LECTURE 08. EXCEL AND DATA TABLES

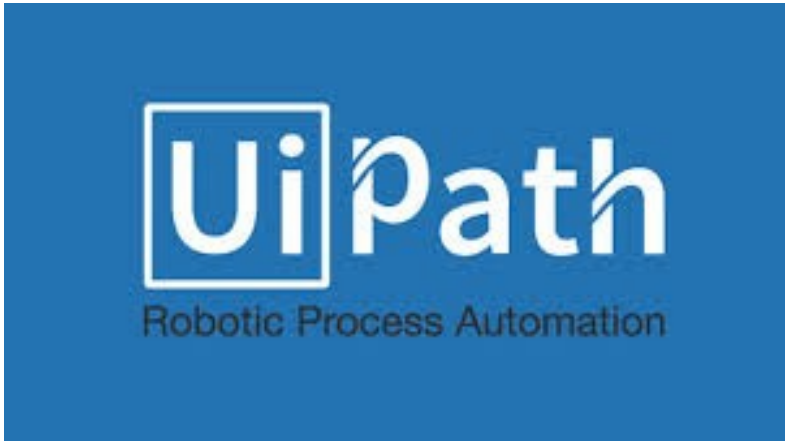
Robotic Process Automation
[21 November 2023]

Elective Course, 2023-2024, Fall Semester

Camelia Chisăliţă-Creţu, Lecturer PhD
Babeş-Bolyai University

Acknowledgements

This course is presented to our Faculty with the support of UiPath Romania.



Contents

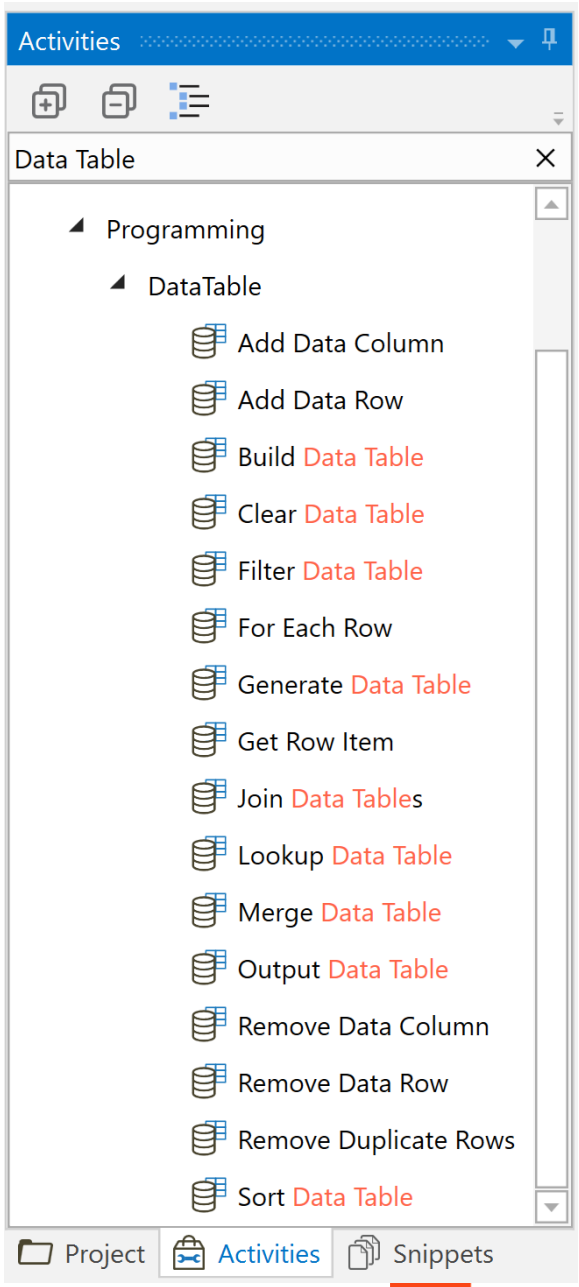
Data Table. Details

- **Data Table** variable type characteristics in UiPath:
  - **a data structure with flexible length;**
  - it is used to store data in the form of rows and columns, similar to Excel;
  - it enables:
    - to migrate data from one database to another;
    - to process data (filter, create new data, etc.);
  - it can be iterated by using a **For Each Row** activity;
  - ways to create a data table:
    - web **data scrapping** (see **Lecture 05. UI Interactions**);
    - building it from scratch:
      - **Build Data Table**, **Add Row**, **Add Data Column** activities;
    - processing data from **Excel sheets** and **.csv** files:
      - various activities: **Excel Application Scope, Read/Write Range, Read/Write Cell, Output Data Table, Select, Filter, etc.;**

Row/ Column	First	Last	Club Member
0	"John"	"Doe"	Yes
1	"Jane"	"Doe"	No
2	"Jane"	"Doe"	Yes
3	"John"	"Doe"	No

Data Table. UiPath Activities

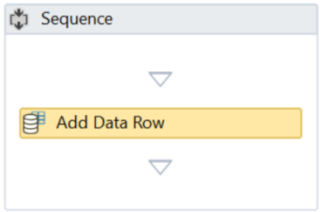
- UiPath provides a series of activities for **Data Table** variables;
- these activities are found under the **Programming** library, **Data Table** section.

