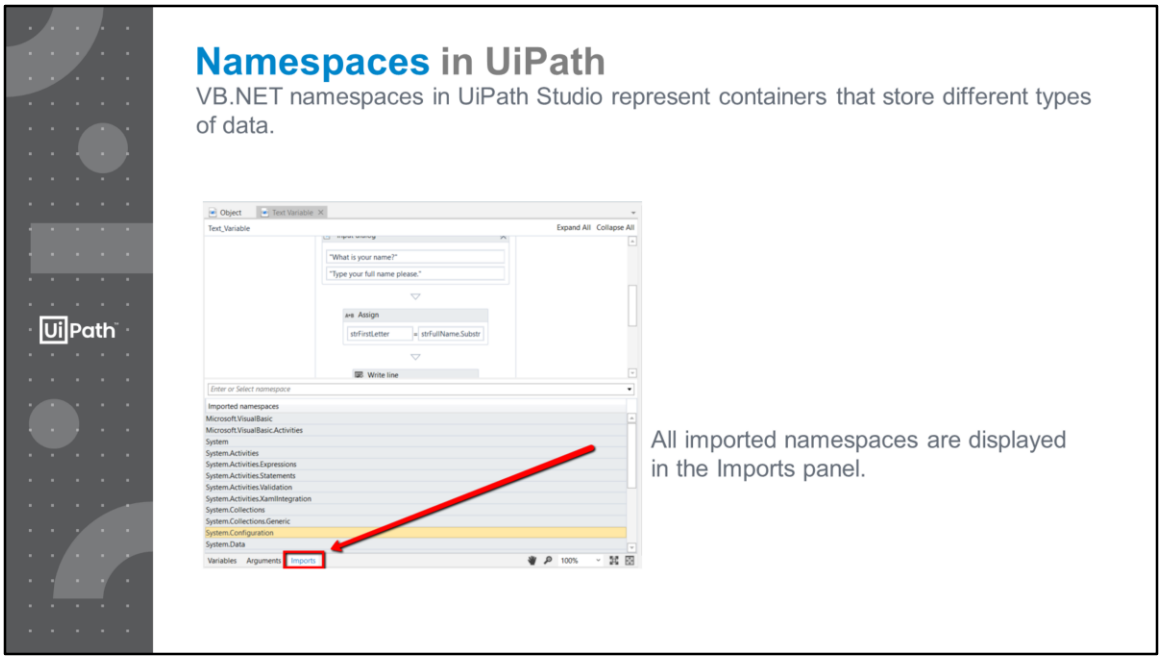


Namespaces are used for grouping symbols and identifiers around a particular functionality.
It is also used to avoid name collisions between multiple identifiers that share the same name.

How do they work?
Namespaces allow you to divide the program into specific spaces so that the names inside the code don't get confused with other bits of code and create clashes.



That means college A should not have two students with the same ID and neither should college B.
But there's no problem for the same ID number to be used in both schools.
If Sandy is a student in college A and Daniel is a student in college B, then it is not a problem for each of them to be have the ID #123.
In this example, the ID number is the identifier, and the college serves as the namespace.
It does not cause problems for the same identifier to identify a different student in each namespace.



VB.NET namespaces in UiPath Studio represent containers that store different types of data. They enable you to easily define the scope of your expressions, variables and arguments. Some namespaces are automatically imported when you browse for a .Net type variable or argument.

For example, if you have the System.Data namespace imported, you can further use DataTable, DataView, DataColumn, DataRow and other classes that are available in it, without having to always type System.Data.DataTable and so on.

All imported namespaces are displayed in the **Imports** panel. Note that some namespaces are automatically imported when you browse for a .Net type variable or argument.

