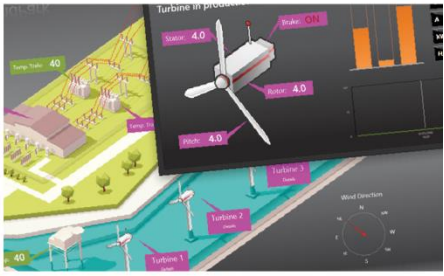
INDUSTRIAL AUTOMATION SOFTWARE
progea

Movicon.next™
Automation Platform.next generation



# Movicon NExT

## 20.0 Tools

**Ver.3.4.268**

# Table of Contents

# 1. Introduction

Movicon.NExT uses some tools, developed by Progea, to execute certain operations or functionalities. These tools are listed below with a brief description of their use. For further information on how they function and how to configure their application of use, please consult their relative help.

**Project Builder**

This is a wizard that is used to modify Movicon.NExT-based projects to allow you customize some of the main functionalities according to your needs.
For further details please refer to the chapters relating to the Project Builder.

**SQL DB configuration**

This tool is used to manage the following Operations the Movicon.NExT project tables:

- Aggregate Data Logger Tables
- Use of Partitioned Tables

For further details please refer to the chapters relating to the SQL Database Configurator Tool.

**Alarm Dispatcher**

This isn't exactly a tool in the real sense but an independent module on the Movicon.NExT client side, which can be started up and stopped using the command line, with alarm and event notification functions from different configurable modules. For further details please refer to the chapters relating to the Alarm Dispatcher.

# 2. Project Builder

## 2.1. Introduction

The Movicon.NExT projects can also be created using automatic procedures that can be completely customized. For instance, wizards can be autonomously created to create or edit Movicon.NExT projects and components.
This offers the expert user a great way to develop projects or repetitive project parts much faster by creating custom wizards that will reduce development times to the minimum.
For instance, projects can be automatically created with screens, variables, alarms, Historian and many other project functions by simply creating the rules to automatically create the automatic creation procedures.
This powerful tool is based on a component designed for this purpose and which can be managed with Movicon VB.NET script or by autonomously creating the tool externally using Visual Studio. This tool is called the Movicon.NExT Project Builder. For example, some users use it to create automated procedures for creating projects or parts of projects, simply by selecting the import sources (e.g. Excel files or external databases). The less expert user may prefer to use the configuration Wizard to define the things they wish to create for the Project Builder to then create the project and all the resources already appropriately configured as required.

### The Project Builder

The Project Builder tool is actually an object (.DLL) that is called when needed by using code. It can be used to create or modify the various Movicon.NExT project resources to be executed, by using the actual project's VB.NET script code, or by autonomously creating an executable tool with Visual Studio. In both cases, the MoviconNextBuilder.dll component may be managed with its own code to automatically create or edit a Movicon.NExT project, or setup guided procedures (wizard) to help users to create and configure the project or part of it.
The Project Builder tool offers the following functions:

**I/O Data Server**

All properties belonging to Tags and Alarms can be edited/customized.
- Create/edit/delete a tag
- Associate/remove an alarm to/from a tag
- Associate/remove an historican to/from a tag
- Create/remove/delete a folder from the tag list
- Create/edit/delete a datalogger
- Add/remove column to/from a datalogger
- Create/remove/delete an alarm area
- Create/remove/delete an alarm source

All those properties belong to alarms can be edited/customized.

- Create/edit/delete an alarm
- Create/edit/delete a Prototype
- Create/edit/delete an Engineering Unit
- Edit the IO server settings (Application Name, etc)
- Add/configure/remove a communication driver

**Screens**
- Create/edit/delete a screen
- Insert/edit/delete a toolbox object from the screen
- Insert/edit/delete a library symbol form the screen

**Texts**
- Add/edit/remove a string
- Add/remove a language

**Parameters**
- Create/delete parameterization file
- Create/delete a parameter (copy alias/variable) from a parameterization file

## 2.2. Visual Studio Project Settings

The Visual Studio project must be set for Net framework 4.6.1 or later versions using the Movicion.NExt installation folder (C:\Program Files\Progea\Movicon.NExT) as the output path.

In order to save the destination folder you may have to open VisualStudio as pc administrator depending on the path in which Movicon NExt was previously installed.

You should find the **using System.Reflection** and **using CommandManager** declarations along with the following **References** in the Visual Studio project:

- **CommandManager.dll** found in the default path: C:\Program Files\Progea\Movicon.NExT\Managers
- **DataLoggerModel.dll** found in the default path: C:\Program Files\Progea\Movicon.NExT
- **DevExpress.Xpo.Vx.x.dll** found in the default path: %windir%\Microsoft.NET\assembly\GAC_MSIL
- **MoviconNextBuilder.dll** found in the default path: C:\Program Files\Progea\Movicon.NExT
- **OPCUAViewModel.dll** found in the default path: C:\Program Files\Progea\Movicon.NExT
- **ScreenParameterSettings.dll** found in the default path: C:\Program Files\Progea\Movicon.NExT
- **ScreenSettings.dll** found in the default path: C:\Program Files\Progea\Movicon.NExT
- **StringModel.dll** found in the default path: C:\Program Files\Progea\Movicon.NExT
- **UFInterfaces.dll** found in the default path: C:\Program Files\Progea\Movicon.NExT
- **UFUAModel.dll** found in the default path: C:\Program Files\Progea\Movicon.NExT
- **ViewModelBase.dll** found in the default path: C:\Program Files\Progea\Movicon.NExT
- **WPFUtilities.dll** found in the default path: C:\Program Files\Progea\Movicon.NExT

Finally, the following node should be added within the **App.config** file:

```
<runtime>
 <assemblyBinding xmlns="urn:schemas-microsoft-com:asm.v1">
```

```
        <probing
        privatePath="DocumentManagers;DocumentManagers\LogicExtensions;Managers;
        Drivers; Toolbox;DataSinks;CommonPlugins;DesignPlugins" />
 </assemblyBinding>
</runtime>
```

## 2.3. Project Instance Manager

In order to operate in a Movicon.NeXT project using the class library described below in this document, you will first need to get the ProjectBuilder object's instance from the MoviconNextBuilder class. In order to do this you will need to insert a specific instance during the VisualStudio application initialization phase as shown in this example:

```
MoviconNextBuilder.ProjectBuilder Builder;
Uri fileprj;
try {
   Builder = new MoviconNExTBuilder.ProjectBuilder();
   // get the Movicon NeXT project path
   fileprj = "ProjectPath\\ProjectName.UFProject"
   // initialize the ProjectBuilder instance
   Builder.Init(fileprj);
 } catch (Exception eX) {
   MessageBox.Show("Error in initializing MoviconNextBuilder dll: " + eX.Message,
"MovNextBuilderEx", MessageBoxButton.OK, MessageBoxImage.Error);
```

At this point you will be able to obtain the Movicon NeXT project manager objects. The objects to use are:

- **IODataServer**: instance of the class with same name that manages all the data server aspects.
- **ParametersContainer**: instance of the Parameters class for screen parameterization.
- **ScreenContainer**: instance of the class with the same name, for managing screens in create and edit modes.
- **StringsResource:** instance of the TextResource class for managing texts and languages.

The objects can be accessed as properties of the ProjectBuilder object.

## 2.4. IODataServer

### 2.4.1. Save, IODataServer Method

| | |
|---|---|
| **Syntax** | Void Save() |
| **Description** | Saves all the changes pending in the IODataServer server. |

| Parameter | Description |
|---|---|
| - | - |

**Result** -

**Example:**

### 2.4.2. AddAlarmArea, IODataServer Method

**Syntax** UFUAModel.UFUAArea AddAlarmArea(string name, [UFUAModel.UFUAArea parent = null])

**Description** Adds an Alarm Area with name in the parent Alarm area.

| Parameter | Description |
|---|---|
| string name | name of new alarm area |
| UFUAModel.UFUAArea parent | reference to eventual parent alarm area (optional). |

**Result** UFUAModel.UFUAArea (object representing the alarm area just added)

**Example:**

```
// instance of IODataServer variable
var server = Builder.IODataServer;
// add a new alarm prototype
// add alarm area
var alarmArea = server.AddAlarmArea("NewAlarmArea");
// add a new alarm prototype in the area
var subarea = server.AddAlarmArea("NewAlarmSubArea", alarmArea);
// saving IOServer
```

### 2.4.3. AddAlarmDefinition, IODataServer Method

**Syntax** UFUAModel.UFUAAlarmDefinition AddAlarmDefinition(string name, UFUAModel.UFUAAlarmSource parent, [bool overwrite = False])

**Description** Adds a new definition to the alarm with name of 'name' within the source. .

| Parameter | Description |
|---|---|
| string name | name of the new alarm definition |
| UFUAModel.UFUAAlarmSource parent | reference to the alarm source in which to define the new alarm |
| bool overwrite | overwrites an existing alarm |

**Result** UFUAModel.UFUAAlarmDefinition (object representing the alarm definition just added)

**Example:**

```
// instance of IODataServer variable
var server = Builder.IODataServer;
// add a new alarm prototype
// add alarm area
var alarmArea = server.AddAlarmArea("NewAlarmArea");
// add alarm source in the area
var source = server.AddAlarmSource("NewAlarmSource", alarmArea);
// add alarm edfinition for the source
var alarm = server.AddAlarmDefinition("NewAlarm", source);
// saving IOServer configuration change
server.Save();
```

### 2.4.4. AddAlarmSource, IODataServer Method

**Syntax** UFUAModel.UFUAAlarmSource AddAlarmSource(string name, UFUAModel.UFUAArea parent)

**Description** Adds a new alarm name definition within the source.

| Parameter | Description |
|---|---|
| string name | name of the new alarm source |
| UFUAModel.UFUAArea parent | reference to the area in which to define new source |

**Result** UFUAModel.UFUAAlarmSource (object represening the alarm source just added)

**Example:**

```
// instance of IODataServer variable
var server = Builder.IODataServer;
// add a new alarm prototype
// add alarm area
```

```
var alarmArea = server.AddAlarmArea("NewAlarmArea");
// add alarm source in the area
var source = server.AddAlarmSource("NewAlarmSource", alarmArea);
// saving IOServer configuration change
server.Save();
```