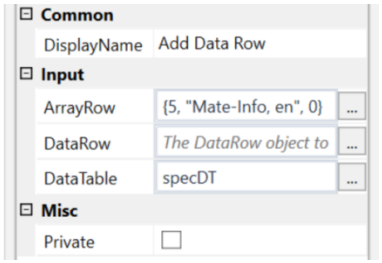


see **Demo4 – BuildDataTable**

Add Data Row Activity. Details

- **Add Data Row** activity
  - allows to add a new data row into a data table;
- relevant properties:
  - **[input] DataTable** = **DataTable** variable
  - **[input] Array Row** = **Array** variable
    - an array consisting of the values that will be added to the data table;
    - each value within the array has the data type that corresponds to the column in the data table;
      - E.g.: {5, "Mate-Info, en", 0} or {(SpecDataTable.Rows.Count+1).ToString,name};
  - **[input] Data Row** = **DataRow** variable
    - if such object is provided then the **Array Row** property is ignored.



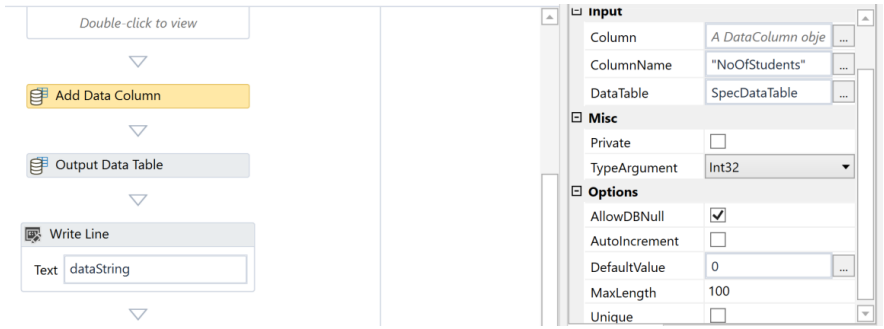


see **Demo4 – BuildDataTable**

UiPath

Add Data Column Activity. Details

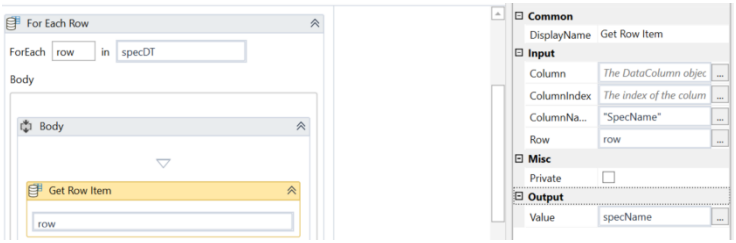
- **Add Data Column** activity
  - allows to change the structure of the data table by adding a new column;
- relevant properties:
  - **[input] DataTable** = **DataTable** variable
  - **[input] Column Name** = **String**
    - E.g.: “NoOfStudents”
  - **[input] TypeArgument** = **Type** variable
    - the data type for the new column that is added to the data table structure;
  - **[options] AllowDBNull, AutoIncrement, DefaultValue, MaxLength, Unique**
    - attributes of the new column that can be set.



see **Demo4 – BuildDataTable**

Get Row Item Activity. Details

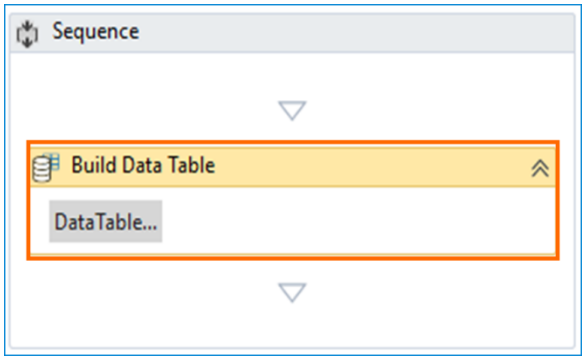
- **Get Row Item** activity
  - allows to get the value of a column from a specific row in the data table;
  - is usually placed in a **For Each Row** activity that allows to iterate over a data table;
- relevant properties:
  - **[input] Row** = **DataRow** variable
    - a variable used to iterate the data table;
  - **identification criterion (one option can be chosen)**
    - **[input] Column** or **[input] ColumnIndex** = 1 or
      - the index of the column whose value is extracted;
      - the first column in the data table has the index 0;
    - **[input] ColumnName** =“SpecName”
      - the name of the column used as sorting criterion;
      - it is the preferred identification criterion compared to **Index**;
  - **[output] Value** = **GenericValue** variable.

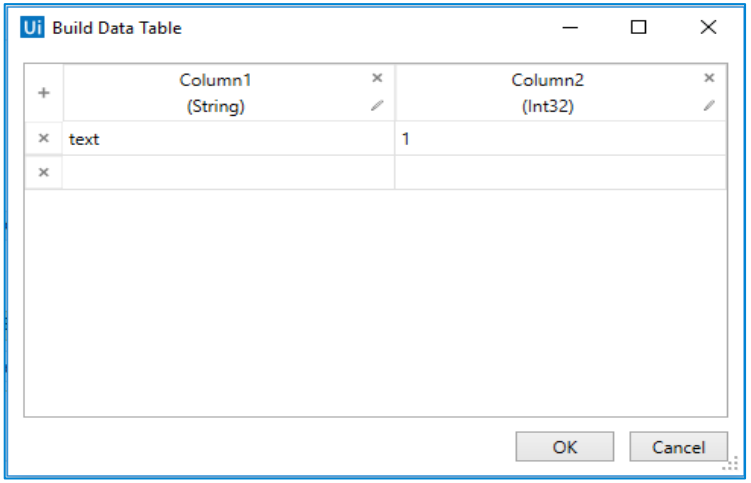


see **Demo4 – BuildDataTable**

Build Data Table Activity. Details

- **Build Data Table** activity
  - is used when a user has to store the data manually inside the data table;
  - allows to reorder existing columns;
- Steps to use the activity:
  1. drag and drop **Build Data Table** activity inside the sequence;
  2. click on the **Data Table** to customize the data table;

