

Department of Computer Science and Engineering

RPA Design and Development (CSE552)-(3-0-0-0)



Dr. Bheemappa H
Dept. of CSE
RIT

UNIT II

RPA TOOL INTRODUCTION AND BASICS:

- RPA Development Life Cycle
- How does RPA Work
- Challenges in RPA
- Variables and Types of Variables
- Variables vs. Arguments
- Namespaces, and Importing New Namespace.

Stages of RPA Lifecycle:

A structure to the automation process where at each stage we can ensure that the implementation is as expected

Stage 1 – Identification

Stage 2 – Discovery/Analysis

Stage 3 – Design

Stage 4 – Development

Stage 5 – Testing

Stage 6 – Implementation

Challenges in RPA

- **Integrations and Dependent Processes**
- **Ownership/Infrastructure issues**
- Process Analysis Issues
- **Identifying right processes and qualifying them for automation**

How does RPA Work ?

- RPA works at a user interface level.
- It interacts with computer the same way as human.
- RPA consists of software bots
 - These software bots use your software programs and systems to complete processes in the same way a human would
- Training the Bot

For example, opening files or emails, copy-pasting fields and inputting data. From this, the bots learn how to complete those tasks (by replicating your actions).

RPA works up-front on the interface of your applications

Robotic Processes Design

- Use a visual interface of **RPA**.

Main types of project supported

Sequence:

Flowchart:

Assistant:

State machine

The user interface of UiPath

- Designer panel
 - Consist of steps and activities of the projects.
 - We will be organizing various activities in a flow or chain to accomplish a task
- Properties panel
 - viewing the properties of the activities.
 - For making any changes, if required.
- Activity panel
 - Contains all the activities that can be used in building the project.
 - Dragging and dropping the required activity into the required location in the **Designer panel**
- **Argument**

Activities

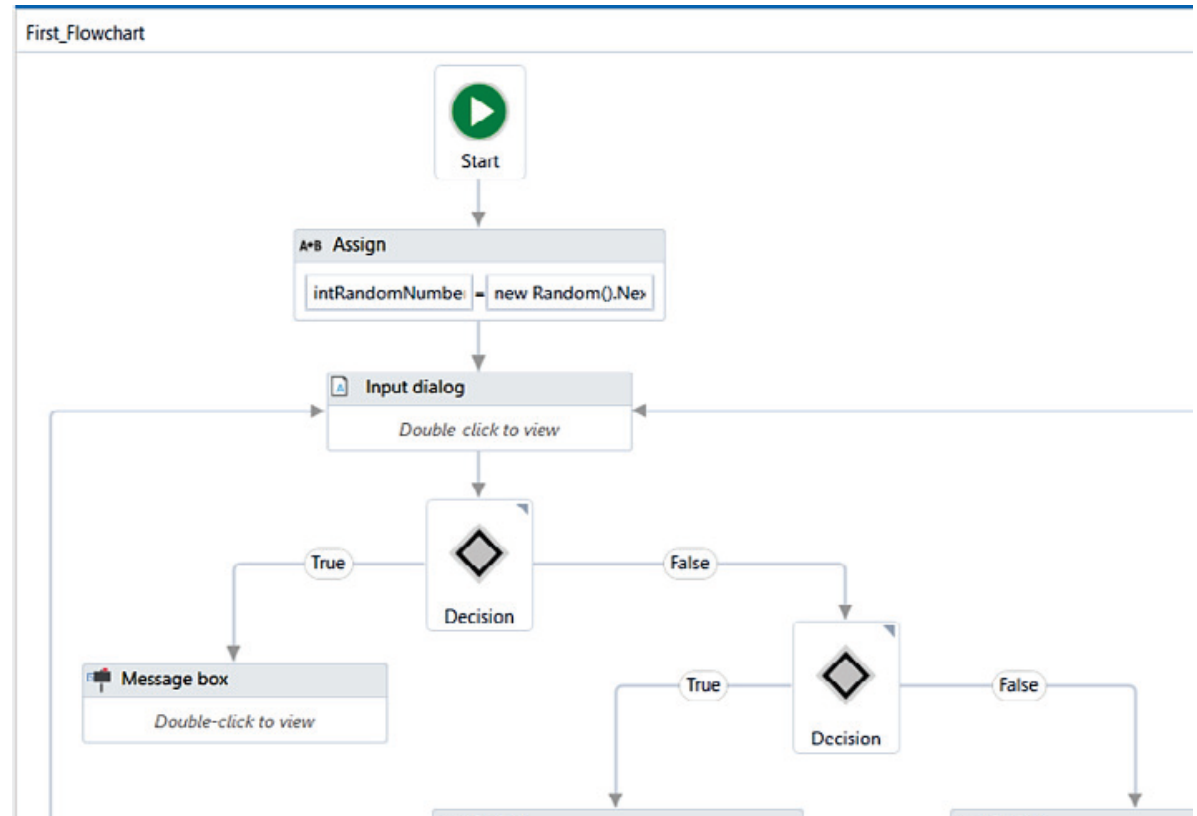
- an **activity** represents the unit of an action. Each activity performs some
- Action. When these activities combine together, it becomes a process.
 - For example : sequence, flowchart, Click, Write line and Message box etc