### 2.4.5. AddBaseAddress, IODataServer Method

**Syntax** UFUAModel.UFUABaseAddress AddBaseAddress([string transport = net.pipe])

**Description** Adds tranport type.

| Parameter | Description |
|---|---|
| string transport | name of new transport, possible values:<br>• net.pipe<br>• net.tcp<br>• opc.tcp<br>• http<br>• https<br>• nosecurityhttp |

**Result** UFUAModel.UFUABaseAddress: object representing the transport just added.

**Example:**

```
// instance of IODataServer variable
var server = Builder.IODataServer;
// add the transport net.tcp to the IOServer
var baseAddress = server.AddBaseAddress("net.tcp");
// changing a property of the transport object
baseAddress.Port = 62847;
// saving IOServer configuration change
server.Save();
```

### 2.4.6. AddDatalogger, IODataServer Method

**Syntax** DataLoggerModel.DataLoggerSettings AddDatalogger(string name, [bool overwrite = False])

**Description** Adds a DataLogger prototype with name as name

| Parameter | Description |
|---|---|

| | |
|---|---|
| string name | name of DataLogger prototype to be added. |
| [bool overwrite = False] | Overwrites an existing prototype, optional. In the presense of a datalogger homonym, the return value will be null if no overwrite is requested. |

**Result** DataLoggerModel.DataLoggerSettings: object that consents access to the various properties of the Datalogger prototype just inserted.

**Example:**

```
// instance of IODataServer variable
var server = Builder.IODataServer;
// getting aa existing tag reference
var tag = server.GetTag("TagName");
if (tag != null) {
  // adding a DataLogger
  var dl = server.AddDatalogger("DataLoggerName");

  if (dl != null) {
    // enabling the DataLogger
    dl.Enable = true;
   // setting the sampling time (in this example the interval time is represented in
hundred nanoseconds (10000 ticks in a millisecond)
    dl.RecordingTimeInterval = new TimeSpan(10000000); // 1 sec of interval
   // adding a DataLogger New column
   var col = server.AddDataLoggerColumn("DataLoggerColumn", dl);
   // adding a tag reference to the new added column
   if (tag != null)
   col.ColumnTag = new UFUAModel.TagEntityReference(tag);
}
// saving IOServer configuration change
server.Save();
```

### 2.4.7. AddDataLoggerColumn, IODataServer Method

**Syntax** DataLoggerModel.DataLoggerColumn AddDataLoggerColumn(string name, DataLoggerModel.DataLoggerSettings datalogger, [bool overwrite = False])

**Description** Consents to adding a new column definition in the datalogger

| Parameter | Description |
|---|---|
| string name | name of the Column to be added |
| DataLoggerModel.DataLoggerSettings datalogger | datalogger to which the new column definition is to be added |
| [bool overwrite = False] | oeverwrites an existing column, optional. In cases of honomous columns, |

| | |
|---|---|
| | the return value will be null unless an overwrite is requested. |

**Result** DataLoggerModel.DataLoggerColumn: object which consents acced to the various added column's properties.

**Example:**

```
// instance of IODataServer variable
var server = Builder.IODataServer;
// getting aa existing tag reference
var tag = server.GetTag("TagName");
if (tag != null) {
  // adding a DataLogger
  var dl = server.AddDatalogger("DataLoggerName");

  if (dl != null) {
    // enabling the DataLogger
    dl.Enable = true;
   // setting the sampling time (in this example the interval time is represented in
hundred nanoseconds (10000 ticks in a millisecond)
    dl.RecordingTimeInterval = new TimeSpan(10000000); // 1 sec of interval
   // adding a DataLogger New column
   var col = server.AddDataLoggerColumn("DataLoggerColumn", dl);
   // adding a tag reference to the new added column
   if (tag != null)
   col.ColumnTag = new UFUAModel.TagEntityReference(tag);
}
// saving IOServer configuration change
server.Save();
```

### 2.4.8. AddEngineeringUnit, IODataServer Method

**Syntax** UFUAModel.UFUAEngineeringUnit AddEngineeringUnit(string name, [bool overwrite = False])

**Description** Adds the engineering unit name

| Parameter | Description |
|---|---|
| string name | name of the engineering unit to be added |
| [bool overwrite = False] | overwrites existing unit, optional. In cases of an homonymous unit the return value will be null if overwrite is not requested. |

**Result** UFUAModel.UFUAEngineeringUnit: object that consents access to the various properties of the added engineering unit.

**Example:**

```
// instance of IODataServer variable
var server = Builder.IODataServer;
// engineering unit creation
var cust = server.AddEngineeringUnit("customUnit");
if (cust != null) {
  cust.EURangeHigh = 1000;
  cust.EURangeLow = 0;
  cust.InstrumentRangeHigh = 10000;
  cust.InstrumentRangeLow = 10;
  cust.UnitName = "twix";
}
// saving IOServer configuration change
server.Save();
```

### 2.4.9. AddFolder, IODataServer Method

**Syntax** UFUAModel.UFUAFolder AddFolder(string name, [UFUAModel.UFUAFolder folder = null])

**Description** Used to add a folder to a specific tree level of the variables in the IO DataServer.

| Parameter | Description |
|---|---|
| string name | name of the Folder to be added |
| [bool overwrite = False] | folder in which to insert the new folder. Optional, if omitted the folder will be added at the root of the tag tree. |

**Result** UFUAModel.UFUAFolder: object that consents access to the various added folder's properties.

**Example:**

```
// instance of IODataServer variable
var server = Builder.IODataServer;
// add a folder in the tag tree root
var tagFolder = server.AddFolder("Folder");
// add a subfolder in the tag tree
tagFolder = server.AddFolder("Folder2", tagFolder);
// saving IOServer configuration change
server.Save();
```

### 2.4.10. AddHistorian, IODataServer Method

**Syntax** UFUAModel.UFUAHistorianSettings AddHistorian(string name, [bool overwrite = False])

**Description** Used to add a new Historian prototype definition.

| Parameter | Description |
|---|---|
| string name | name of Historian prototype to be added |
| [bool overwrite = False] | overwrites an existing prototype, optional. In cases of honomyous prototypes, the return value will be null if no overwrite is requested. |

**Result** UFUAModel.UFUAHistorianSettings: object which allows access to the various properties of the added Historian prototype.

In the example which follows, you will also see how to add a tag to the Historian just created.

**Example:**

```
// instance of IODataServer variable
var server = Builder.IODataServer;
// getting an existing tag reference
var tag = server.GetTag("TagName");
if (tag != null) {
  var historianName = "NewHistorian";
  // new Historian Prototype
  var historian = server.AddHistorian(historianName);
  if (historian != null) {
    // some properties of Historian Prototype
    historian.ExceptionDeviationFormat = UFUAModel.DeviationType.AbsoluteValue;
    historian.ExceptionDeviation = 1.0;
    historian.MaxTimeInterval = new TimeSpan(0, 0, 2); // 1 sec of inreval
    historian.MinTimeInterval = new TimeSpan(0, 0, 2); // 1 sec of inreval
  }
// assign historian reference to the tag
tag.HistorianSettings = historianName;
// saving IOServer configuration change
server.Save();
```

### 2.4.11. AddNewDriver, IODataServer Method

**Syntax** bool AddNewDriver(string assemblyname, string friendlyname, string factory)

**Description** Used to add a driver to the IO DataServer.

| Parameter | Description |
|---|---|
| String assemblyname | name of driver dll |
| String friendlyname | mnemonic reference name to driver in the Movicon NeXT project |
| String factory | name of builder |

**Result** Boolean

**Example:**

```
// instance of IODataServer variable
var server = Builder.IODataServer;
// add the ModbusTcp IODriver
server.AddNewDriver("ModbusTCP.dll", "ModbusTcp", "Modbus");
// saving IOServer configuration change
```

### 2.4.12. AddPrototype, IODataServer Method

**Syntax** UFUAModel.UFUATagPrototype AddPrototype(string name, [bool overwrite = False])

**Description** Adds a data model prototype to the IO DataServer.

| Parameter | Description |
|---|---|
| String name | name of the data model prototype to be added |
| [bool overwrite = False] | overwrites an existing prototype, optional. In the presense of an homonymous prototype, the value will return null if not requested to overwrite. |

**Result** UFUAModel. UFUATagPrototype : object that allows access to the various properties of the added data model prototype.

**Example:**

```
// instance of IODataServer variable
var server = Builder.IODataServer;
// prototype creation
string prototypeName = "structVar";
var proto = server.AddPrototype(prototypeName);
// saving IOServer configuration change
server.Save();
```

### 2.4.13. AddPrototypeMember, IODataServer Method

**Syntax** UFUAModel.UFUATag AddPrototypeMember(string name, UFUAModel.UFUATagPrototype proto, [UFUAModel.DataType type = 2], [bool overwrite = False])

**Description** Adds a member to an existing data model prototype.

| Parameter | Description |
|---|---|
| String name | name of member to be added. |
| UFUAModel.UFUATagPrototype proto | data model prototype to add member to. |
| [UFUAModel.DataType type = 2] | member's data type. The data types allowed are those deinge in the UFUAModel.DataType enumeration:<br>• Boolean<br>• Byte<br>• Double<br>• Float<br>• Int16<br>• Int32<br>• Int64<br>• Sbyte<br>• String<br>• Uint16<br>• Uint32<br>• Uint64 |
| [bool overwrite = False] | overwrites an existing member, optional. In cases where a prototype of the same name exists, the value will return null if not requested to overwrite. |

**Result** UFUAModel. UFUATag : object allowing access to the various properties of the added member.

In the following example you will also see how to a create a tag from the data model prototype just created.