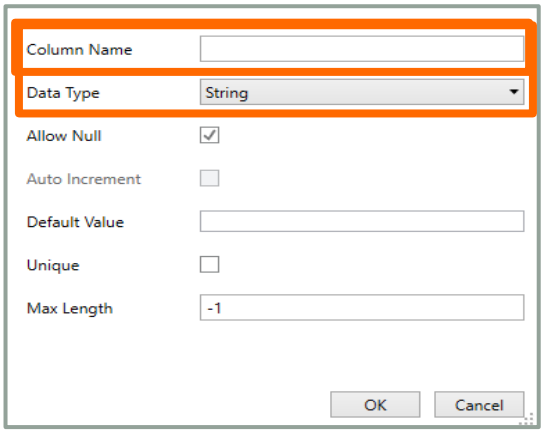


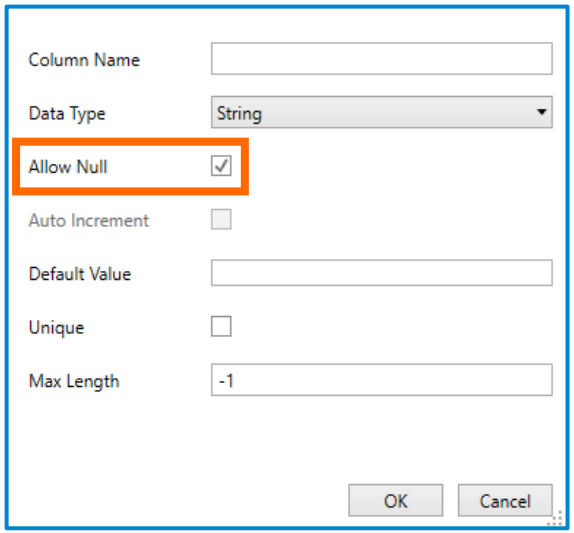


see **Demo4 – BuildDataTable**

Build Data Table Activity. Steps (1)

- Steps to use the activity:
  1. drag and drop **Build Data Table** activity inside the sequence;
  2. click on the **Data Table** to customize the data table;
  3. click on the "+" sign in order to add a column button;
     - set the name and data type of the column;
     - if **Allow Null** is checked, it is not compulsory to have additional data in the column;

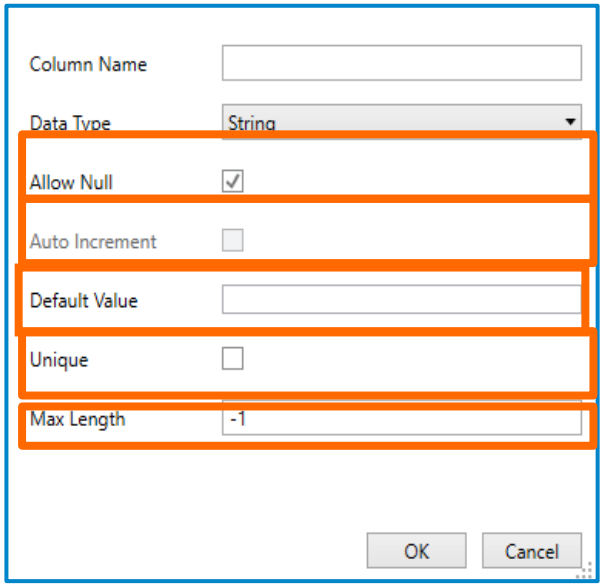




see **Demo4 – BuildDataTable**

Build Data Table Activity. Steps (2)

- Steps to use the activity:
  5. Other attribute can be set:
     - **Default value:** if the column is blank, then it will automatically take default value inside it;
     - **Max Length:** the number of characters allowed for the column; if the user does not want to apply the length of maximum data, then the -1 is set;
     - **Unique:** if the dataset is selected for a specific value then the user has to create unique data in the data table;
     - **Auto Increment:** when the user sets the **Data Type** in the form to **Int32** then the checkbox is enabled; the data automatically increase by 1 every time a new row is added.

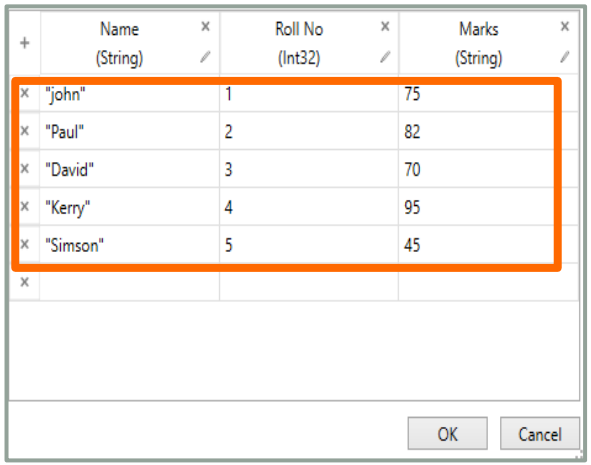


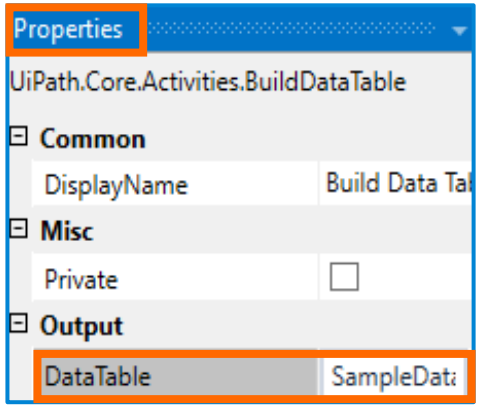
see **Demo4 – BuildDataTable**

UiPath

Build Data Table Activity. Steps (3)