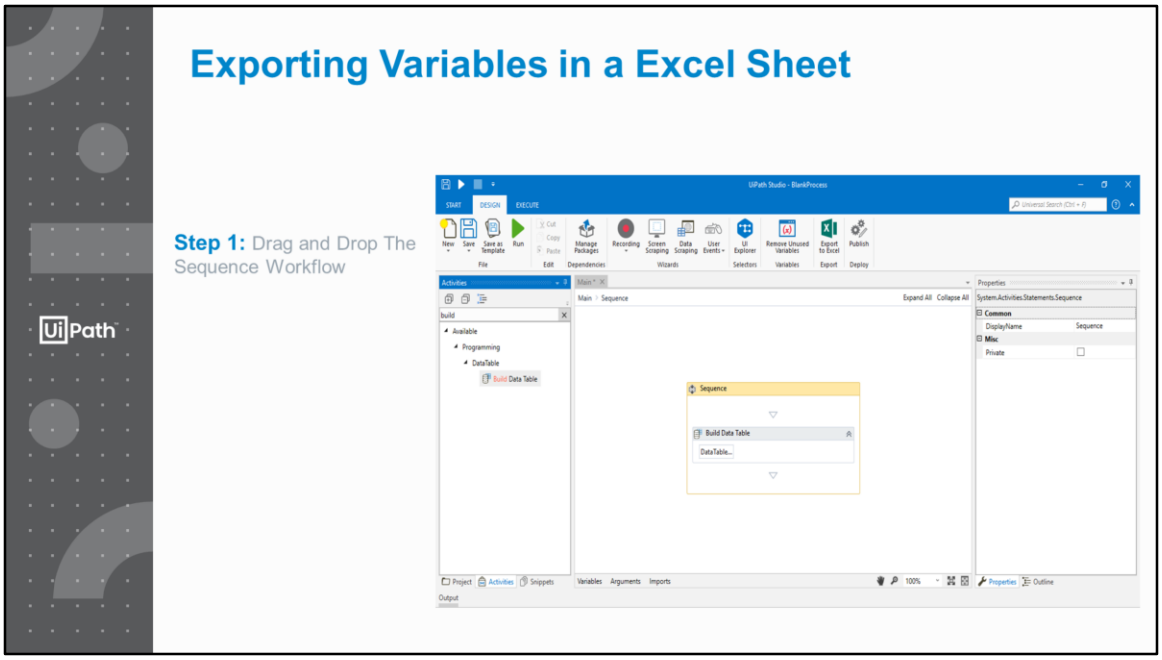
To open this panel, click **Imports** in the **Designer** panel.

Back to our Exercise

Think about the variables you need in a excel sheet

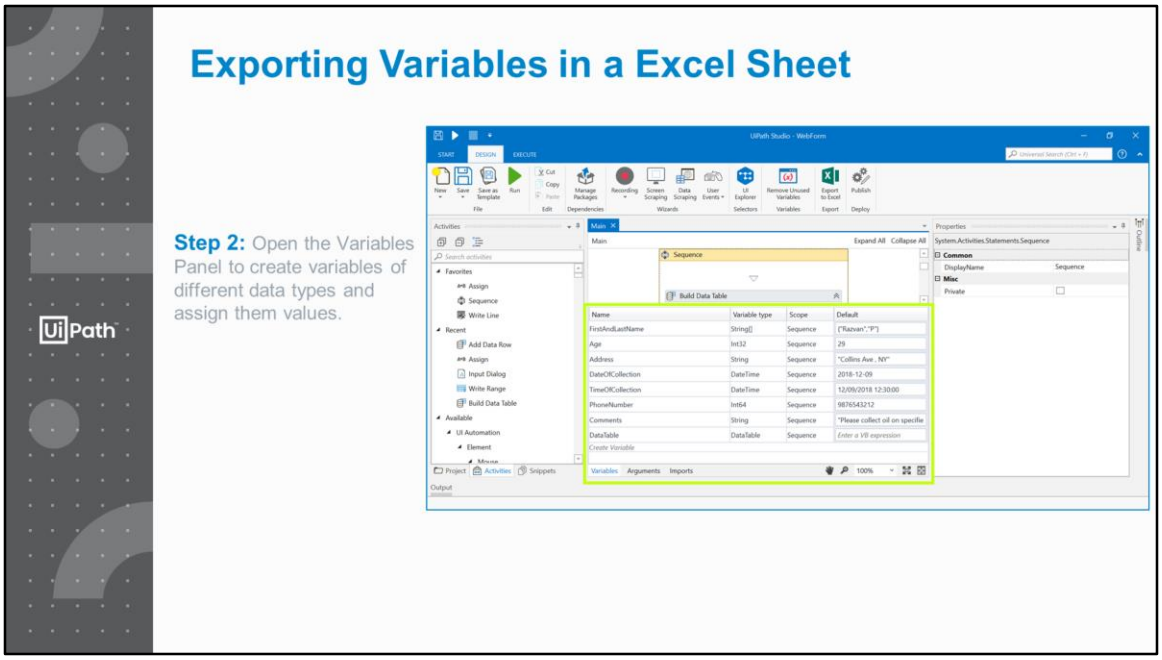
Now that you are familiar with Variables, Arguments and Namespaces in UiPath, can you define the project task into UiPath Studio?