**Example:**

```
// instance of IODataServer variable
var server = Builder.IODataServer;
// prototype creation
string prototypeName = "structVar";
var proto = server.AddPrototype(prototypeName);
// add prototype members
if (proto != null) {
  server.AddPrototypeMember("Indication", proto, UFUAModel.DataType.Int16);
  server.AddPrototypeMember("Label", proto, UFUAModel.DataType.String);
  server.AddPrototypeMember("Distance", proto, UFUAModel.DataType.Double);
```

```
  server.AddPrototypeMember("Present", proto, UFUAModel.DataType.Boolean);
  server.AddPrototypeMember("Quantity", proto, UFUAModel.DataType.Int32);
}
// prototype element tag creation
var tag = server.AddTag("Element");
if (tag != null) {
  tag.ModelType = UFUAModel.ModelType.ObjectType;
  tag.PrototypeModel = prototypeName;
  tag.Description = "Production element";
}
// saving IOServer configuration change
server.Save();
```

### 2.4.14. AddTag, IODataServer Method

**Syntax** UFUAModel.UFUATag AddTag(string name, [UFUAModel.UFUAFolder folder = null], [UFUAModel.DataType type = 2], [bool overwrite = False])

**Description** Used to define a new tag and to add it to the IO DataServer

| Parameter | Description |
|---|---|
| String name | Name of tag to be added. |
| UFUAModel. UFUAFolder folder | Folder in which to add tag, optional. |
| [UFUAModel.DataType type = 2] | Tag's data type, optional. The data types allows are those defined in the UFUAModel.DataType enumeration:<br>• Boolean<br>• Byte<br>• Double<br>• Float<br>• Int16<br>• Int32<br>• Int64<br>• Sbyte<br>• String<br>• Uint16<br>• Uint32<br>• Uint64 |
| [bool overwrite = False] | Overwrites an existing tag, optional. When a tag with the same name exists already, the value will return null unless requested to overwrite it. |

**Result** UFUAModel.UFUATag: object allowing access to the various added tag's properties.

**Example:**

```
// instance of IODataServer variable
```

```
var server = Builder.IODataServer;
// integer tag creation
tag = server.AddTag("NewIntegerTag", null, UFUAModel.DataType.UInt16, true);
  if (tag != null) {
    tag.ArrayDimension = 0;
    tag.Description = "New integer tag";
    tag.InitialValue = "0";
  }
// add a folder in the tag tree root
var tagFolder = server.AddFolder("Folder");
// boolean tag creation into the new folder
  tag = server.AddTag("NewBooleanTag", tagFolder, UFUAModel.DataType.Boolean,
true);
  if (tag != null) {
    tag.ArrayDimension = 0;
    tag.Description = "New boolean tag";
    tag.InitialValue = "false";
  }
// saving IOServer configuration change
server.Save();
```

### 2.4.15. AssignAlarmToTag, IODataServer Method

**Syntax** bool AssignAlarmToTag(UFUAModel.UFUATag tag, UFUAModel.UFUAAlarmDefinition alarm, [int type = 0], [string text = ])

**Description** Assigns an existing alarm prototype to the tag.

| Parameter | Description |
|---|---|
| UFUAModel.UFUATag tag | Tag to which the alarm prototype is to be assigned. |
| UFUAModel.UFUAAlarmDefinition alarm | Alarm prototype to be assigned. |
| [int type = 0] | Assignment mode, optional<br>• 0: Single<br>• 1: AnyTagBit<br>• 2: AntTagElement (for tag array) |
| [string text = ] | alarm text, optional |

**Result** -

**Example:**

```
// instance of IODataServer variable
var server = Builder.IODataServer;
// getting a tag reference from IOServer
var tag = server.GetTag("TagName");
// getting an alarm reference from IOServer
```

```
var alarm = server.GetAlarmDefinition("NewAlarm",
server.GetAlarmSource("NewAlarmSource", server.GetAlarmArea("NewAlarmArea")));
if (tag != null && alarm != null)
  // assign an alarm to the tag
  server.AssignAlarmToTag(tag, alarm, 0, "New Alarm text");
// saving IOServer configuration change
server.Save();
```

### 2.4.16. DeleteAlarmArea, IODataServer Method

**Syntax** bool DeleteAlarmArea(string name, [UFUAModel.UFUAArea parent = null])

**Description** Used to delete alarm area definition

| Parameter | Description |
|---|---|
| string name | Name of alarm area to delete. |
| [UFUAModel.UFUAArea parent = null] | Any area in which the area to be deleted is defined, optional. |

**Result** Boolean

This function will delete the alarm area even when empty.

**Example:**

```
// instance of IODataServer variable
var server = Builder.IODataServer;
// delete alarm area
server.DeleteAlarmArea("NewAlarmArea");
// saving IOServer configuration change
server.Save();
```

### 2.4.17. DeleteAlarmDefinition, IODataServer Method

**Syntax** bool DeleteAlarmDefinition(string name, UFUAModel.UFUAAlarmSource parent)

**Description** Used to delete alarm definition

| Parameter | Description |
|---|---|
| string name | name of alarm definition to be deleted |

| | |
|---|---|
| UFUAModel.UFUAAlarmSource parent | name of the alarm source containing the alarm definition |

**Result** Boolean

This function will delete the alarm definition even when already assigned a tag.

**Example:**

```
// instance of IODataServer variable
var server = Builder.IODataServer;
// get the alarm area instance
var area = server.GetAlarmArea("NewAlarmArea");
// get the alarm source instance
var source = server.GetAlarmSource("NewAlarmSource", area);
// delete the alarm definition
server.DeleteAlarmDefinition("NewAlarm", source);
// saving IOServer configuration change
server.Save();
```

### 2.4.18. DeleteAlarmSource, IODataServer Method

**Syntax** bool DeleteAlarmSource(string name, UFUAModel.UFUAArea parent)

**Description** Deletes the definition of the alarm source

| Parameter | Description |
|---|---|
| string name | name of the alarm source to be deleted |
| UFUAModel.UFUAArea parent | alarm area containing the source to be deleted |

**Result** Boolean

This function will delete the alarm source even when empty.

**Example:**

```
// instance of IODataServer variable
var server = Builder.IODataServer;
// get the alarm area instance
var area = server.GetAlarmArea("NewAlarmArea");
// delete alarm source
server.DeleteAlarmSource("NewAlarmSource", area);
// saving IOServer configuration change
server.Save();
```

### 2.4.19. DeleteDatalogger, IODataServer Method

**Syntax** bool DeleteDatalogger(string name)

**Description** Deletes a DataLogger

| Parameter | Description |
|---|---|
| string name | name of DataLogger to be deleted |

**Result** Boolean

This function will delete data logger even when empty.

**Example:**

```
// instance of IODataServer variable
var server = Builder.IODataServer;
// delete DalaLogger
server.DeleteDatalogger("DataLoggerName");
// saving IOServer configuration change
server.Save();
```

### 2.4.20. DeleteDataloggerColumn, IODataServer Method

**Syntax** bool DeleteDataloggerColumn(string name,
DataLoggerModel.DataLoggerSettings datalogger)

**Description** Deletes the definition of a column in the indicated DataLogger

| Parameter | Description |
|---|---|
| string name | name of column to be deleted |
| DataLoggerModel.DataLoggerSettings datalogger | DataLogger from which column is to be deleted |

**Result** Boolean

**Example:**

```
// instance of IODataServer variable
var server = Builder.IODataServer;
// getting DataLogger instance
```

```
var dataLogger = server.GetDatalogger("DataLoggerName");
// delete DataLogger column
server.DeleteDataloggerColumn("DataLoggerColumnName", dataLogger);
// saving IOServer configuration change
server.Save();
```

### 2.4.21. DeleteEngineeringUnit, IODataServer Method

**Syntax** bool DeleteEngineeringUnit(string name)

**Description** Deletes indicated engineering unit

| Parameter | Description |
|---|---|
| string name | name of engineering unit to be deleted |

**Result** Boolean

**Example:**

```
// instance of IODataServer variable
var server = Builder.IODataServer;
// delete Engineering Unit
server.DeleteEngineeringUnit("customUnit");
// saving IOServer configuration change
server.Save();
```

### 2.4.22. DeleteFolder, IODataServer Method

**Syntax** bool DeleteFolder(string name, [UFUAModel.UFUAFolder folder = null])

**Description** Deletes the folder indicated by the folder passed as second parameter. If the second parameter is not indicated, the folder will be cancelled from the root.

| Parameter | Description |
|---|---|
| string name | Name of folder to be deleted. |
| [UFUAModel.UFUAFolder folder = null] | Name of any eventual folder containing the folder to be deleted, if allowed the folder will be deleted from the root. To obtain the folder object instance, use IODataServer.GetFolder, optional. |

**Result** Boolean

This function will delete the folder even when empty.

**Example:**

```
// instance of IODataServer variable
var server = Builder.IODataServer;
// delete Folder
server.DeleteFolder("Folder2",server.GetFolder("Folder"));
// saving IOServer configuration change
server.Save();
```

### 2.4.23. DeleteHistorian, IODataServer Method

**Syntax** bool DeleteHistorian(string name, [bool removefromtag = False])

**Description** Deletes the indicated Historian prototype removing any tag associations.

| Parameter | Description |
|---|---|
| string name | Name of Historian prototype to be deleted. |
| [bool removefromtag = False] | When set to True, removes any associations to the Historian of previously assigned tags. Optional. |

**Result** Boolean

**Example:**

```
// instance of IODataServer variable
var server = Builder.IODataServer;
// delete Historian
server.DeleteHistorian("NewHistorian");
// saving IOServer configuration change
server.Save();
```

### 2.4.24. DeletePrototype, IODataServer Method

**Syntax** bool DeletePrototype(string name)

**Description** Deletes the indicated tag prototype.

| Parameter | Description |
|---|---|
| string name | Name of prototype to be deleted. |

**Result** Boolean

This function deletes the prototype even when empty and/or if there are tags based on it.