What is a Sequence?

- A Sequence is a group of logical steps. Each step represents an action or a piece of work.
- Flowchart and State machine can have several sequences.

Flowcharts

- Flowchart is generally used for complex business processes
- It provides decision-making facilities and can be used for both small and large projects
- A Flowchart provides multiple branching logical operators to make decisions.



Building 'Hello World' Robot

Print "Hello" by using Sequence activity

Algorithm:

Step 1: START

Step 2: Add Message Box activity and write " Hello" into Flow Chart Activity

Step 3: Add Write Line activity and write "Hello" in Flow Chart Activity

Step 4: STOP

Activities involved

- ✓ "Sequence" and "Assign" activity.
- ✓ "Message Box" activity.
- ✓ "Write Line" activity.

Print "Hello" by using Flowchart activity

Algorithm:

Step 1: START

Step 2: Add **Message Box activity** and write " Hello" into Flow Chart Activity

Step 3: Add Write Line activity and write "Hello" in Flow Chart Activity

Step 4: STOP

Activities involved

- ✓ **"Flowchart"** activity.
- ✓ **"Message Box"** activity.
- ✓ **"Write Line"** activity.

Variables



Name

Give a proper name to your variables



Variable Type

Specify the data type for your variable



Array

Integer

String

Generic

Boolean



Scope

Specify the scope of your variable



Default

Specify if your variable should have a default value or not.





- **String** – a. All strings must be placed in double quotation.
Default value = “ ”
- **Bool** – aTrue | False or true | false
- **Int32** – a. Default value = 0
- **Array** – a. String array = {“value1”, “ Value2”, “ Value3”}
Int array = {12,13,14,5}
- **DateTime** a. – System date time.
- **DataTable** a. To store structured data in rows and columns.
- **Generic**
Int32, String, Bool , DateTime
- **List** – Same as array

GenericValue is a type of variable that is particular to UiPath Studio

- It 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

| Name | Variable type | Scope | Default |
|------------------------|---------------|----------|------------------------------|
| firstName | String | Sequence | <i>Enter a VB expression</i> |
| lastName | String | Sequence | <i>Enter a VB expression</i> |
| <i>Create Variable</i> | | | |
| | | | |

VariablesArgumentsImports

100%

Arguments panel

- These Arguments are used for interacting with different workflows by exchanging data between them.
 - Either giving the value to some
 - Workflow or receiving the value from another workflow.
 - **In:** When we have to receive the value from another workflow.
 - **Out:** This is the current value if we have to send the value to a workflow.
 - **In/Out:** This specifies both; it can take or receive the value.
 - **Property:** This specifies that it is not being used currently:

Variables vs. Arguments

Variables

- only used inside a single workflow file(.xaml) in your project
- Types of Variable: Boolean, Array, Integer, text., Datatable
- It is used to pass it other activities.
- You can change the variable name multiple times

Arguments

- used to pass data from one workflow file to another
- have a specific direction – In, Out, In/Out
- Type of Argument: Argument, Property, Direction, Default etc.