To follow a step-by-step presentation of how this solution is built with UiPath, navigate to the next slides.



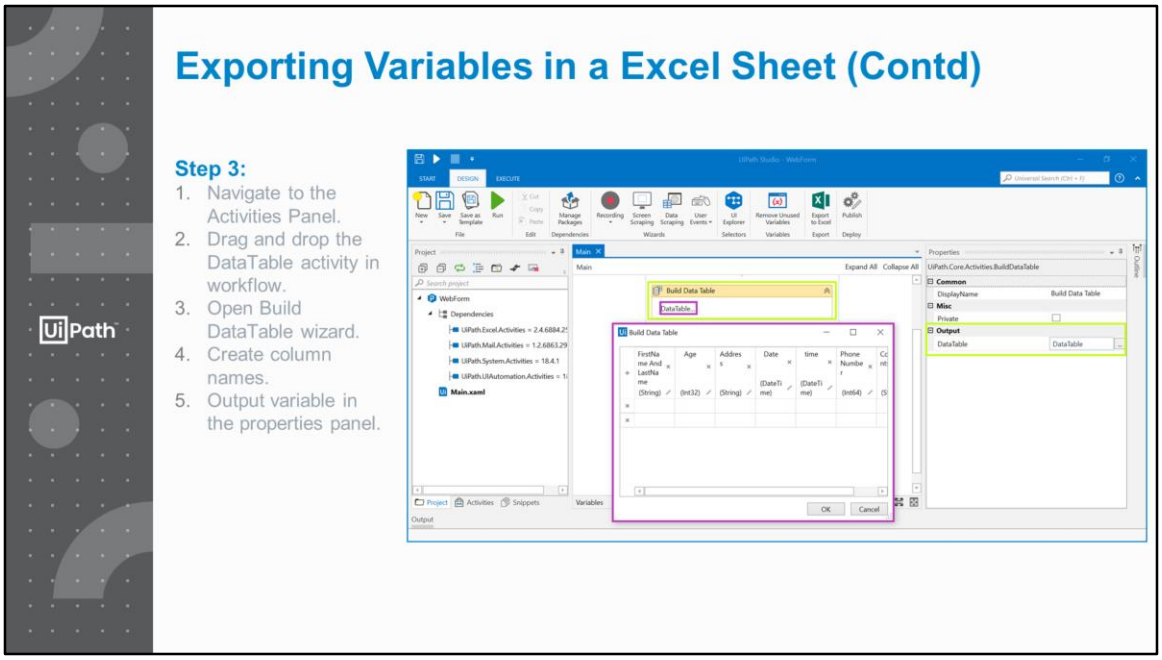
With the help of this example you can understand the variable exporting method in a Excel sheet.

Step 1: Drag and drop a sequence
Step 2: Drag the build data table in sequence.



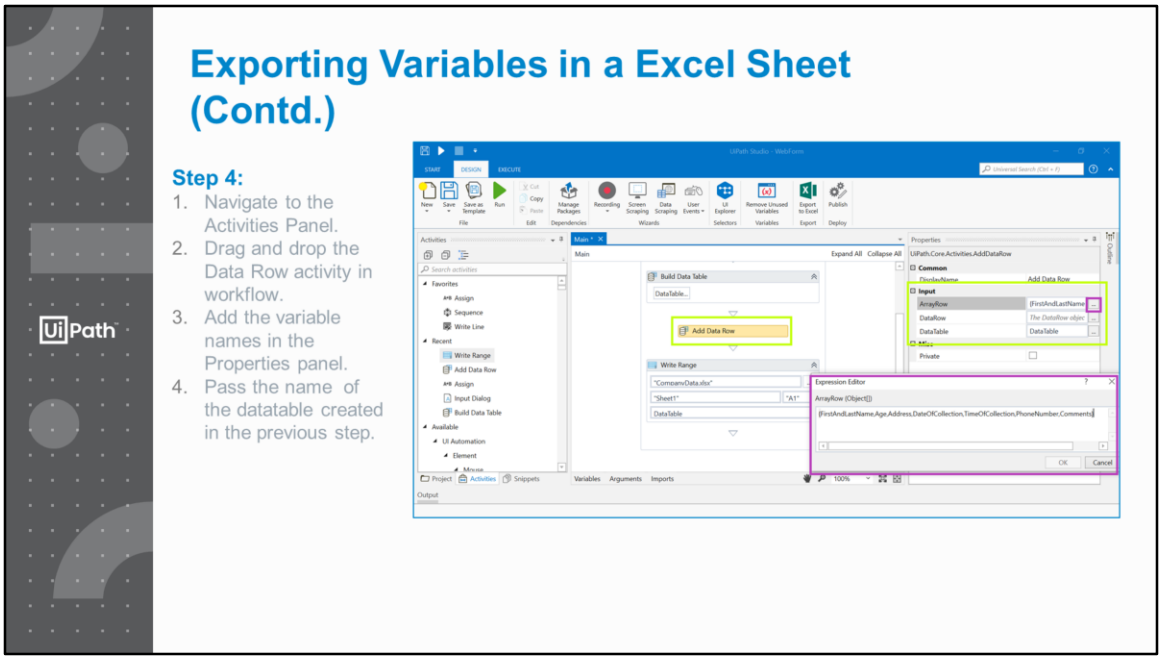
Purpose: we have to extract data from a webpage and fill it in an excel for that we have to create multiple variables according to textboxes(Forms) to be filled and by using BuildDataTable, we can fill all the details in excel.