- Steps to use the activity:
  6. add data inside the data table;
  7. click on the **Properties** tab of the built **Data Table**;
     - create the variable in the **Output** section with the name “Sample Data.”

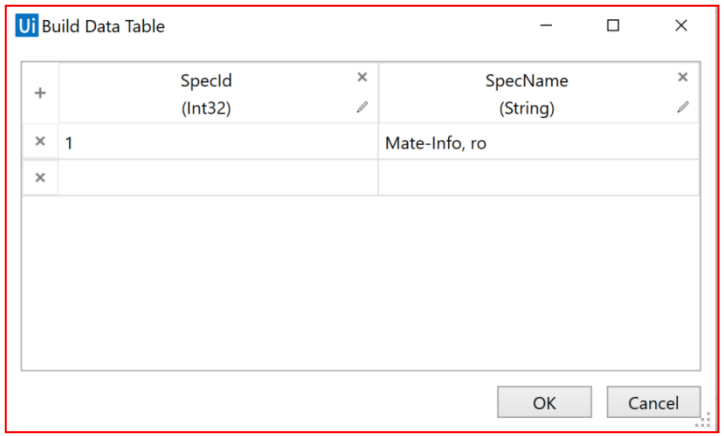




see **Demo4 – BuildDataTable**

Demo 4. Build Data Table

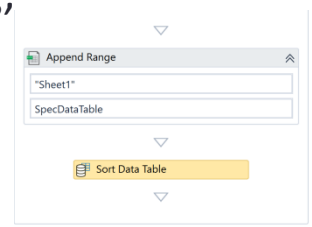
- Use the **Build Data Table, Add Data Row** and **Add Data Column** activities to build the data table with the following data:
  - **SpecId** (int);
  - **SpecName** (string);
  - **NoOfStudents** (int);
- the first two data fields are added by using **Build Data Table** activity, while the third one is added by using **Add Data Column** activity;
- the values for the first two data fields are read from the standard input, while the values for the third data field is computed (in a subsequent demo).

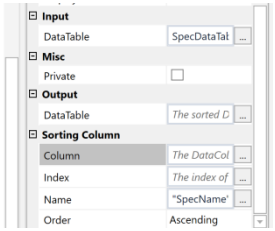


see **Demo4 – BuildDataTable**

Sort Data Table Activity. Details

- **Sort Data Table** activity
  - allows to sort the data from a given data table considering some column details;
  - can be placed inside or outside an **Excel Application Scope** activity, as long as the scope of the **DataTable** variables used are set to include the **Sort Data Table** activity;
- relevant properties:
  - **[input] DataTable** = **DataTable** variable *and* **[output] DataTable** = **DataTable** variable
    - the resulting data table after performing sorting;
    - the output can be the input data table;
  - **sorting criterion (one option can be chosen)**
    - **[input] Column** or **[input] Index** = 1 or
      - the index of the column used as the sorting criterion;
      - the first column in the data table has the index 0;
    - **[input] Name** =“year”
      - the name of the column used as the sorting criterion;
      - it is the preferred sorting criterion compared to **Index**;
  - **[input] Order** = **Ascending/Descending.**