Variable

| Name | Variable type | Scope | Default |
|------|---------------|----------|------------------------------|
| ABC | Int32 | Sequence | <i>Enter a VB expression</i> |

Arguments

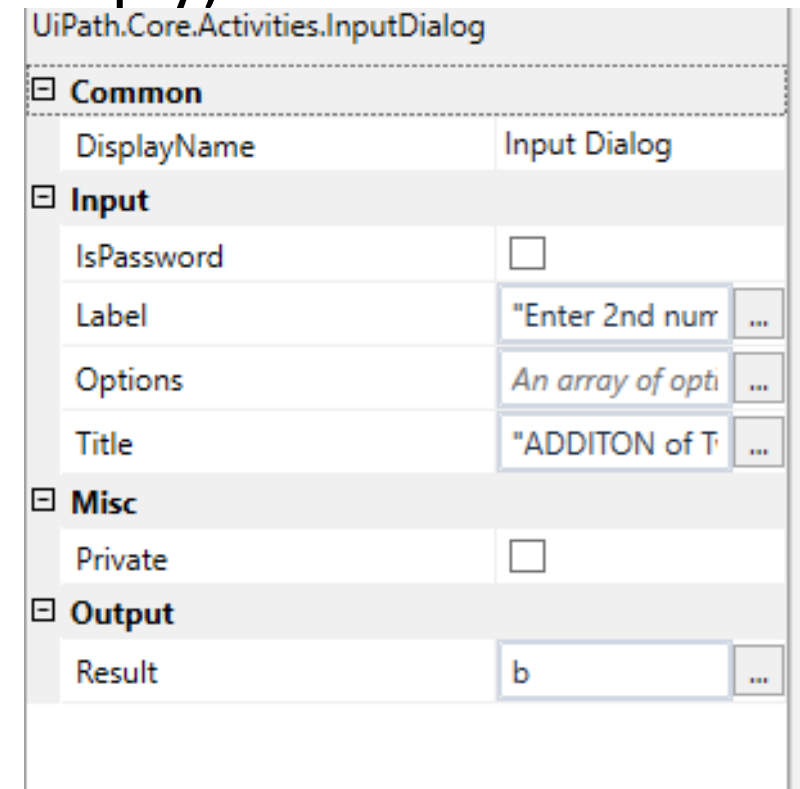
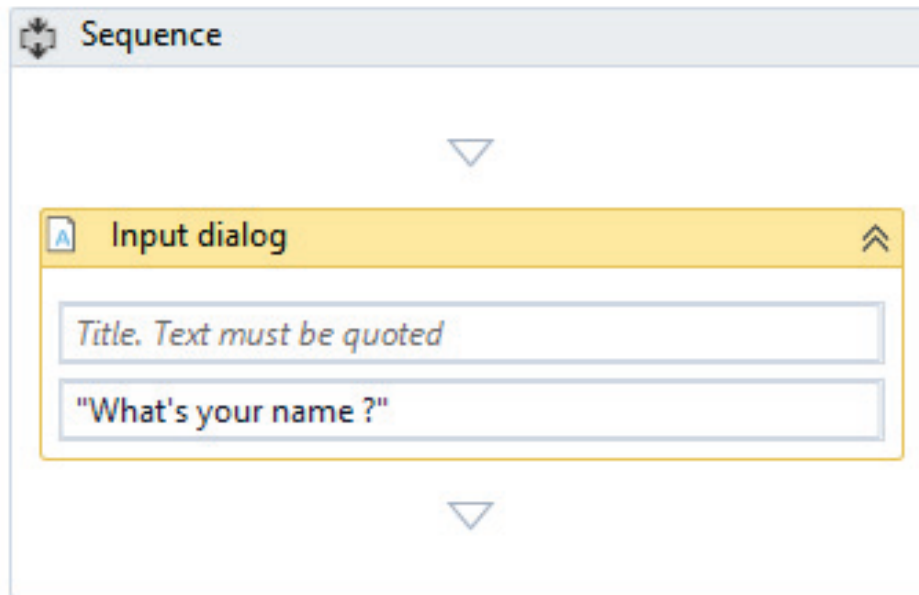
| Name | Direction | Argument type | Default value |
|------|-----------|---------------|------------------------------------|
| XYZ | In | String | <i>Enter a VB expression</i> |
| ABC | Out | String | <i>Default value not supported</i> |

Reading name and printing onto screen using flow chart

- **Step 1:** START
- **Step 2:** Open Flowchart activity.
- **Step 3:** Ask for the username in an **Input dialog**
- **Step 3:** Display the username in a **Message box**
- **Step 4:** STOP

Input dialog activity.