Steps 3: Open the Variables Panel to create variables of different data types and assign them values.



Navigate to the Activities Panel, search for the DataTable activity and drag and drop it in our workflow. Click on the DataTable button to get the Build DataTable wizard to create column names. Set the types same as the variables and create and output variable in the properties panel. It should have datatype as Datatable.

A. Navigate to the Activities Panel.
B. Drag and drop the DataTable activity in workflow.
C. Open Build DataTable wizard.
D. Create column names.
E. Output variable in the properties panel.



Step 4: Navigate to the Activities panel again, search for a Data Row activity and drag and drop it in the workflow. In the Properties panel under ArrayRow add the variable names as shown and pass the datatable name as created in the previous step.

Additional Note*

1. Data Row or Add Data Row?

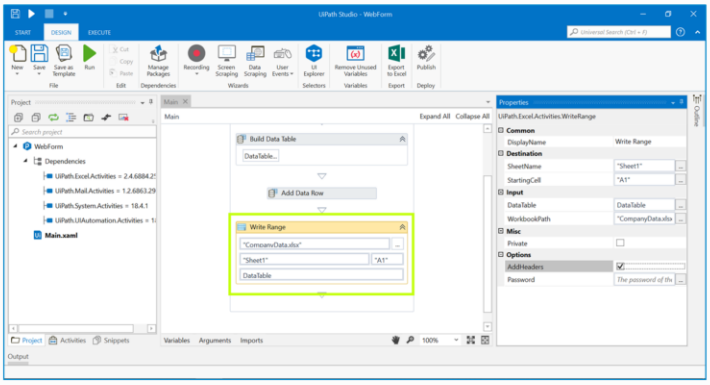
Data Row: In a datatable when you read a data row wise that is called as a data row.

Add data Row: It's an activity its add an datarow in the specified data table.

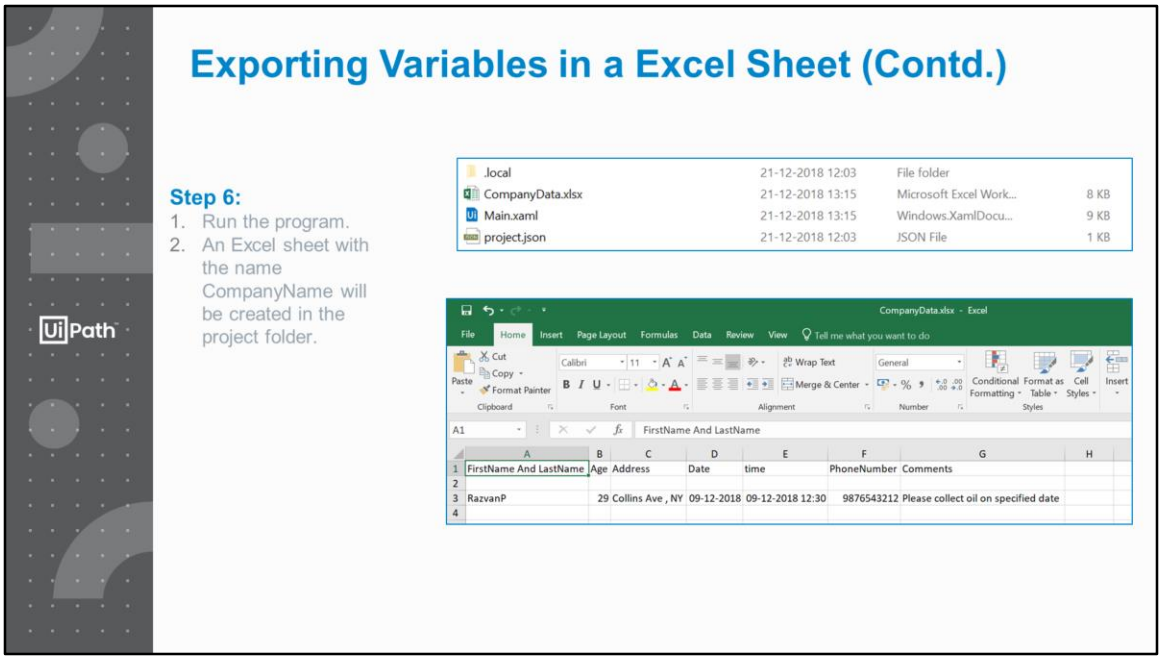
Exporting Variables in a Excel Sheet (Contd.)