**Example:**

```
// instance of IODataServer variable
var server = Builder.IODataServer;
// delete Prototype
server.DeletePrototype("structVar");
// saving IOServer configuration change
server.Save();
```

### 2.4.25. DeleteTag, IODataServer Method

**Syntax** bool DeleteTag(string name, [UFUAModel.UFUAFolder folder = null])

**Description** Deletes indicated Tag.

| Parameter | Description |
|---|---|
| string name | Name of tag to be deleted. |
| [UFUAModel.UFUAFolder folder = null] | Instance of the folder object containing the tag. If allowed, it will deleted from the root. To get the folder object's instance you must use IODataServer.GetFolder. Optional. |

**Result** Boolean

**Example:**

```
// instance of IODataServer variable
var server = Builder.IODataServer;
// getting the tag folder
var tagFolder server.GetFolder("tagFolder");
// delete Tag
server.DeleteTag("tagName", tagFolder);
// saving IOServer configuration change
server.Save();
```

### 2.4.26. GetAlarmArea, IODataServer Method

**Syntax** UFUAModel.UFUAArea GetAlarmArea(string name, [UFUAModel.UFUAArea parent = null])

**Description** Gets the instance of a UFUAModel.UFUAArea type object represening the indicated alarm area.

| Parameter | Description |
|---|---|
| string name | Name of Alarm Area to retrieve. |
| [UFUAModel.UFUAFolder folder = null] | Instance of the alarm area containing the area to be retrieved. If the alarm area has been added, it will be contained at the root. Optional. |

**Result** UFUAModel.UFUAArea: instance of the desired AlarmArea object. When in error, will return null.

**Example:**

```
// instance of IODataServer variable
var server = Builder.IODataServer;
// getting the alarm area
var area server.GetAlarmArea("NewAlarmArea");
```

### 2.4.27. GetAlarmDefinition, IODataServer Method

**Syntax** UFUAModel.UFUAAlarmDefinition GetAlarmDefinition(string name, UFUAModel.UFUAAlarmSource parent)

**Description** Gets the instance of an UFUAModel.UFUAAlarmDefinition object representing the indicated alarm definition.

| Parameter | Description |
|---|---|
| string name | Name of the alarm definition to be retrieved. |
| UFUAModel.UFUAAlarmSource parent | Instance of the Alarm Source object in which the alarm is defined. |

**Result** UFUAModel.UFUAAlarmDefinition: Instance of the desired alarm definition. When in error returns null.

**Example:**

```
// instance of IODataServer variable
var server = Builder.IODataServer;
// getting the alarm
var alarm server.GetAlarmDefinition("NewAlarm",
server.GetAlarmSource("NewAlarmSource", server.GetAlarmArea("NewAlarmArea")));
```

### 2.4.28. GetAlarmSource, IODataServer Method

**Syntax** UFUAModel.UFUAAlarmSource GetAlarmSource(string name, UFUAModel.UFUAArea parent)

**Description** Gets the instance of an UFUAModel.UFUAAlarmSource object represening the indicated alarm source.

| Parameter | Description |
|---|---|
| string name | Name of the desired alarm source. |
| UFUAModel.UFUAArea parent | Instance of the alarm area in which source is defined. |

**Result** UFUAModel.UFUAAlarmSource: Instance of the alarm source desired. Returns null when in error.

**Example:**

```
// instance of IODataServer variable
var server = Builder.IODataServer;
// getting the alarm source
var source server.GetAlarmSource("NewAlarmSource",
server.GetAlarmArea("NewAlarmArea"));
```

### 2.4.29. GetBaseAddress, IODataServer Method

**Syntax** UFUAModel.UFUABaseAddress GetBaseAddress(string transport)

**Description** Gets the instance of the UFUAModel.UFUABaseAddress object representing the indicated transport.

| Parameter | Description |
|---|---|
| string transport | transport type to retrieve. |

**Result** UFUAModel. UFUABaseAddress: indicated transport object instance. When in error, returns null.

Returned object can be used to delete transport type.

**Example:**

```
// instance of IODataServer variable
var server = Builder.IODataServer;
// getting the base address
var transport server.GetBaseAddress("net.tcp");
// deleting transport object
transport.Delete();
```

```
// saving IOServer configuration change
server.Save();
```

### 2.4.30. GetDatalogger, IODataServer Method

**Syntax** DataLoggerModel.DataLoggerSettings GetDatalogger(string name)

**Description** Gets the indicated DataLoggerModel.DataLoggerSettings object instance.

| Parameter | Description |
|---|---|
| string name | Name of DataLogger to retrieve. |

**Result** DataLoggerModel.DataLoggerSettings: indicated DataLogger object instance. When in error returns Null.

**Example:**

```
// instance of IODataServer variable
var server = Builder.IODataServer;
// getting the data logger
var dataLogger server.GetDatalogger("DataLoggerName");
```

### 2.4.31. GetDataloggerColumn, IODataServer Method

**Syntax** DataLoggerModel.DataLoggerColumn GetDataloggerColumn(string name, DataLoggerModel.DataLoggerSettings datalogger)

**Description** Gets instance of the indicated DataLoggerModel.DataLoggerColumn column object beloning to the indicated DataLoggerModel.DataLoggerSettings Datalogger.

| Parameter | Description |
|---|---|
| string name | Name of the column to be retrieved |
| DataLoggerModel.DataLoggerSettings datalogger | Instance of the Datalogger object which column belongs to. |

**Result** DataLoggerModel.DataLoggerColumn: indicated olumn object instance. When in error returns null.

**Example:**

```
// instance of IODataServer variable
var server = Builder.IODataServer;
// getting the data logger
```

```
var dataLogger server.GetDatalogger("DataLoggerName");
// getting the data logger column
var dataLoggerColumn = server.GetDataloggerColumn("DataLoggerName",
dataLogger);
```

### 2.4.32. GetDataloggerList, IODataServer Method

**Syntax** System.Collections.Generic.List<string> GetDataloggerList()

**Description** Get the list of all Dataloggers configured in the project.

| Parameter | Description |
|---|---|
| - | - |

**Result** System.Collections.Generic.List<string>: DataLogger list.

**Example:**

```
// instance of IODataServer variable
var server = Builder.IODataServer;

// getting the data logger
var DataloggerList server.GetDataloggerList();
```

### 2.4.33. GetEngineeringUnit, IODataServer Metod

**Syntax** UFUAModel.UFUAEngineeringUnit GetEngineeringUnit(string name)

**Description** Gets instance of the desired UFUAModel.UFUAEngineeringUnit object.

| Parameter | Description |
|---|---|
| string name | name of engineering unit to be retrieved. |

**Result** DataLoggerModel.DataLoggerColumn: indicated column object instance. When in error, returns null.

**Example:**

```
// instance of IODataServer variable
var server = Builder.IODataServer;
// getting the engineering unit
var engUnit server.GetEngineeringUnit("customUnit");
```

### 2.4.34. GetFolder, IODataServer Metod

**Syntax** UFUAModel.UFUAFolder GetFolder(string name, [UFUAModel.UFUAFolder folder = null])

**Description** Gets an instance of the desired UFUAModel.UFUAFolder object if contained in another folder.

| Parameter | Description |
|---|---|
| string name | Name of folder to retrieve. |
| [UFUAModel.UFUAFolder folder = null] | Any eventual object folder instance containing the desired folder. |

**Result** UFUAModel.UFUAFolder: instance object of indicated folder. Returns null when in error.

**Example:**

```
// instance of IODataServer variable
var server = Builder.IODataServer;
// get folder
var folder = server.GetFolder("Folder2",server.GetFolder("Folder"));
// saving IOServer configuration change
server.Save();
```

### 2.4.35. GetHistorian, IODataServer Metod

**Syntax** UFUAModel.UFUAHistorianSettings GetHistorian(string name)

**Description** Gets an instance from the indicated UFUAModel.UFUAHistorianSettings object.

| Parameter | Description |
|---|---|
| string name | name of historian to be retrieved |

**Result** UFUAModel. UFUAHistorianSettings: indicated Historian's instance object. Returns null when in error.

**Example:**

```
// instance of IODataServer variable
var server = Builder.IODataServer;
// getting Historian
var historian = server.GetHistorian("NewHistorian");
```

### 2.4.36. GetHistorianList, IODataServer Method

**Syntax** System.Collections.Generic.List<string> GetHistorianList()

**Description** Get the list of all the Historians configured in the project.

| Parameter | Description |
|---|---|
| - | - |

**Result** System.Collections.Generic.List<string>: Historian list.

**Example:**

```
// instance of IODataServer variable
var server = Builder.IODataServer;

// getting the historian list
var HistorianList server.GetHistorianList();
```

### 2.4.37. GetPrototype, IODataServer Method

**Syntax** UFUAModel.UFUATagPrototype GetPrototype(string name)

**Description** Gets the instance of the indicated UFUAModel.UFUATagPrototype object.

| Parameter | Description |
|---|---|
| string name | name of prototype to retrieve. |

**Result** UFUAModel.UFUATagPrototype : instance object of indicated prototype. When in error, returns null.

**Example:**

```
// instance of IODataServer variable
var server = Builder.IODataServer;
// getting Prototype
var proto = server.GetPrototype("structVar");
```

### 2.4.38. GetServerConfiguration, IODataServer Method

**Syntax** UFUAModel.UFUAConfiguration GetServerConfiguration()

**Description** gets an instance from the UFUAModel.UFUAConfiguration object of the IODataServer server.

| Parameter | Description |
|---|---|
| - | - |

**Result** UFUAModel.UFUAConfiguration: instance object of the IODataServer server. When in error returns null.

**Example:**

```
// instance of IODataServer variable
var server = Builder.IODataServer;
// getting server configuration
var serverConfig = server.GetServerConfiguration();
```

### 2.4.39. GetServerEntityReference, IODataServer Method

**Syntax** OPCUAViewModel.OPCUAEntityReference GetServerEntityReference()

**Description** Gets an instance from the OPCUAViewModel.OPCUAEntityReference object of the IODataServer server.

| Parameter | Description |
|---|---|
| - | - |

**Result** OPCUAViewModel.OPCUAEntityReference: instance object of the IODataServer server. In case of error returns null.