- Dialog box that appears with a message or a question, in response to which the user is required to put in his or her reply):

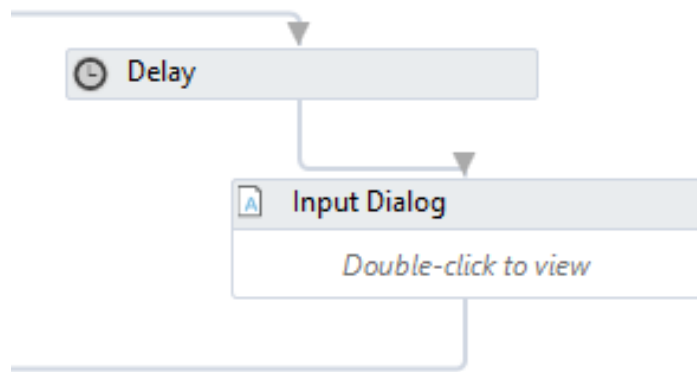


Delay activity

- The **Delay** activity, as the name suggests, is used to delay or slow down an automation by pausing it for a defined period of time
- It is in the **hh:mm:ss** format.

This activity plays a significant role when we need a waiting period during automation

Ex: a waiting period required for a particular application to open.



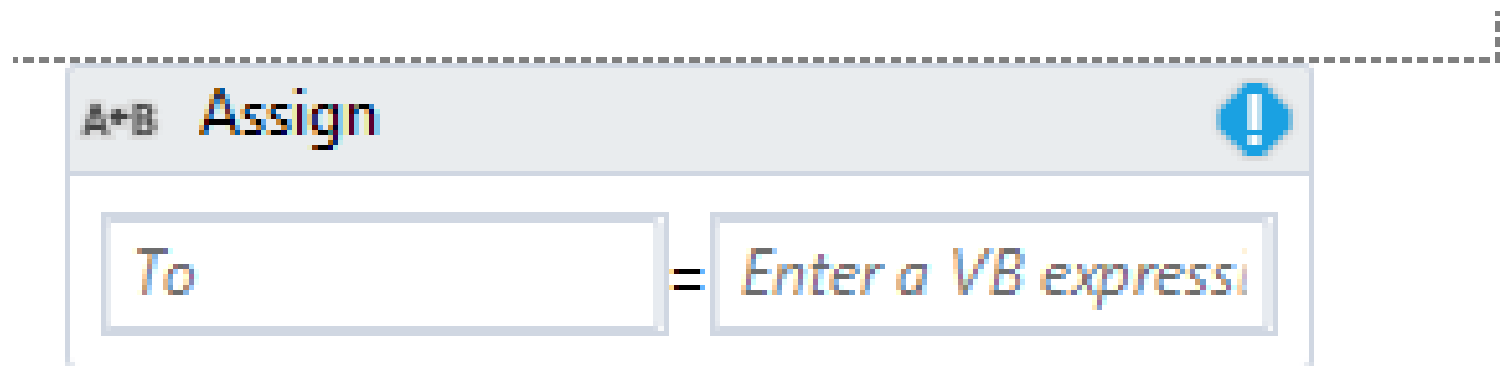
| Properties | |
|--|--------------------------|
| System.Activities.Statements.Flowchart | |
| Common | |
| DisplayName | Read |
| Misc | |
| Private | <input type="checkbox"/> |
| ValidateUnconnectedNodes | <input type="checkbox"/> |

Mouse activities

- Three mouse activities in UiPath Studio:
 - Click activity
 - Used to click on a UI element on the screen
 - Double-click on the **Click** activity and Click on **Indicate on screen** and indicate the UI element you want to click on.
 - Double-click activity
 - The Double Click activity is similar to the Click activity
 - It just performs the double-click action
 - You have to use the Double click activity instead of the Click activity and indicate the UI element.
 - *Hover activity*
 - The **Hover** activity is used to hover over a UI element