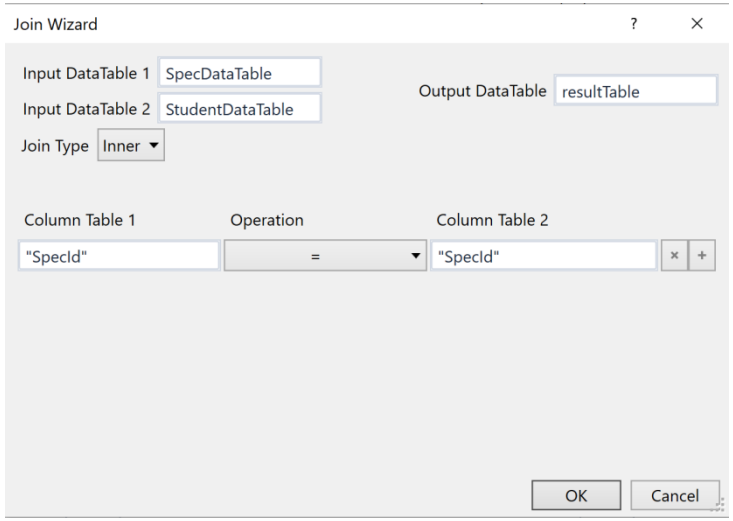




see **Demo5 – ExcelActivities**

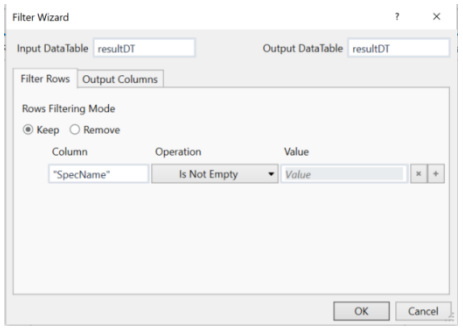
Join Data Table Activity. Details

- [**Application Integration** activities --> **DataTable** section]
- **a join operation** allows
  - to combine data from multiple tables based on certain conditions (usually when field values are equal);
- there are several types of join actions: **left join, inner, full;**
- **Join Data Table** activity
  - allows to build new data starting from multiple data tables;
- relevant properties:
  - **[input] Input DataTable 1** = **DataTable** variable
  - **[input] Input DataTable 2** = **DataTable** variable
  - **[output] Output DataTable** = **DataTable** variable
  - **[input] Type** = **Inner/Left/Full** join
  - **[input] Condition**
    - uses logical operators (and, or).



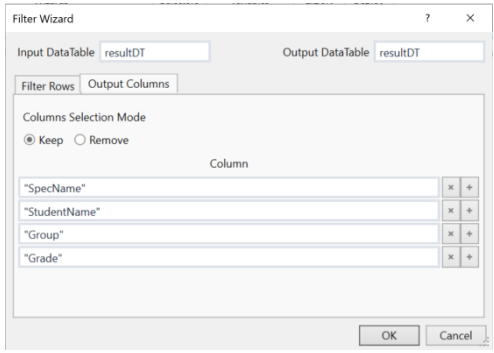
see **Demo5 – ExcelActivities**

Filter Data Table Activity. Details



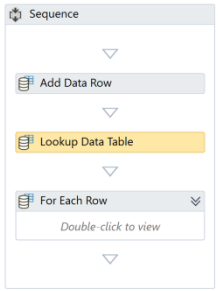
see **Demo5 – ExcelActivities**

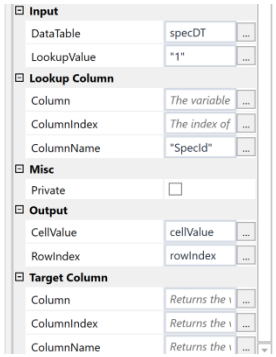
- **Filter Data Table** activity
  - provides a wizard that allows to configure the filter actions;
- relevant properties:
  - **[input]** **DataTable** = **DataTable** variable
  - **[output]** **DataTable** = **DataTable** variable
    - the input and output data table can be the same **DataTable** variable;
  - **[options]** **Filter Rows, Output Columns**
    - attributes for the filtering actions are set based on the wizard options:
      - **Filter Rows:**
        - allows to indicate the filtering conditions;
        - for numeric columns UiPath convert the value to **Double**;
          - E.g.: “SpecId”, “=“, 3.00 even if the data type of **SpecId** field is **Int32**;
      - **Output Columns:**
        - allows to change the structure of the output data table by instantiating a **DataTable** variable and populating it with data, accordingly.



Lookup Data Table Activity. Details

- **Lookup Data Table** activity
  - looks for a specified value into the given data table;
  - can be customized in various ways in order to provide specific found value and/or row index only;
- relevant properties:
  - **[input]** **DataTable** = **DataTable** variable
  - **[input]** **LookupValue** = **String** variable
    - the value searched for into the data table;
  - **[lookup column]** **Column** or **ColumnName** or **ColumnIndex**
    - one of available options can be chosen;
  - **[output]** **CellValue** = **GenericValue** variable
    - the found value that was found on the looked for column at a returned row index;
  - **[output]** **RowIndex** = **Int32** variable
    - the row index of the returned value on **CellValue** property;
  - **[target column]** **Column** or **ColumnName** or **ColumnIndex**
    - if set, it returns in **CellValue** the found value, otherwise **CellValue=null**.





see **Demo5 – ExcelActivities**

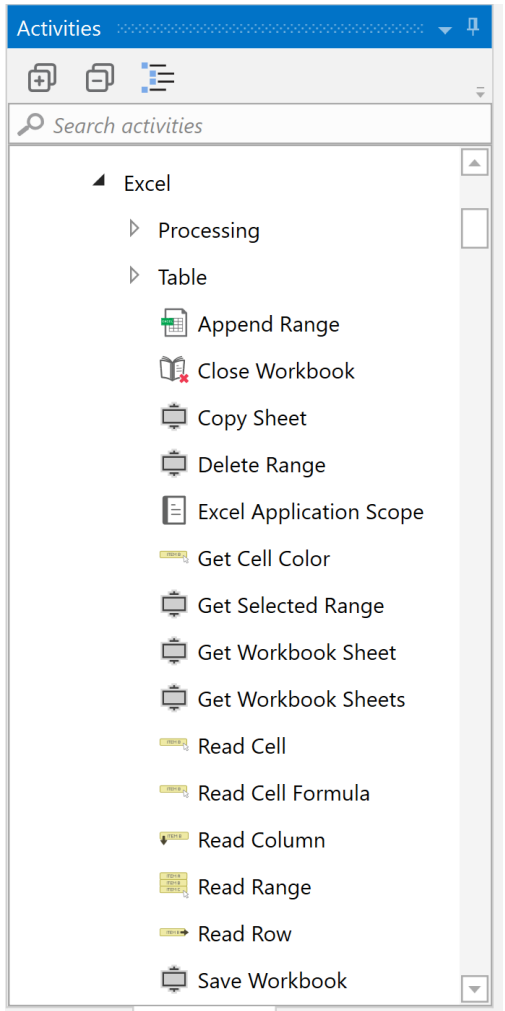
Excel and Data Table. Details

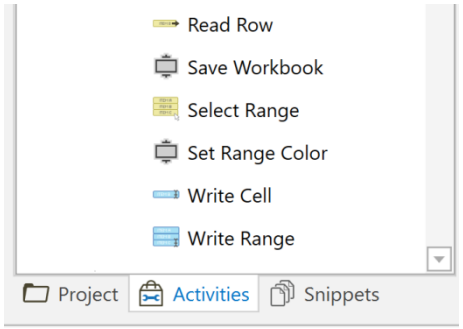
- **Excel** is an application whereas the **Data Table** variables in UiPath Studio mimic Excel operations;