**Example:**

```
// instance of IODataServer variable
var server = Builder.IODataServer;
// getting server entity reference
var serverReference = server.GetServerEntityReference();
```

### 2.4.40. GetTag, IODataServer Method

**Syntax** UFUAModel.UFUATag GetTag(string name, [UFUAModel.UFUAFolder folder = null])

**Description** Gets the UFUAModel.UFUATag object's instance of the indicated tag in the folder it belongs to.

| Parameter | Description |
|---|---|
| string name | name of tag to be retrieved |
| [UFUAModel.UFUAFolder folder = null] | any eventual folder object instance containing the desired tag. |

**Result** UFUAModel.UFUATag: the desired tag's instance object. Returns null when in error.

**Example:**

```
// instance of IODataServer variable
var server = Builder.IODataServer;
// getting the tag folder
var tagFolder server.GetFolder("tagFolder");
// getting Tag
var tag = server.GetTag("tagName", tagFolder);
```

### 2.4.41. GetTagEntityReference, IODataServer Method

**Syntax** OPCUAViewModel.OPCUAEntityReference GetTagEntityReference(string name, [string instance = null])

**Description** Gets an instance of the indicated tag's OPCUAViewModel.OPCUAEntityReference object. Attention, this method only functions when the Save method has been called after having added the tag.

| Parameter | Description |
|---|---|
| string name | name of tag to retrieve, possibly composed according to the folder in which it is contained (eg. Folder\varname) |
| [string instance = null] | instance name when tag is a prototype type. |

**Result** OPCUAViewModel.OPCUAEntityReference: desired tag's instance object. Return null when in error.

The object returned by the server function as parameter in objects belonging to the toolbox which have been inserted in a screen.

**Example:**

```
// instance of IODataServer variable
var server = Builder.IODataServer;
// getting Tag
```

```
var tagReference = server.GetTagEntityReference("tagName");
// getting Proto Tag
var tagReference = server.GetTagEntityReference("memberName","tagName");
```

**Syntax** OPCUAViewModel.OPCUAEntityReference
(**overloaded**) GetTagEntityReference(UFUAModel.UFUATag tag)
**Description** Ottiene un'istanza dell'oggetto OPCUAViewModel.OPCUAEntityReference della tag indicata.

| Parameter | Description |
|---|---|
| UFUAModel.UFUATag Tag: | UFUAModel.UFUATag object type that represents the tag. An object of this type can be retrieved by calling the GetTag, or as a result of an Addtag. This method also functions for tag recently added to the server. |

**Result** OPCUAViewModel.OPCUAEntityReference: instance object of the tag desired. When in error returns null.

**Example:**

```
// instance of IODataServer variable
var server = Builder.IODataServer;
// getting the tag folder
var tagFolder server.GetFolder("tagFolder");
// getting Tag
var tag = server.GetTag("tagName", tagFolder);
// getting Tag
var tagReference = server.GetTagEntityReference(tag);
// getting Proto Tag
tag = server.GetUFATag("tagName", "memberName");
// getting Proto Tag
tagReference = server.GetTagEntityReference(tag);
```

### 2.4.42. GetTagOPCUAEntityReference, IODataServer Metod

**Syntax** OPCUAViewModel.OPCUAEntityReference
GetTagOPCUAEntityReference(string tagname, [string membername = null])
**Description** Gets an instance from the OPCUAViewModel.OPCUAEntityReference object of the tag indicated, if structure type.

| Parameter | Description |
|---|---|
| string tagname | name of tag to be retrieved, eventually composed according to the folder in which it is contained. (eg. Folder\varname) |

| | |
|---|---|
| [string memberinstance = null] | eventual structure member name, if the tag is structure type. The syntax must mirror the prototype structure (eg. membername or folder\membername or substructure\membername) |

**Result** OPCUAViewModel.OPCUAEntityReference: object instance of the desired tag. When in error will return null.

The object returned by the function is needed as a parameter in those objects inserted on screen and that belong to the toolbox.

**Example:**

```
// instance of IODataServer variable
var server = Builder.IODataServer;

// getting Tag in a folder
var tagReference = server.GetTagOPCUAEntityReference("folderName\\tagName");

// getting prototype member Tag
tagReference = server.GetTagOPCUAEntityReference("protoTagName","memberName");
```

### 2.4.43. GetVarTagEntityReference, IODataServer Metod

**Syntax** UFUAModel.TagEntityReference GetVarTagEntityReference(string tagname, [string membername = null])

**Description** Gets an instance from the UFUAModel.TagEntityReference object of the tag indicated, if structure type.

Attention: This method only functions when the Save method has been called after adding the tag.

| Parameter | Description |
|---|---|
| string tagname | name of tag to be retrieved, eventually composed according to the folder in which it is contained. (eg. Folder\varname) |
| [string memberinstance = null] | eventual structure member name, if the tag is structure type. The syntax must mirror the prototype structure (eg. membername or folder\membername or substructure\membername) |

**Result** UFUAModel.TagEntityReference: object instance of the desired tag. When in error will return null.

The object returned by the function is needed as a parameter in datalogger column objects.

**Example:**

```
// instance of IODataServer variable
var server = Builder.IODataServer;

// getting Tag in a folder
var tagReference = server.GetVarTagEntityReference("folderName\\tagName");

// getting prototype member Tag
tagReference = server.GetVarTagEntityReference("protoTagName","memberName");
```

**Syntax (Overloaded)** UFUAModel.TagEntityReference GetVarTagEntityReference(UFUAModel.UFUATag tag)

**Description** Obtains an instance of the UFUAModel.TagEntityReference object of the specified tag, possibly structure type.

| Parameter | Description |
|---|---|
| UFUAModel.UFUATag tag: | UFUAModel.UFUATag object type that represents the tag. An object of this type can be obtained by calling the GetTag, or as a result of an Addtag. This method also functions for tags recently added in the server. |

**Result** UFUAModel.TagEntityReference: instance object of the desired tag. When in error returns null.

**Example:**

```
// instance of IODataServer variable
var server = Builder.IODataServer;

// getting the tag folder
var tagFolder server.GetFolder("tagFolder");

// getting Tag
var tag = server.GetTag("tagName", tagFolder);

// getting Tag
var tagReference = server.GetVarTagEntityReference(tag);

// getting Proto Tag
tag = server.GetUFATag("tagName", "memberName");

// getting Proto Tag
tagReference = server.GetVarTagEntityReference(tag);
```

### 2.4.44. GetUFATag, IODataServer Metod

**Syntax** UFUAModel.UFUATag GetUFATag(string tagname, [string membername = null])

**Description** Gets an UFUAModel.UFUATag object's instance of the tag indicated, possibly structure type.

| Parameter | Description |
|---|---|
| string name: | name of tag to retrieve, possibly composed according to the folders in which it is containted (eg. Folder\varname) |
| [string memberinstance = null] | structure member name if tag is structure type. The syntax must reflect the prototype structure (eg. nomember or folder/nomember or substructure/nomember). |

**Result** UFUAModel.UFUATag: object instance of the desired tag. When in error returns null.

The object returned by the function is needed as a parameter in objects belonging to the toolbox inserted on screen.

**Example:**

```
// instance of IODataServer variable
var server = Builder.IODataServer;
// getting Tag in a folder
var tag = server.GetUFATag("folderName\\tagName");
// getting prototype member Tag
var tag = server.GetUFATag("protoTagName","memberName");
```

### 2.4.45. IODataServer, IODataServer Method

**Syntax** IODataServer(DocumentManager.ComponentService.IDocument parent, UFUAEditorManagerComponent editormanager)

**Description** This is the class builder. An already built class instance object that is created with the initialization procedure described in initialization phase (see the MoviconNextBuilder.ProjectBuilder project instance manager)

| Parameter | Description |
|---|---|
| - | - |

**Result** -

**Example:**

### 2.4.46. RemoveAlarmFromTag, IODataServer Method

**Syntax** bool RemoveAlarmFromTag(UFUAModel.UFUATag tag, UFUAModel.UFUAAlarmDefinition alarm)

**Description** Removes the assignment of a tag to an alarm.

| Parameter | Description |
|---|---|
| UFUAModel.UFUATag tag | Object instance that represents the tag from which to remove the assignment to the alarm. |
| UFUAModel.UFUAAlarmDefinition alarm | Object instance representing the alarm from which to remove the assignment. |

**Result** Boolean

**Example:**

```
// instance of IODataServer variable
var server = Builder.IODataServer;
// getting the tag object
var tag server.GetTag("tagName");
// getting the alarm object
var alarm = server.GetAlarmDefinition("NewAlarm",
server.GetAlarmSource("NewAlarmSource", server.GetAlarmArea("NewAlarmArea")));
// remove alarm from tag
server.RemoveAlarmFromTag(tag, alarm);
// saving IOServer configuration change
server.Save();
```

## 2.5. Parameters

### 2.5.1. Save, Parameter Method

**Syntax** Void Save()

**Description** Saves all the changes pending in the Parameters

| Parameter | Description |
|---|---|
| - | - |

**Result** -

**Example:**

### 2.5.2. AddParameterFile, Parameter Method

**Syntax** ScreenParametersSettings.Documents.ScreenParametersDocument AddParameterFile(string name)

**Description** Adds a parameter file with the specified name.

| Parameter | Description |
|---|---|
| string name | name of the parameter file to be added |

**Result** ScreenParametersSettings.Documents.ScreenParametersDocument (instance of an object representing the created parameter file. Returns null if in error)

**Example:**

```
// instance of Parameters object
var paramCollection = Builder.ParametersContainer;
var parmFile = paramCollection.AddParameterFile("NewParamFile")
// saving Parameters change
paramCollection.Save();
```

### 2.5.3. DeleteParameterFile, Parameter Method

**Syntax** ScreenParametersSettings.Documents.ScreenParametersDocument GetParameterFile(string name)

**Description** Gets the instance of an object representing the indicated parameter files.

| Parameter | Description |
|---|---|
| string name | Name of parameter file desired. |

**Result** Boolean

**Example:**

```
// instance of Parameters object
var paramCollection = Builder.ParametersContainer;
// deleting parameter file
paramCollection.DeleteParameterFile("NewParamFile")
// saving Parameters change
paramCollection.Save();
```

### 2.5.4. GetParameterFile, Parameter Method

**Syntax** System.Uri GetParameterUri(string name)

**Description** Gets the indicated parameter file's Uri.

| Parameter | Description |
|---|---|
| string name | name of desired parameter file |

**Result** Parameter file's Uri. Returns null when in error.

**Example:**

```
// instance of Parameters object
var paramCollection = Builder.ParametersContainer;
// getting parameter file
var parameters = paramCollection.GetParameterFile("NewParamFile")
```

### 2.5.5. GetParameterUri, Parameter Method

**Syntax** bool DeleteParameterFile(string name)

**Description** Deletes the indicated parameter file.

| Parameter | Description |
|---|---|
| string name | Name of parameter file to delete |

**Result** ScreenParametersSettings.Documents.ScreenParametersDocument (instance of an object representing the parameter file. Returns null when in error).

