

Модуль `builtin_data`

```
InputTable: Union[DataSourceClass, None]
InputTables: Tuple[Union[DataSourceClass, None], ...]
InputVariables: VariablesClass
OutputTable: ConfigurableOutputTableClass ''' или OutputTableClass, если
отключена настройка "Разрешить формировать выходные столбцы из кода" '''
OutputTables: Tuple[ConfigurableOutputTableClass, ...] ''' или
OutputTableClass, если отключена настройка "Разрешить формировать выходные
столбцы из кода" '''
```

```
class DataType(enum.IntEnum):
```

```
    None = 0
    Boolean = 1
    DateTime = 2
    Float = 3
    Integer = 4
    String = 5
    Variant = 6
```

```
class DataKind(enum.IntEnum):
```

```
    Undefined = 0
    Continuous = 1
    Discrete = 2
```

```
class UsageType(enum.IntEnum):
```

```
    Unspecified = 0
    Excluded = 1
    Useless = 2
    Active = 3
    Used = 3
    Input = 3
    Predicted = 4
    Output = 4
    Key = 5
    Group = 6
    Value = 7
```

Transaction = 8

Item = 9

```
class ColumnInfo(Protocol):
```

 Name: str, required

 DisplayName: str, optional

 DataType: int, optional

 DataKind: int, optional

 DefaultUsageType: int, optional

```
class ColumnClass(Sequence[Union[bool, int, str, float, datetime.datetime,  
None]]):
```

 Index: int, read-only

 Name: str, read-only

 DisplayName: str, read-only

 DataType: int, read-only

 DataKind: int, read-only

 DefaultUsageType: int, read-only

 RowCount: int, read-only

 Get(row: int) -> Union[bool, int, str, float, datetime.datetime, None]

 IsNull(row: int) -> boolean

```
class InputColumnClass(ColumnClass):
```

 UsageType: int, read-only

```
class OutputColumnClass(ColumnClass):
```

```
    Set(value: Union[bool, int, str, float, datetime.datetime, None]) ->  
None
```

```
class ConfigurableOutputColumnClass(OutputColumnClass):
```

 DisplayName: str

 DataType: int

 DataKind: int

 DefaultUsageType: int

```
''' y InputTable ColumnClass – это InputColumnClass,  
    y OutputTable ColumnClass – это OutputColumnClass или  
ConfigurableOutputColumnClass '''
```

```
class ColumnsClass(Mapping[Union[int, str], ColumnClass],  
Sequence[ColumnClass])
```

```
class DataSourceClass:
```

```

    ColumnCount: int, read-only
    Columns: ColumnsClass, read-only
    RowCount: int, read-only
    Get(row: int, col: Union[int, str]) -> Union[bool, int, str, float,
datetime.datetime, None]
    GetColumn(col: Union[int, str]) -> ColumnClass
    IsNull(row: int, col: Union[int, str]) -> bool

class OutputTableClass(DataSourceClass):
    Columns: ColumnsClass, read-only
    GetColumn(col: Union[int, str]) -> OutputColumnClass
    Append() -> None
    Set(col: Union[int, str], value: Union[bool, int, str, float,
datetime.datetime, None]) -> None

class ConfigurableOutputTableClass(OutputTableClass):
    GetColumn(col: Union[int, str]) -> ConfigurableOutputColumnClass
    AssignColumns(source: Iterable[ColumnInfo]) -> None
    AddColumn(ColumnInfo: Optional[ColumnInfo], Name: Optional[str],
              DisplayName: Optional[str], DataType: Optional[int],
              DataKind: Optional[int], DefaultUsageType: Optional[int])
-> OutputColumnClass
    InsertColumn(Index: int, ColumnInfo: Optional[ColumnInfo],
              Name: Optional[str], DisplayName: Optional[str],
              DataType: Optional[int], DataKind: Optional[int],
              DefaultUsageType: Optional[int]) -> OutputColumnClass
    DeleteColumn(col: Union[int, str]) -> None
    ClearColumns() -> None

class VariableClass:
    Index: int, read-only
    Name: str, read-only
    DisplayName: str, read-only
    DataType: int
    Value: Union[bool, int, str, float, datetime.datetime, None]

class VariableItemsClass(Mapping[Union[int, str], VariableClass],
Sequence[VariableClass])

class VariablesClass:

```

Items: VariableItemsClass, read-only
Count: int, read-only

Примечание: Принимать аргументы по ключевым словам могут только методы `OutputTable.AddColumn` и `OutputTable.InsertColumn`, остальные принимают только позиционированные аргументы.

Модуль `builtin_pandas_utils`

```
#Метод осуществляет преобразование InputTable в pandas.DataFrame
def to_data_frame(table: DataSourceClass, useNullableArrays=False) ->
pandas.DataFrame

#Метод задает структуру полей OutputTable по pandas.DataFrame
def prepare_compatible_table(table: OutputTableClass, dataframe:
pandas.DataFrame, with_index=False) -> None

#Метод осуществляет запись из pandas.DataFrame в OutputTable
def fill_table(table: OutputTableClass, dataframe: pandas.DataFrame,
with_index=False) -> None
```