Step 5:

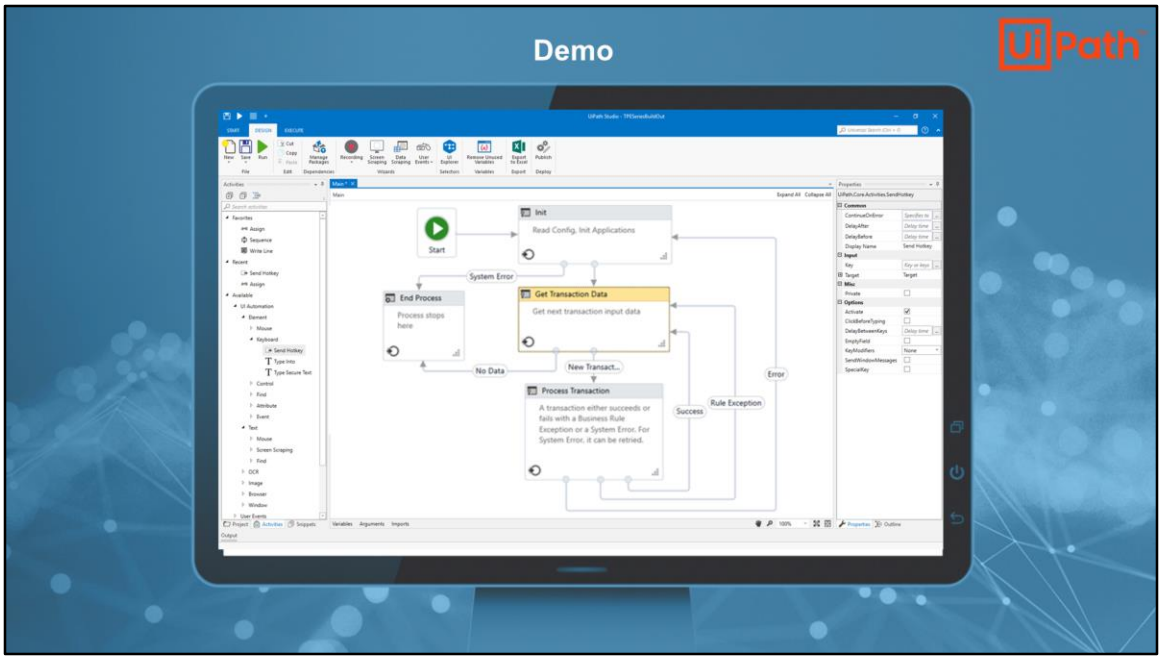
1. Navigate to the Activities Panel.
2. Drag and drop the Write Range activity in workflow.
3. Write "CompanyData.xlsx" in workbook path.
4. Pass the datatable output variable name created in the previous step.