**Example:**

```
// instance of Parameters object
var paramCollection = Builder.ParametersContainer;
// getting parameter file
var parameters = paramCollection.GetParameterFile("NewParamFile")
```

### 2.5.6. Parameters, Parameter Method

**Syntax** Parameters(DocumentManager.ComponentService.IDocument parent, UFProjectManagerComponent editormanager)

**Description** This is the class builder. A class object instance which is already built and created with the initialization procedure during the initialization phase (see MoviconNextBuilder.ProjectBuilder project instance manager)

| Parameter | Description |
|---|---|
| - | - |

**Result** -

**Example:**

## 2.6. ScreenContainer

### 2.6.1. ToolBoxFolder, ScreenContainer Method

**Syntax** string ToolBoxFolder()

**Description** Gets the path in which the toolbox element files are contained.

| Parameter | Description |
|---|---|
| - | - |

**Result** String: Toolbox file path

**Example:**

```
// instance of ScreenContainer object
var screenCollection = Builder.ScreensContainer;
// getting screen Uri
var typesFolder = screencollection.ToolBoxFolder();
```

### 2.6.2. AddFolder, ScreenContainer Method

**Syntax** bool AddFolder(string folder)

**Description** Adds a folder by the name indicated in the screen tree.

| Parameter | Description |
|---|---|
| string name | name of screen folder to be added. Diverse folder levels can be inserted simply by indicating them in the parameter eg: "FirstFolder\\LastFolder" adds the LastFolder folder content in the FirstFolder folder |

**Result** Bool: operation result.

**Example:**

```
// instance of ScrenContainer object
var screenCollection = Builder.ScreenContainer;
// adding screen folder
screenCollection.AddFolder("NewFolder");

// saving ScreenContainer change
screenCollection.Save();
```

### 2.6.3. AddGeoData, ScreenContainer Method

**Syntax** bool AddGeoData(string name, double latitude, double longitude, [OPCUAViewModel.OPCUAEntityReference longitudeTag = null], [OPCUAViewModel.OPCUAEntityReference latitudeTag = null], [string subfolder = null], [bool overwrite = False])

**Description** Adds a pair of geographical coordinates to the indicated screen.

| Parameter | Description |
|---|---|
| string name | name of the screen in which to insert the references to the coordinates |
| double latitude | latitude coordinate |
| double longitude | longitude coordinate |
| [OPCUAViewModel.OPCUAEntityReference longitudeTag = null] | instance of an object that represents the tag from which to retrieve the value of the latitude, optional. |

| | |
|---|---|
| [OPCUAViewModel.OPCUAEntityReference latitudeTag = null] | instance of an object that represents the tag from which to retrieve the value of the longitude, optional. |
| [string subfolder = null] | folder in which the screen is inseted in the screen tree, optional. |
| [bool overwrite = False] | overwrite flag, optional |

**Result** Bool: operation result.

**Example:**

```
// instance of ScreenContainer object
var screenCollection = Builder.ScreensContainer;
// adding geo coordinates
screencollection.AddGeoData("NewScreen", 12.1, -35.5);
// saving ScreenContainer change
screenCollection.Save();
```

### 2.6.4. AddScreen, ScreenContainer Method

**Syntax** ScreenSettings.ScreenDocument AddScreen(string name, string modelname, [string subfolder = null])

**Description** Adds a screen with the name indicated based on the template modelname.

| Parameter | Description |
|---|---|
| string name | name of screen to be added |
| string modelname | name of template to be used for the created screen. The template names are the names of the files contained in the folder (and subfolders) ProgramData\Progea\Movicon.NExT.3.0\NewScreenTypes\ |
| string subfolder = null | name of the folder in which the function's screen object is contained. |

**Result** ScreenSettings.ScreenDocument: instance of an object representing the created screen. Returns null if in error.

**Example:**

```
// instance of ScreenContainer object
var screenCollection = Builder.ScreenContainer;
// adding screen
var screen = screenCollection.AddScreen("NewScreen","full_HD_anthracite");
// saving screenContainer change
screenCollection.Save();
```

### 2.6.5. AddSymbol, ScreenContainer Method

**Syntax** MoviconNextBuilder.ScreenElement AddSymbol(string screen, string item, string name, double x, double y, double width, double height, [string subfolder = null], [OPCUAViewModel.OPCUAEntityReference tag = null], [bool linkonly = True])

**Description** Adds an element from the symbol library to the indicated screen.

| Parameter | Description |
|---|---|
| string name | element name added |
| string screen | name of screen to add element to |
| string item | name of library symbol to be used. The name of the symbols are the names of files contained folder (and subfolders) obtained with the NewScreenTypesFolder function of this class. |
| double x | x coordinate in pixels |
| double y | y coordinate in pixels |
| double width | element's width |
| double height | element's length |
| [string subfolder = null] | folder where screen is defined. Optional: |
| [OPCUAViewModel.OPCUAEntityReference tag = null] | object representing the tag to be connect to the added element. Optional. |
| [bool linkonly = True] | Insertion flag as link. Optional. |

**Result** MoviconNextBuilder.ScreenElement: istanza di un oggetto che rappresenta l'elemento aggiunto allo screen. In caso di errore ritorna null.

**Example:**

```
// instance of ScreenContainer object
var screenCollection = Builder.ScreenContainer;
// adding symbol
var screen = screenCollection.AddSymbol("NewScreen", "Acces_Traffic1"
                , "access"
                , 200, 200, 70, 70);
// saving screenContainer change
screenCollection.Save();
```

### 2.6.6. AddToolboxItem, ScreenContainer Method

**Syntax** MoviconNextBuilder.ScreenElement AddToolboxItem(string screen, string item, string name, double x, double y, double width, double height, [string subfolder = null], [OPCUAViewModel.OPCUAEntityReference tag = null])

**Description** Adds an element from the toolbox to the indicated screen.

| Parameter | Description |
|---|---|
| string name | Name of added element. |
| string screen | Name of scrren to which to add element. |
| string item | Name of toolbox element. The names of toolbox elements are the names of files contained in the folder (and subfolders) obtained from the ToolBoxFolder function of this class. |
| double x | x coordinate in pixels |
| double y | y coordinate in pixels |
| double width | element's width |
| double height | element's height |
| [string subfolder = null] | folder in which screen is defined. Optional. |
| [OPCUAViewModel.OPCUAEntityReference tag = null] | Obect representing the tag to connect to the added object. Optional. |

**Result** MoviconNextBuilder.ScreenElement: instance of an object representing the element added to the screen. Return null when in error.

**Example:**

```
// instance of ScreenContainer object
var screenCollection = Builder.ScreenContainer;
// adding toolbox item
var screen = screenCollection.AddToolboxItem("NewScreen", "Rectangular
CommandButton  D"              , "Button", 200, 200, 70, 70);
// saving screenContainer change
screenCollection.Save();
```

### 2.6.7. DeleteElement, ScreenContainer Method

**Syntax** ScreenSettings.ScreenDocument GetScreen(string name, [string subfolder = null])

**Description** Gets the object instance representing the indicated screen.

| Parameter | Description |
|---|---|
| string name | screen name |
| [string subfolder = null] | folder where screen is defined, optional |

**Result** ScreenSettings.ScreenDocument: screen object instance. Returns null if in error.

**Example:**

```
// instance of ScreenContainer object
var screenCollection = Builder.ScreensContainer;
// getting screen
screencollection.GetScreen("NewScreen");
```

### 2.6.8. DeleteScreen, ScreenContainer Method

**Syntax** bool DeleteScreen(string name, [string subfolder = null])

**Description** Deletes the indicated screen

| Parameter | Description |
|---|---|
| string name | Name of the screen to be deleted. |
| [string subfolder = null] | Folder within which the screen is defined. Optional. |

**Result** Boolean

**Example:**

```
Bool: esito dell'operazione
Esempio
// instance of ScreenContainer object
var screenCollection = Builder.ScreensContainer;
// deleting element
screencollection.DeleteScreen("NewScreen");
// saving ScreenContainer change
```

```
screenCollection.Save();
```

### 2.6.9. GetScreen, ScreenContainer Method

**Syntax** bool DeleteElement(string screen, string elementname, [string subfolder = null])

**Description** Deletes the indicated element from screen.

| Parameter | Description |
|---|---|
| string elementname | Name of element to be deleted. |
| string screen | Name of screen from which element is to be deleted. |
| [string subfolder = null] | folder in which screen is defined. Optional. |

**Result** Boolean

**Example:**

```
// instance of ScreenContainer object
var screenCollection = Builder.ScreensContainer;
// deleting element
screencollection.DeleteElement("NewScreen","Button");
// saving ScreenContainer change
screenCollection.Save();
```

### 2.6.10. GetScreenUri, ScreenContainer Method

**Syntax** System.Uri GetScreenUri(string name, [string subfolder = null])

**Description** Tets the inidcated screen's Uri

| Parameter | Description |
|---|---|
| string name | name of screen |
| [string subfolder = null] | Folder within which the screen is defined. Optional. |

**Result** System.Uri: Uri of the desired screen. Returns null if in error.

**Example:**

```
// instance of ScreenContainer object
var screenCollection = Builder.ScreensContainer;
// getting screen Uri
screencollection.GetScreenUri("NewScreen");
```