Assign activity

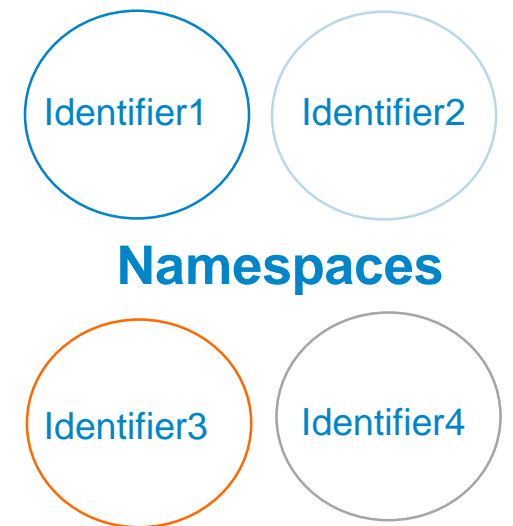
- The **Assign** activity is used to designate a value to the variable.
- The Assign activity can be used for different purposes,
 - incrementing the value of a variable in a loop
 - To get results of a sum, difference, multiplication, or division of variables and assigning it to another variable.



What are **Namespaces**?

- **Namespaces** are containers created to hold a logical grouping of unique names (identifier).
- A namespace ensures that all of a given set of objects have unique names so that they can be easily identified.
- 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

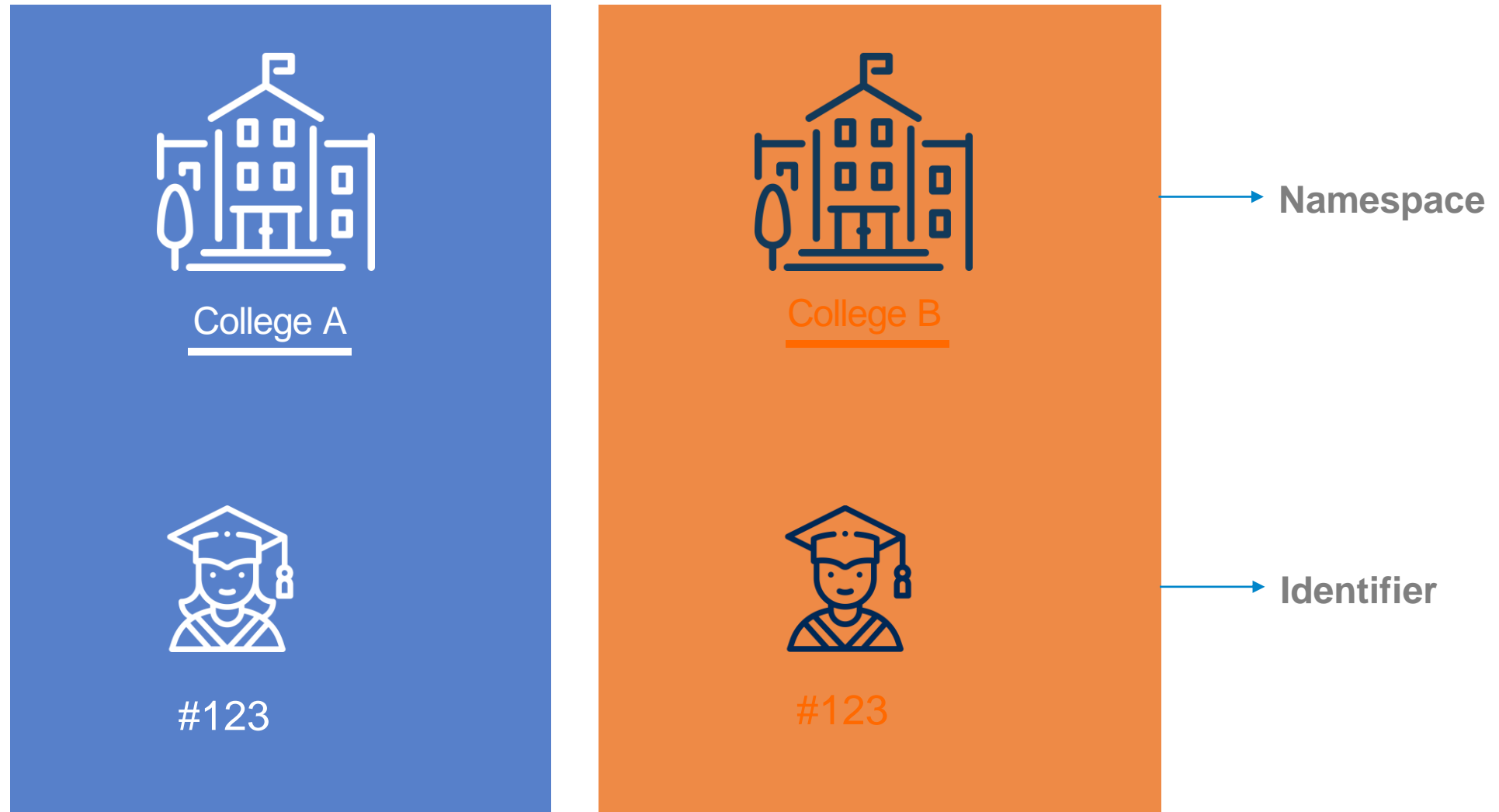
It is also used to avoid name collisions between multiple identifiers that share the same name