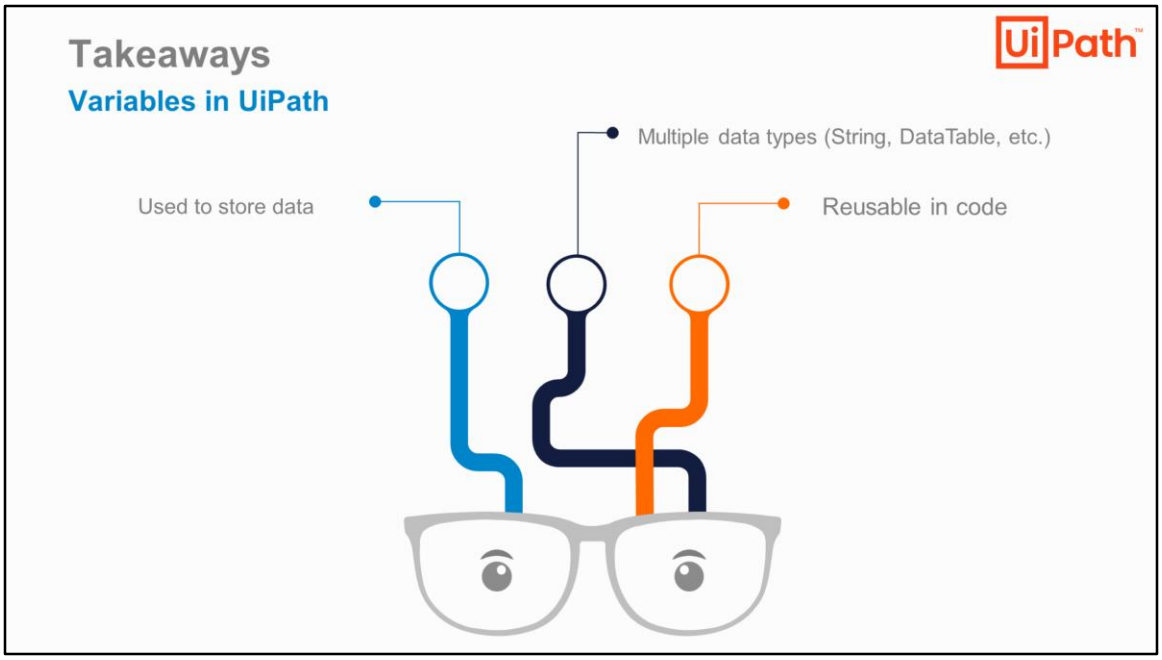
Step 5: Navigate to the Activities Panel again, search for a Write Range activity and drag and drop it in the workflow. Then write "CompanyData.xlsx" in workbook path and pass the datatable output variable name created in step 2 in the datatable property field.



Step 6: Run the program. An Excel sheet with the name CompanyName will be created in the project folder.



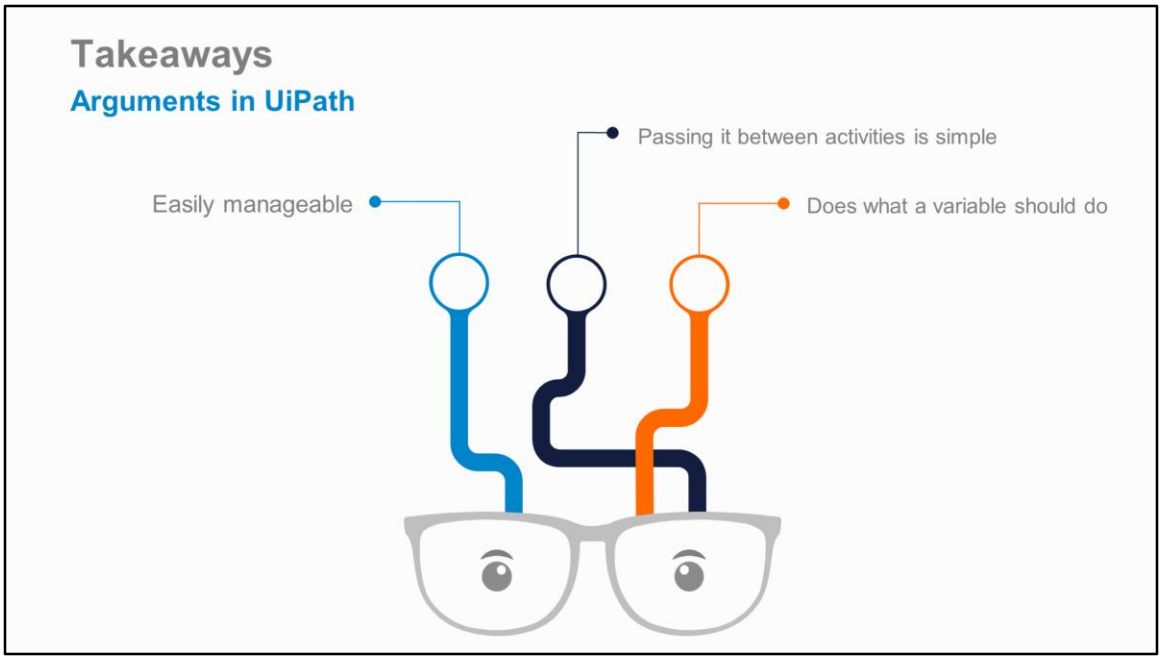
Let’s now have a look at an example workflow built in UiPath where Variables are used to automate a process.