### 2.6.11. NewScreenTypesFolder, ScreenContainer Method

**Syntax** string NewScreenTypesFolder()

**Description** Gets the path containing the symbol file contents.

| Parameter | Description |
|---|---|
| - | - |

**Result** String

**Example:**

```
// instance of ScreenContainer object
var screenCollection = Builder.ScreensContainer;
// getting screen Uri
var typesFolder = screencollection.NewScreenTypesFolder ();
```

### 2.6.12. Save, ScreenContainer Method

**Syntax** void Save()

**Description** Saves all the changes pending in the ScreenContainer.

| Parameter | Description |
|---|---|
| - | - |

**Result** -

### 2.6.13. Screen Container Examples

You can cast an added screen object with the desired properties in the following way as shown in this example:

## Button

**Content Property:**

If the object is the return value from the **AddToolboxItem** function:

```
// instance of ScreenContainer object
var screenCollection = Builder.ScreenContainer;
// add a new toolbox item to the screen
var button = screenCollection.AddToolboxItem("ScreenName", "Rectangular
CommandButton  D", "ButtonName", 100, 100, 100, 50);
// to manage some properties of the added object you can use a type casting
((System.Windows.Controls.Button)((System.Windows.Controls.ContentControl)button.El
ement).Content).Content = "Button Text";
Nel caso in cui si debba intervenire sulle proprietà di un oggetto aggiunto in
precedenza ad uno screen allora si deve agire in questo modo:
// instance of ScreenContainer object
var screenCollection = Builder.ScreenContainer;
// get instance of ScreenDocument object
var screenDoc = screencollection.GetScreen("ScreenName");
// get instance of ScreenEntity object
var btnEntity = (screenDoc.MapScreenEntities)["ButtonName"];
// get access to the content property by mean of Element object with a cast to the
appropriate type
((System.Windows.Controls.Button)((System.Windows.Controls.ContentControl)btnEntity
.Element).Content).Content = "Button Text";
```

## TextBox

**FontSize and Text Property:**

If the object is the return value from the **AddToolboxItem** function:

```
// instance of ScreenContainer object
var screenCollection = Builder.ScreenContainer;
// add a new toolbox item to the screen
var button = screenCollection.AddToolboxItem("ScreenName", "Text", "TextName", 100,
100, 100, 50);
// to manage some properties of the added object you can use a type casting
((System.Windows.Controls.TextBox)text.Element).FontSize = 15;
((System.Windows.Controls.TextBox)text.Element).Text = "Text Content";
```

### 2.6.14. ScreenContainer, ScreenContainer Method

| | |
|---|---|
| **Syntax** | ScreenContainer(DocumentManager.ComponentService.IDocument parent, UFProjectManagerComponent manager) |
| **Description** | This is the class builder.  A class instance object, that is therefore already build and is created with the initialization procedure described in the |

initialization phase (see MoviconNextBuilder.ProjectBuilder project instance manager)

| Parameter | Description |
|---|---|
| - | - |

**Result** -

### 2.6.15. SearchXamlItem, ScreenContainer Method

**Syntax** string SearchXamlItem(string itemname, string path)

**Description** Searches for an .xaml file in start folder in a starting folder. This is needed, for example, to get the complete path of an object from the toolbox or symbol library.

| Parameter | Description |
|---|---|
| string itemname | Name of file to search for. |
| string path | Start path. |

**Result** String: complete path of the object searched for.

**Example:**

```
// instance of ScreenContainer object
var screenCollection = Builder.ScreensContainer;
// getting screen Uri
var itemPath = screencollection.SearchXamlItem("AlarmWindowControl",
"C:\\ProgramData\\Progea\\Movicon.NExT.3.1")
```

### 2.6.16. SetBackground, ScreenContainer Method

**Syntax** bool SetBackground(string name, System.Windows.Media.Brush background, [string subfolder = null])

**Description** Sets the indicated screen background.

| Parameter | Description |
|---|---|

| | |
|---|---|
| string name | Name of screen. |
| System.Windows.Media.Brush background | System.Window.Media.Brushes enumeration element. |
| [string subfolder = null] | Folder in which screen is defined. Optional. |

**Result** Boolean

**Example:**

```
// instance of ScreenContainer object
var screenCollection = Builder.ScreensContainer;
// deleting element
screencollection.SetBackground("NewScreen" , Brushes.Black);
// saving ScreenContainer change
screenCollection.Save();
```

### 2.6.17. SetScreenEntity, ScreenContainer Metod

**Syntax** bool SetScreenEntity(string screen, string name, OPCUAViewModel.OPCUAEntityReference tag, [string subfolder = null])

**Description** sets the Tag Item for the nameobject name in the screenname screen.

| Parameter | Description |
|---|---|
| **String screen** | name of screen |
| **String name** | name of object in screen |
| **OPCUAViewModel.OPCUAEntityReference tag** | refers to the tag to be used in the object |
| **[string subfolder = null]** | folder within which the optional screen is defined. |

**Result** Boolean

**Example:**

```
// instance of ScreenContainer object
var screenCollection = Builder.ScreensContainer;
var server = Builder.IODataServer
var tagscreen = server.GetTagEntityReference("NomeTag");

// setting screen element tag
screenCollection.SetScreenEntity(screenName, "editVar", tagscreen, "screenFolder");
// saving ScreenContainer change
```

```
screenCollection.Save();
```

# 2.7. TextResource

## 2.7.1. TextResource, TexResource Method

**Syntax** TextResource(DocumentManager.ComponentService.IDocument parent, StringEditorManagerComponent editormanager)

**Description** This is the class builder. An object instance of the class that is already built and created with the initialization procedure describedin initialization phase (see MoviconNextBuilder.ProjectBuilder project instance builder)

| Parameter | Description |
|---|---|
| - | - |

**Result** -

**Example:**

## 2.7.2. AddLanguage, TexResource Method

**Syntax** System.Globalization.CultureInfo AddLanguage(string language)

**Description** Adds the indicated language in the project.

| Parameter | Description |
|---|---|
| string language | name of the language/culture to add |

**Result** ScreenParametersSettings.Documents.ScreenParametersDocument (instance of an object that represents the created parameter file. Returns null if in error.

**Example:**

```
// defining a Project Builder string resource
var texts = Builder.StringsResource;
// add a new en-US language definition
texts.AddLanguage("en-US");
// add a new it-IT language definition
```

```
texts.AddLanguage("it-IT");
// saving text configuration change
texts.Save();
```

### 2.7.3. AddString, TexResource Method

**Syntax** System.Collections.Generic.List<StringModel.UFStringLocaleText> AddString(string stringid)

**Description** Adds with indicated id.

| Parameter | Description |
|---|---|
| string stringid | string id to be added |

**Result** System.Collections.Generic.List<StringModel.UFStringLocaleText>: List of objects that represent the string ids just inserted. Returns null when in error.

**Example:**

```
// defining a Project Builder string resource
var texts = Builder.StringsResource;
// add a new en-US language definition
texts.AddLanguage("en-US");
// add a new it-IT language definition
texts.AddLanguage("it-IT");
// add a new stringId and Locale string definition for each language
var str = texts.AddString("first_Id");
if (str.Count == 2) {
  str[0].Locale = "Open";
  str[1].Locale = "Apri";
}
// add a new stringId and Locale string definition for each language
str = texts.AddString("second_Id");
if (str.Count == 2) {
  str[0].Locale = "Close";
  str[1].Locale = "Chiudi";
}
// add a new stringId and Locale string definition for each language
str = texts.AddString("third_Id");
if (str.Count == 2) {
  str[0].Locale = "Up";
  str[1].Locale = "Su";
}
// saving text configuration change
texts.Save();
```

### 2.7.4. DeleteLanguage, TexResource Method

**Syntax** bool DeleteString(string stringId)

**Description** Deletes the indicated string from the project.

| Parameter | Description |
|---|---|
| string stringId | Name of the string id to be deleted. |

**Result** Boolean

**Example:**

```
// defining a Project Builder string resource
var texts = Builder.StringsResource;
// delete the en-US language definition
texts.DeleteString("second_Id");
// saving text configuration change
texts.Save();
```

### 2.7.5. DeleteString, TexResource Method

**Syntax** bool DeleteLanguage(string language)

**Description** Deletes the indicated language from the project.

| Parameter | Description |
|---|---|
| string language | Name of the language/culture to be deleted. |

**Result** Boolean

**Example:**

```
// defining a Project Builder string resource
var texts = Builder.StringsResource;
// delete the en-US language definition
texts.DeleteLanguage("en-US");
// saving text configuration change
texts.Save();
```

### 2.7.6. GetString, TextResources Metod

**Syntax** System.Collections.Generic.List<StringModel.UFStringLocaleText> GetString(string stringid)

**Description** Retrieves the string indicated.

| Parameter | Description |
|---|---|
| string stringId | name of id string to retrieve. |

**Result** System.Collections.Generic.List<StringModel.UFStringLocaleText>: List of objects that represent the strings of the id just inserted. When in error returns null.