Excel	Data Table
• the Excel application enables spreadsheet activities at cell and spreadsheet level.	• Data Table is a **prototype of spreadsheet** that presents data in rows and columns.

Excel Integration. UiPath Activities

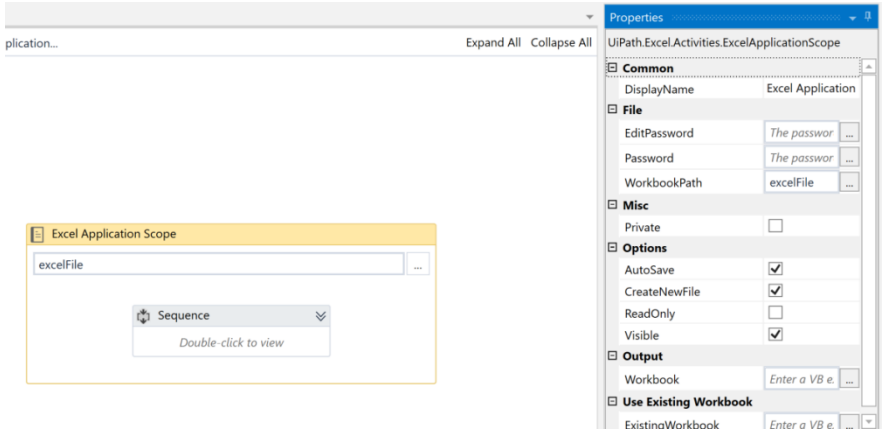
- UiPath provides a series of activities to work with **Excel** files;
- rhese are activities that are already integrated ---> being found under the **Application Integration** library, **Excel** section.





Excel Application Scope Activity. Details (1)

- **Excel Application Scope** activity
  - is a container for other activities that work on the same Excel file;
- relevant properties:
  - **WorkbookPath** = *path*+“filename.xlsx”
    - *path* is relative to the location of the current project;
  - **Visible** = **checked/unchecked**
    - **checked** = it reads the file using MS Excel;
    - **unchecked** = it performs the operations internally, directly on the file;



see **Demo5 – ExcelActivities**

Excel Application Scope Activity. Details (2)

Checked = Use Excel Application

- it requires MS Excel to be installed;
  - the actions are performed through MS Excel application;
- multiple processes can use the same file;
- visible real-time changes into the file;
  - recommended for:
    - debugging;
    - checking the progress of workflow.

Unchecked = Direct Access

- it does note require MS Excel;
- only one process can use the file;
- it works only for .xlsx file format only.

Demo 5. Excel and Data Table Activities

- **Consider the Specs.xlsx file with the following data columns:**
  - **SpecId (int), SpecName (string), NoOfStudents (int).**
- **Consider the Students.xlsx file with the following data columns:**
  - **SpecId (int), StudentName (string), Group (int), Lab (int), Project (int), Grade (int).**

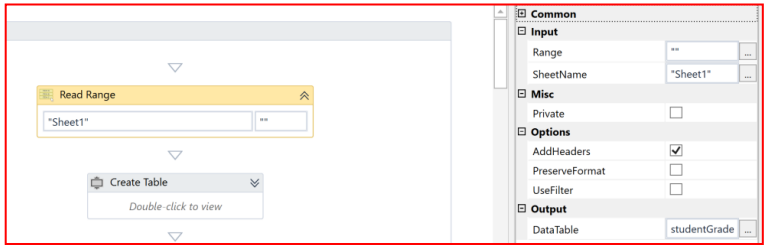
- **Design a process that allows to:**

1. **Compute the final Grade for each student, based on the Lab an Project grades;**
2. **Build a ranking file Ranking.xlsx consisting of the following details: SpecName (string), StudentName (string), Group (int), Grade (int), ordered descending by their grade;**
3. **Build a classbook file Classbook.xlsx with the following details:**
   - **For each specialization there a distinct spreadsheet, that consists of the followings: StudentName (string), Group (int), Grade (int), ordered by group and student names;**
4. **Update the last data column from Spec.xlsx file with the number of students from the corresponding specialization.**

see **Demo5 – ExcelActivities**

Read Range Activity. Details

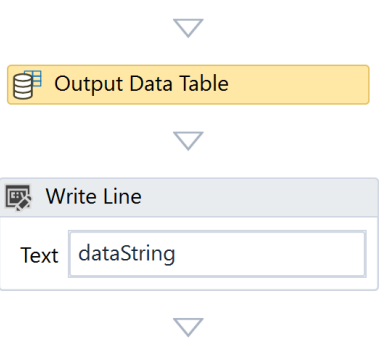
- **Read Range** activity
  - is reads a portion of the Excel file and stores it intro a **Data Table** variable, used for later operations;
- relevant properties:
  - **[input] SheetName**= “Sheet1”
    - the name of the spreadsheet that will be read from the Excel file;
  - **[input] Range** = “” or “A1:L10”
    - the cell range that will be read from the spreadsheet;
    - “” is the **default range**, i.e., all the data from the spreadsheet;
  - **[input] AddHeaders** = checked/unchecked
    - if checked, the output data table column can be accessed by their names;
  - **[output] DataTable** = **DataTable** variable.