The point of the Recap & Summary section is to go through the most important points covered in the lesson, after the students had the chance to see them in practice and obtain a consolidated view.

The teacher should use facilitation questions to help the students map the key points and offer a safe space to get questions and comments from them.

Some examples of facilitation questions are:
1. What are the Variable??
2. What are the types of variables?
3. How does String variable differ from Array variable?
4. What is GenericValue variable?

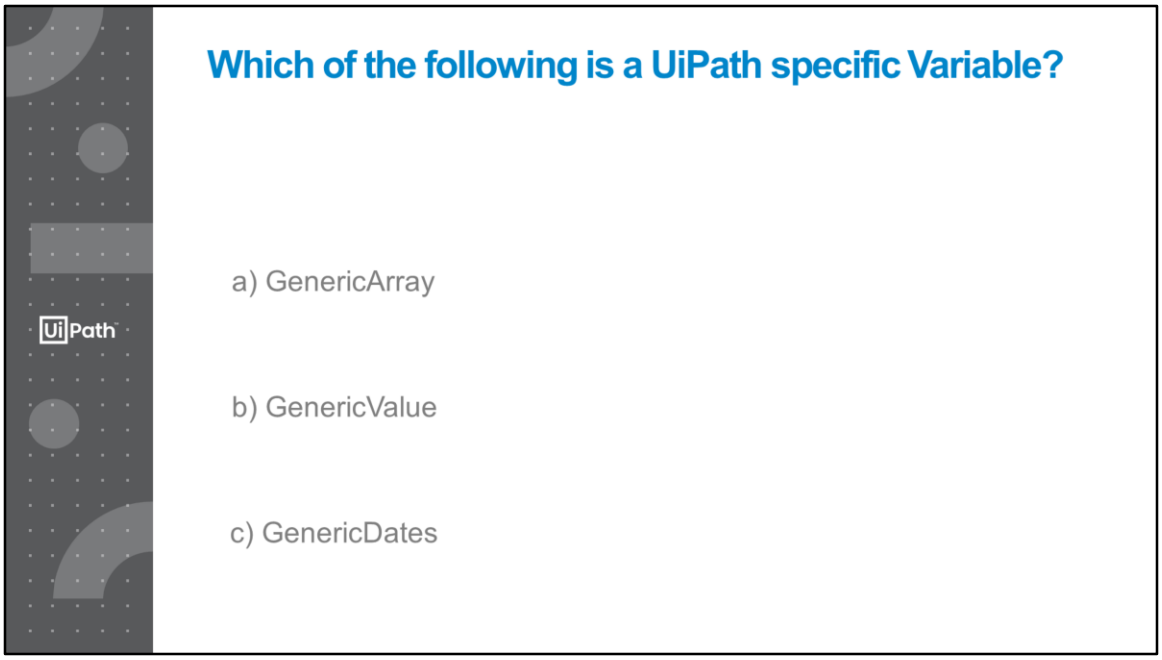


The point of the Recap & Summary section is to go through the most important points covered in the lesson, after the students had the chance to see them in practice and obtain a consolidated view.