**Example:**

```
// defining a Project Builder string resource
var texts = Builder.StringsResource;
// retrieve the en-US locale string
var retStrings = texts.GetString("first_Id");
if (retStrings != null) {
  foreach(StringModel.UFStringLocaleText rs in retStrings) {
    if (rs.Culture = "en-US"){
      MessageBox.Show("String from " + "'first_Id'" + " in " + rs.Culture + " is: " +
rs.Locale);
    }
  }
}
```

### 2.7.7. Save, TextResource Method

**Syntax** Void Save()

**Description** Saves all the changes pending in the screen container

| Parameter | Description |
|---|---|
| - | - |

**Result** -

**Example:**

### 2.7.8. SetString, TexResource Method

**Syntax** bool SetString(string stringid, string language, string newvalue)

**Description** Changes the indicated string.

| Parameter | Description |
|---|---|
| string language | Language which string is changed to. |
| string stringId | Name of the string id to be changed. |
| string newvalue | new string value |

**Result** Boolean

**Example:**

```
// defining a Project Builder string resource
var texts = Builder.StringsResource;
// delete the en-US language definition
texts.SetString("second_Id","en-US","New string value");
// saving text configuration change
texts.Save();
```

# 3. SQL Database Configurator

## 3.1. Database Configuration Tool

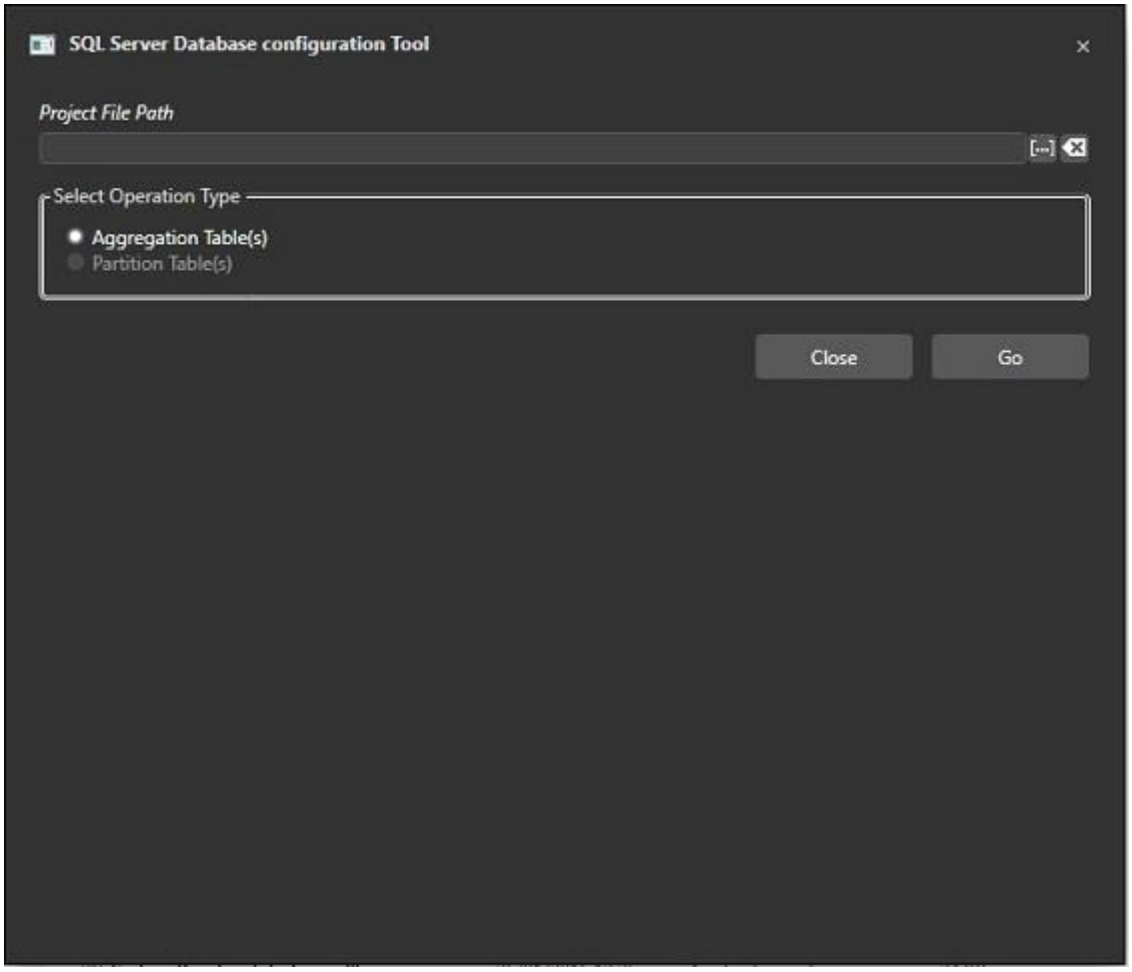
The "SQL Database Configuration Tool" is used to create tables deriving from the main DataLogger table with average, minim, maximum data value aggregations based on minutes, hours and days. Basically, each table contains a record for each aggregation time interval (minutes, hours, days) and each column from the DataLogger of origin will generate 3 columns, one with the average value, one with the minimum value and one with the maximum reference value.

### DataLogger table aggregation

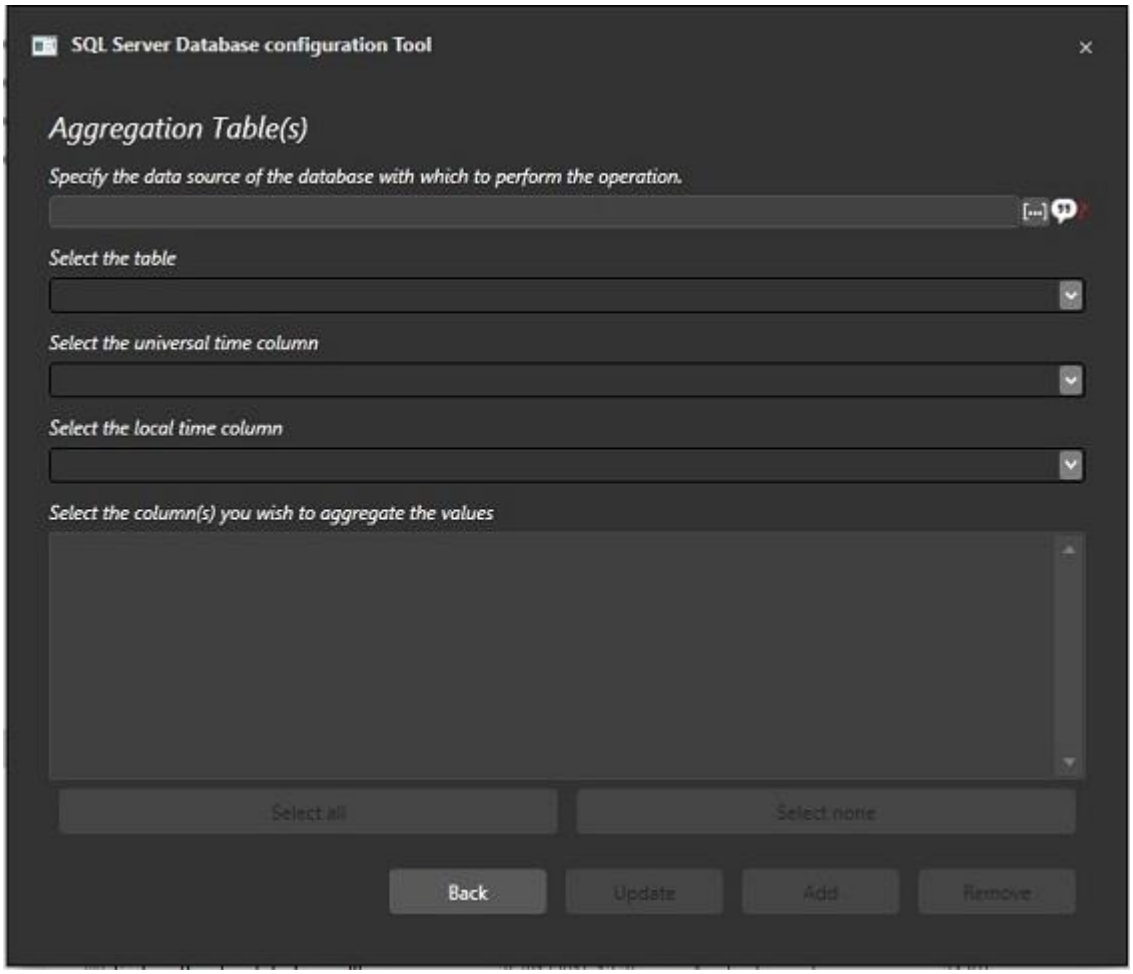
Aggregating Datalogger tables is basically done in two ways:

**Using the User Interface**

Launch the C:\Program Files\Progea\Movicon.NExT x.y\SQLDatabaseConfiguration.exe executable with a double click to open this interface:




Select a project from the first parameter and press the Go button to start up the procedure to aggregate tables of DataLoggers configured in the project.
Table aggregation can also be done without selecting a project by using the Go button as shown in the screen shot below:




- **Data Source:** use to specify the database in which the DataLoggers to be aggregated are.
- **Table:** use to specify the DataLogger table to aggregate.
- **Universal Time Column:** use to enter the name to the column containing the UTC data.
- **Local Time Column:** use to enter the name to the column containing Local Time data.
- **Select columns:** use to select the columns, numerically, to aggregate.

Once all these settings have been completed, the **'Update'** and '**Add'** buttons will activate to allow existing aggregation updates to add aggregated tables respectively, along with the necessary storage procedures and triggers.
If the selected table has already been subjected to aggregation beforehand, the Update button will also be enabled.

**Using the command line**

The tool in question can also be called using the command line with the following options:

**/S:** silent execution
**/F:** NExT project path
**/W:** project protection password if any
**/O:** operation type to be performed: "1" to aggregate tables

Example of command line parameters:

/F"E:\MyDownloads\TestDatalogger1\TestDatalogger1\TestDatalogger1.UFProject" /W"mypassword" /O"1" /S

Other advanced options which reflect the description of the user interface already seen above:

**/P:** command type: "A" (Add), "R" (Remove), "U" (Update), "C" (Check)
**/D:** database connection string, in ADO.NET or DevExpress format
**/T:** table name
**/U:** UTC column name
**/L:** Local time column name
**/D:** Data column names to be aggregated
**/H:** Hide column names to exclude them from the interface selection list.

INDUSTRIAL AUTOMATION SOFTWARE

progea


**Progea Srl**
Via D'Annunzio, 295
I-41123 Modena
info@progea.com
Tel +39 059 451060

**Progea International SA**
via Sottobisio, 28
6828 Balerna (CH)
international@progea.com
Tel +41 91 96 76 610

**Progea Deutschland GmbH**
Marie-Curie Str., 12
D-78048 VS Villingen
info@progea.de
Tel +49 (0)7721 99838 0

**Progea North America Corp.**
2380 State Road 44, Suite C
Oshkosh, WI 54904
info@progea.us
Tel. +1 (888) 305-2999