Input Datasets

To access data from the **Input data source [N]** ports, the `InputTables[]` tuple is used, where the data source of a port is accessed by its number:

`InputTables[N]`,

where N is the port number (index). The first port has an index of 0.

The elements of the tuple are of type `DataSourceClass` (see Python: Full API Description). Since the first port is present in the **Python** node by default, a separate `InputTable` class is provided for accessing its data.

Data Source Properties

▼ Columns

Contains a read-only iterable collection of columns. Implements the `Mapping` and `Sequence` protocols. Returns a value of type `ColumnsClass`. Elements can be accessed via bracket notation [] by name and by index. The elements are of type `InputColumnClass`, inherited from `ColumnClass`, which implements the `Sequence` protocol (see Full API Description).

▼ ColumnCount

Contains the read-only number of columns. Returns a value of type `int`.

▼ RowCount

Contains the read-only number of rows. Returns a value of type `int`.

Data source methods

▼ Get

Get(row, col)

- `row`: The row index. Accepts a value of type `int`.
- `col`: The column index or name. Accepts values of type `int` or `str`.

> The method returns the value of the specified column in the specified row. The returned value can be of the following types: `bool`, `int`, `float`, `str`, `datetime.datetime`, `None`.

▼ GetColumn

> **GetColumn(col)**
>
> - `col`: The column index or name. Accepts values of type `int` or `str`.
>
> The method returns a column from the input set. The returned value is of type `InputColumnClass`, inherited from `ColumnClass`, which implements the `Sequence` protocol (see Full API Description).

▼ IsNull

> **IsNull(row, col)**
>
> - `row`: The row index. Accepts a value of type `int`.
> - `col`: The column index or name. Accepts values of type `int` or `str`.
>
> The method returns a Boolean value `true` if the column in the specified row has a missing value. Otherwise, it returns `false`.

Using the builtin_pandas_utils module

The builtin_pandas_utils module implements the `to_data_frame` method for creating a `pandas.DataFrame` from an input port's dataset (see examples).

▼ to_data_frame

> **to_data_frame(table, useNullableArrays)**
>
> - `table`: A reference to the input set (a value of type `DataSourceClass`).
> - `useNullableArrays`: If the argument's value is `True`, the table is converted to a `pandas.DataFrame` without changing the data types of fields that contain missing values. This is an optional argument. The default value is `False`.

Examples

```
from builtin_data import InputTable, InputTables
from builtin_pandas_utils import to_data_frame
import numpy as np

inputTable0 = InputTables[0]                    #Data source from port #1
inputTable1 = InputTables[1]                    #Data source from port #2
colOutlook0 = inputTable0.Columns["OUTLOOK"]    #Get a reference to a
column by name
colOutlook1 = inputTable1.Columns[1]            #Get a reference to a
column by index

#Outputting column property values
for column in InputTable.Columns:
    print("Index: ", column.Index)
    print("Name: ", column.Name)
    print("DisplayName: ", column.DisplayName)
    print("DataType: ", column.DataType)
    print("DataKind: ", column.DataKind)
    print("UsageType: ", column.UsageType)
    print("DefaultUsageType: ", column.DefaultUsageType)
    print("RowCount: ", column.RowCount, "\n")

#Outputting column values
for index, value in enumerate(colOutlook0):
    print("Index {}, value {}".format(index, value))

#Reading values from the input table using the Get method
for i in range(InputTable.RowCount):
    #Output the values of the column with index 0
    print("InputTable.Get({}, 0) = {}".format(i, InputTable.Get(i, 0)))
    #Output the values of the column named "CLASS"
    print("InputTable.Get({}, 'CLASS') = {}".format(i, InputTable.Get(i,
"CLASS")))

#Getting an array of values from the "CLASS" column
arrayOfColumnValues = np.array(InputTable.Columns["CLASS"])
print(arrayOfColumnValues)

#Creating a pd.DataFrame from the input set
```

```
data = to_data_frame(InputTable)
print(data)
```

Read on: Python: Input Variables