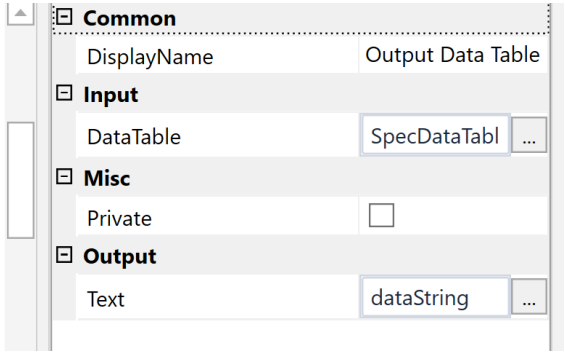


see **Demo5 – ExcelActivities**

Output Data Table Activity. Details

- **Output Data Table** activity
  - converts the information from the data table to **String** value in order to print it properly;
  - is not meant to print the data table;
  - is used together with a **Write Line** or **Message Box** activity;
- relevant properties:
  - **[input] DataTable** = **DataTable** variable
    - a data table for which data will be converted to a **String** value;
  - **[output] Text** = **String** variable
    - a variable that stores the data converted to a **String** value.

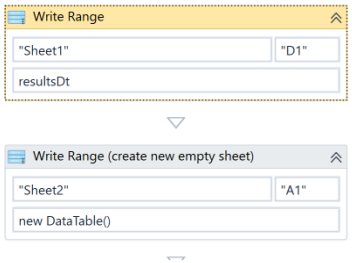


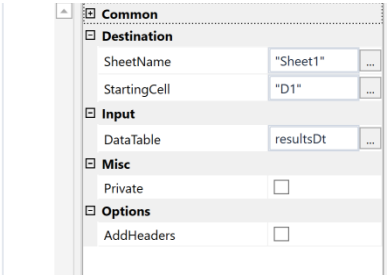


see **Demo5 – ExcelActivities**

Write Range Activity. Details

- **Write Range** activity
  - allows to write the content of a data table into the file with the specified name inner most **Excel Application Scope** activity;
  - if the file does not exist, it will be created;
  - if there is data on the output file, it will be overwritten, i.e., similar to a Paste operation that starts from the **StartingCell**;
- relevant properties:
  - **[input] DataTable** = **DataTable** variable
  - **[input] SheetName**= “Sheet1”
    - if it does not exists, a new one is created;
  - **[input] StartingCell** = “A1”
    - the cell range that the writing will start from;
  - **[input] AddHeaders** = checked/unchecked
    - if checked, the written data in the file will have column headers written as well.

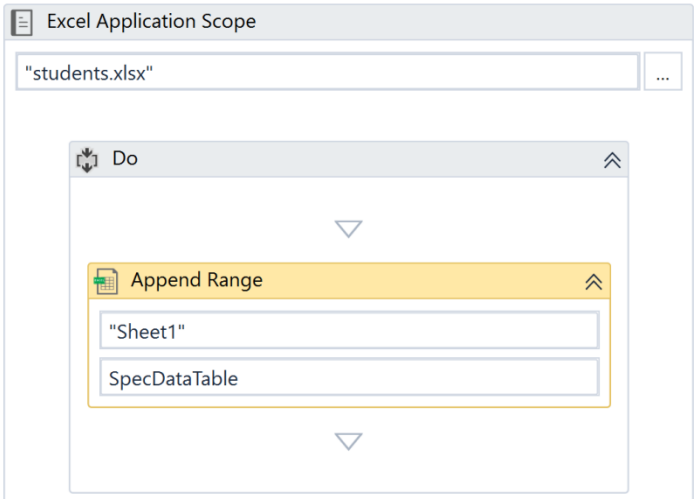


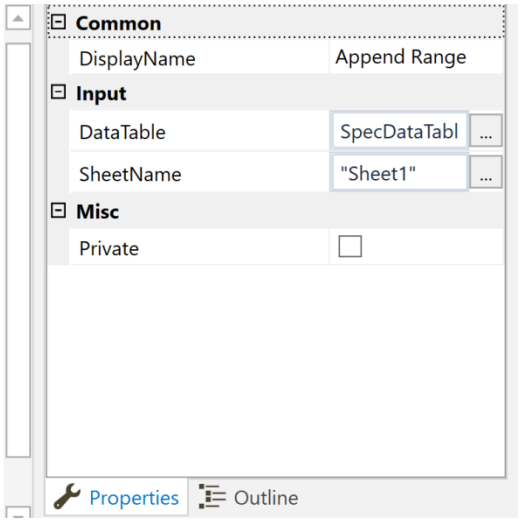


see **Demo5 – ExcelActivities**

Append Range Activity. Details

- **Append Range** activity
  - adds the given data after all existing data into the Excel file, without overwriting;
  - is always enclosed within an **Excel Application Scope** activity;
- relevant properties:
  - **[input] DataTable** = **DataTable** variable
  - **[input] SheetName**= “Sheet1”