Namespaces - Scenario

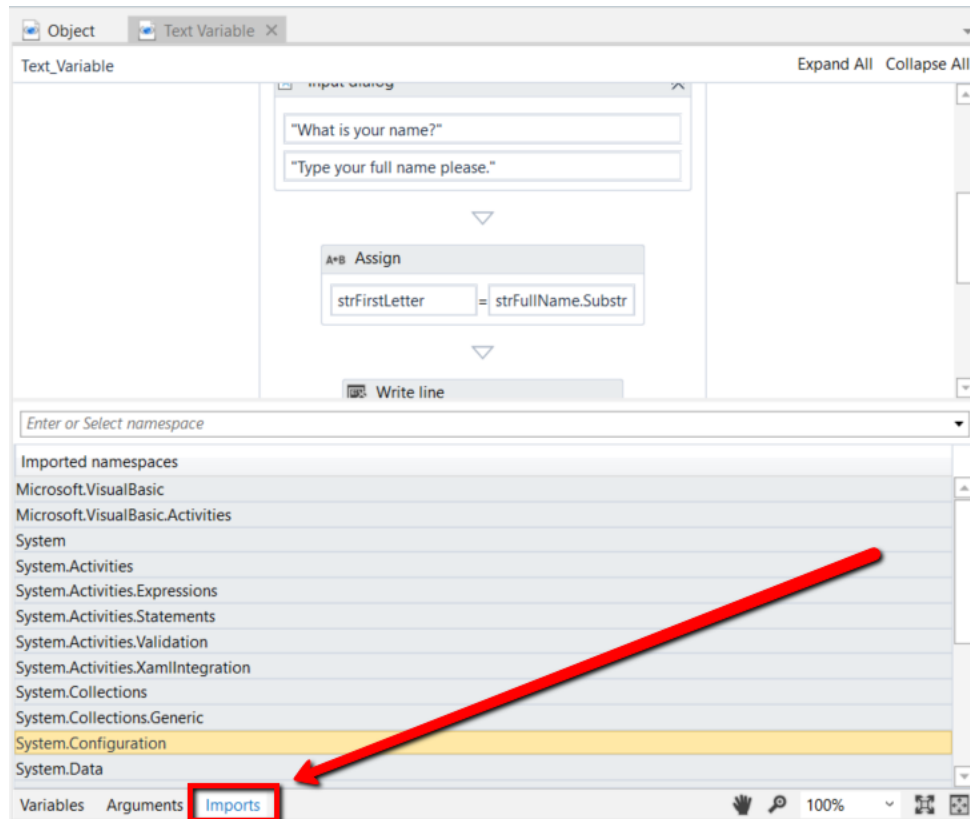
Think of it this way...

There are two colleges, A and B and they both need to assign a unique ID to their students.



Namespaces in UiPath

VB.NET namespaces in UiPath Studio represent containers that store different types of data.



All imported namespaces are displayed in the Imports panel.