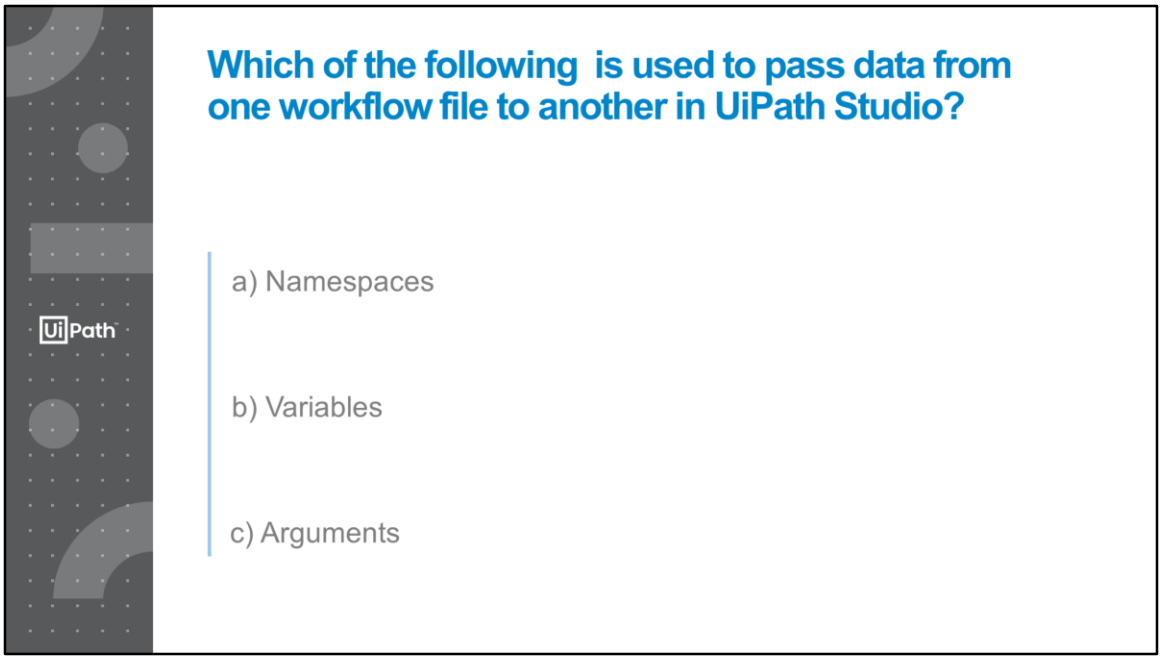
The teacher should use facilitation questions to help the students map the key points and offer a safe space to get questions and comments from them.

Some examples of facilitation questions are:
How does an argument differ from a variable?
What are Namespaces?

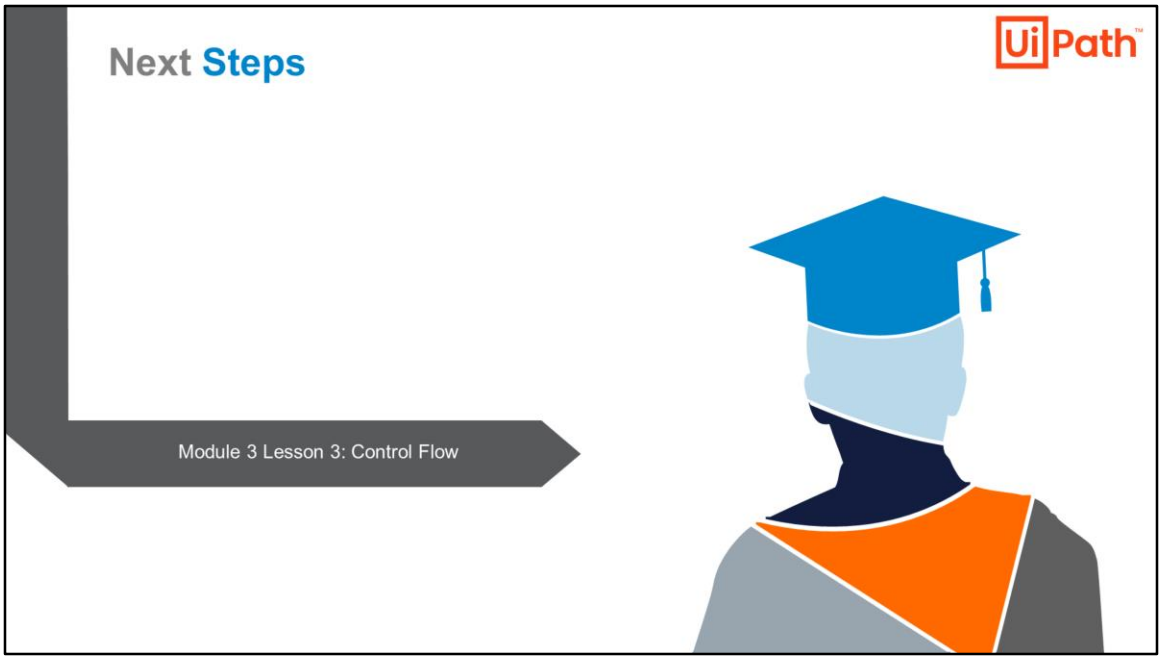
Q&A
Now it’s your turn. What’s on your mind at the end of this?



Correct answer: b) GenericValue is a UiPath specific Variable.



Correct answer: c) Arguments are used to pass data from one workflow file to another in UiPath Studio.



In the next lesson, we will be covering Control Flow and its importance in RPA Automation.