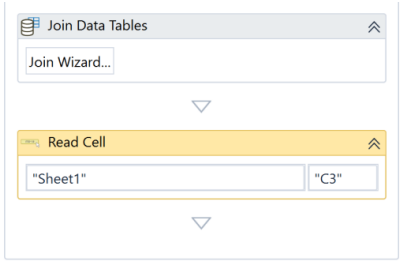


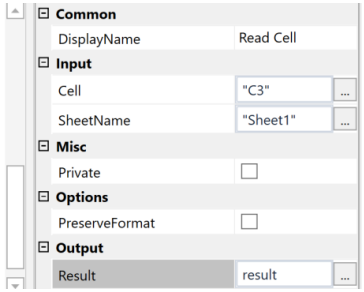


see **Demo5 – ExcelActivities**

Read Cell Activity. Details

- **Read Cell** activity
  - allows to extract the data from a designated cell and save it into a variable for later use;
  - is always enclosed within an **Excel Application Scope** activity;
- relevant properties:
  - **[input] Cell** = “C2”
    - a string that indicates the cell to read from;
  - **[input] SheetName**= “Sheet1”
    - the name of the spreadsheet the data will be read from;
  - **[output] Result** = **GenericValue** variable
    - the name of the variable used to store the extracted data.





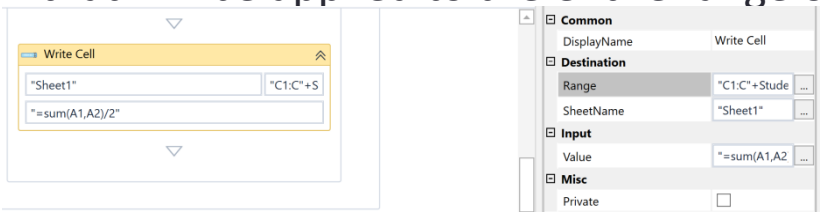
see **Demo5 – ExcelActivities**

UiPath

Write Cell Activity. Details

- **Write Cell** activity
  - allows to write the given data into the specified cell of the Excel file indicated by the inner most **Excel Application Scope** activity;
  - is always enclosed within an **Excel Application Scope** activity;
- relevant properties:
  - **[input] Range**= “C2” or "C1:C"+StudentDataTable.Rows.Count.ToString
    - a string that indicates the cell where to write to;
  - **[input] SheetName**= “Sheet1”
    - the name of the spreadsheet where the data will be written to;
  - **[input] Value** = **GenericValue** variable *or*

    a formula-based expression, e.g., "=sum(A1,A2)/2"
    - the name of the variable that stores the value to be written into the file or a formula-based expression that will be applied to the entire range of cells.

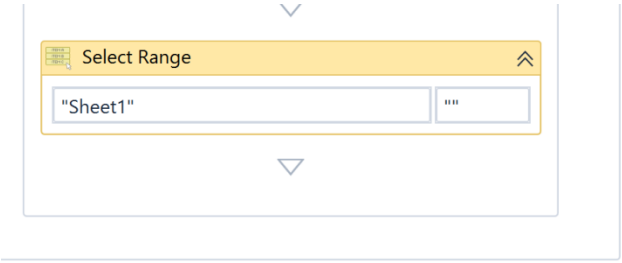


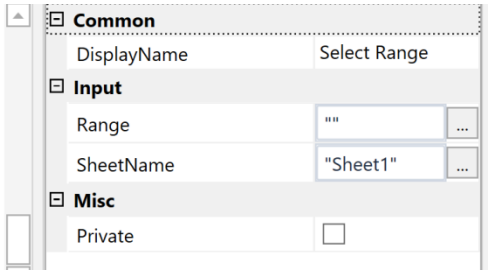
see **Demo5 – ExcelActivities**

UiPath

Select Range Activity. Details

- **Select Range** activity
  - allows to select the cells found in a specified range;
  - **on its own it does not have a direct outcome**, i.e., it is usually combined with a **copy**, **delete**, or other operation, **directly on MS Excel**;
  - is always enclosed within an **Excel Application Scope** activity;
- relevant properties:
  - **[input] Range** = “C2:C4”
    - a string that indicates the cell range to be selected;
  - **[input] SheetName**= “Sheet1”
    - the name of the spreadsheet that contains the selected cell range.





see **Demo5 – ExcelActivities**

UiPath

Demo 5. Excel and Data Table Activities

- **Consider the Specs.xlsx file with the following data columns:**
  - **SpecId (int), SpecName (string), NoOfStudents (int).**
- **Consider the Students.xlsx file with the following data columns:**
  - **SpecId (int), StudentName (string), Group (int), Lab (int), Project (int), Grade (int).**

- **Design a process that allows to:**

1. **Compute the final Grade for each student, based on the Lab an Project grades;**
2. **Build a ranking file Ranking.xlsx consisting of the following details: SpecName (string), StudentName (string), Group (int), Grade (int), ordered descending by their grade;**
3. **Build a classbook file Classbook.xlsx with the following details:**
   - **For each specialization there a distinct spreadsheet, that consists of the followings: StudentName (string), Group (int), Grade (int), ordered by group and student names;**
4. **Update the last data column from Spec.xlsx file with the number of students from the corresponding specialization.**

see **Demo5 – ExcelActivities**

Next lecture…

- **Lecture 9**
  - **Image and Text Automation**

References

- UiPath Docs - https://docs.uipath.com/
- UiPath Studio Docs - https://docs.uipath.com/studio/standalone/2023.4
- UiPath Forum - https://forum.uipath.com/
- UiPath Academy - https://academy.uipath.com/