# Dokumentation des Assistantmoduls "Breanos Connectors"

FKE

7. November 2018

# Inhaltsverzeichnis

# 1 Beschreibung

Breanos Connectors ist eine Solution die als Sammlung unterschiedlicher Nuget-Packages angelegt wurde die im Themenbereich der Übermittluing von Daten über unterschiedliche Übertragungswege angesiedelt sind. Dazu gehören:

- ActiveMqConnector
- BreanosConnectors.Interface
- BreanosConnectors.Kpu.Communication.Common
- BreanosConnectors.Kpu.Communication.Utilities
- BreanosConnectors.SerializationHelper / .Standard
- BreanosConnectors.Utilities
- OpcUaConnector

**ActiveMqConnector** Dies ist eine Kapselung des ActiveMq Clients zur Verbindung zu einem ActiveMq-Server. Der ActiveMqConnector implementiert die Interfaces von BreanosConnectors.Interface zur Offenlegung seiner Funktionalitäten.

**BreanosConnectors.Interface** Interfaceprojekt für ActiveMqConnector und eventuelle zukünftige weitere Konnektoren.

**BreanosConnectors.Kpu.Communication.Common** Sammlung unterschiedlicher Klassen die zur Kommunikation mit KPUs standardmäßig verwendet werden, z.B. Model Update.

**BreanosConnectors.Kpu.Communication.Utilities** Beinhaltet eine Weiterentwicklung des Batchers aus BreanosConnectors.Utilities zur gesammelten, strukturierten Weitergabe von Model Updates.

**BreanosConnectors.SerializationHelper / .Standard** Kapselung des Xml-Serializers von Microsoft zur direkten, einfachen Verwendung mit Strings zur Deserialisierung bzw. typisierten Objekten zur Serialisierung.

**BreanosConnectors.Utilities** Beinhaltet die Batcherklassen die sich auch in der Assistant-Solution befinden.

**OpcUaConnector** Kapselung für OPCFoundation.NetStandard.Opc.Ua opc client.

## 2 Verwendung

Zur Verwendung dieser Projekte können diese entweder direkt als Abhängigkeiten in ein bestehendes Projekt eingebunden werden oder aber als Nugetpakete vom öffentlichen Nuget Server von Breanos heruntergeladen werden.

**ActiveMqConnector** Listing 1 zeigt die rudimentäre Verwendung des ActiveMqConnectors mit Registrierung für Nachrichten von der Queue mmyQueue".

```
using System;
using System.Threading.Tasks;
    class Program
{
  static void Main(string[] args)
  {
    BreanosConnectors.Interface.IMqConnector connector = new
    BreanosConnectors.ActiveMqConnector.Connector();
    InitConnector(connector);
    Console.ReadLine();
  }
  public static async Task InitConnector(BreanosConnectors.
    Interface.IMqConnector connector)
  {
    await connector.ConnectAsync("tcp:activemq://127.0.0.1:61616
    ", "admin", "admin");
    connector.Message += OnIncomingActiveMqMessage;
    await connector.ListenAsync("myQueue");

  }

  private static void OnIncomingActiveMqMessage(object sender,
    BreanosConnectors.Interface.OnMessageEventArgs e)
  {
    Console.WriteLine(e.Content);
  }
}
```

Listing 1: ActiveMqConnector example

**SerializationHelper** Listing 2 zeigt die Verwendung des SerializationHelpers zur Serialisierung einer gegebenen Klasse, der Output des Programs wird in Listing 3 dargestellt.

```
using System;
using System.Threading.Tasks;
public class DataStorageClass
{
  public int MyInteger { get; set; }
  public string MyString { get; set; }
```

```
7     public DataStorageClass2 InternalSomething { get; set; }
8   }
9   public class DataStorageClass2
10  {
11    public double SomeDouble { get; set; }
12  }
13  class Program
14  {
15    static void Main(string[] args)
16    {
17      var masterStorage = new DataStorageClass()
18      {
19        MyInteger = 42,
20        MyString = "Hello World",
21        InternalSomething = new DataStorageClass2()
22        {
23          SomeDouble = 3.1415926
24        }
25      };
26      var serialized = BreanosConnectors.SerializationHelper.
        Serialize(masterStorage);
27      Console.WriteLine(serialized);
28    }
29  }
```

Listing 2: SerializationHelper code example

```
1  <?xml version="1.0" encoding="utf-16"?>
2  <DataStorageClass xmlns:xsi="http://www.w3.org/2001/XMLSchema-
       instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema">
3  <MyInteger>42</MyInteger>
4  <MyString>Hello World</MyString>
5  <InternalSomething>
6  <SomeDouble>3.1415926</SomeDouble>
7  </InternalSomething>
8  </DataStorageClass>
9  Press any key to continue . . .
```

Listing 3: SerializationHelper code example

# 3 Installation

Siehe Abschnitt 2

# 4 Konfiguration

Die Komponenten werden rein über ihre Codingschnittstellen konfiguriert und gesteuert.

## 5 Abhängigkeiten

Ggf. vom öffentlichen Nugetserver abrufbare Nugetpakete die in den einzelnen Projekten benötigt werden.

## 6 Anforderungen

**ActiveMqConnector** Benötigt zum sinnvollen Einsatz einen laufenden ActiveMq Server.

**OpcUaConnector** Benötigt zum sinnvollen Einsatz einen laufenden Opc UA Server.

# Listings