new solutions GmbH · Marktler Straße 61 · D-84489 Burghausen · Telefon +49 (0) 8677. 8776-0 · Fax +49 (0) 8677. 8776-11 · http://www.new-solutions.com

new solutions
business software

Technical Concept

Interface description for Finito Digital Shift Book

With an example of a PI Server connection

new solutions
business software

1. History

Version	Date	Editor	Change
1.0	25.01.2017	Kreutzberg	Creation
1.1	02.02.2017	Hüsch	Review

2. Objective

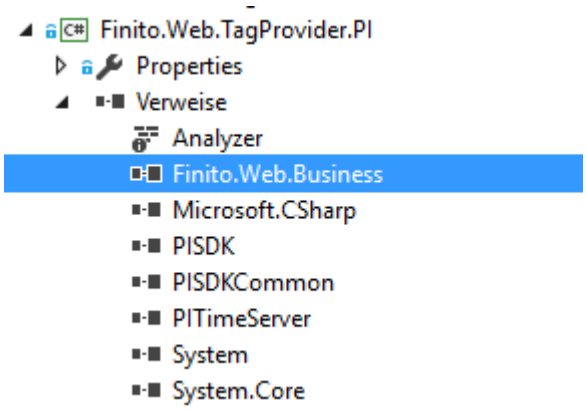
The Finito Digital Shift Book application is to be linked to an external system and will receive various data from it via interfaces.
In this example, a PI Server connection will be developed for Finito.
The received data can be used to fill the universal fields in the Finito application.

3. Method

3.1. Creation of a new project for web applications

First, create a new project in Visual Studio 2015 or higher.
Use the .NET Framework 4.5.2 as the target framework.
The project has to integrate the file *Finito.Web.Business.dll*. This file can be found in your Finito installation directory.

Additionally, you'll have to integrate some external components. In our example of a PI interface, those are *PISDK.dll, PISDKCommon.dll*, and *PITimeServer.dll*.



The *Finito.Web.Business.dll* contains an interface called **ITagProvider** which provides three methods: **GetBoolValue**, **GetDoubleValue,** and **GetStringValue**.
To build a new interface for an external system, a new class needs to be created. For this example, we'll call it **PITagProvider**. This class also implements another interface, called **IDisposable**.

```
namespace Finito.Web.TagProvider.PI
{
    public class PITagProvider : ITagProvider, IDisposable
    {
    //[...]
    }
}
```

The **IDisposable** interface provides a mechanism for releasing unmanaged resources, and the **ITagProvider** interface defines the three methods above, which we'll need later.
Those methods have to be implemented depending on the requirements of your application.
For the PI Server example, we will set up a connection that creates a snapshot value and returns it as either a boolean, a double, or a string.

```
public bool? GetBoolValue(string tagName, string connectionString, string username,
string password)
{
     piServer = sdk.Servers[connectionString];
     try
     {
          return Convert.ToBoolean(readSnapshotValue(tagName));
     }
     catch { }
     return null;
 }

public double? GetDoubleValue(string tagName, string connectionString,
string username, string password)
{
     piServer = sdk.Servers[connectionString];
     try
     {
          return Convert.ToDouble(readSnapshotValue(tagName));
     }
     catch { }
     return null;
 }

public string GetStringValue(string tagName, string connectionString, string username,
string password)
{
      piServer = sdk.Servers[connectionString];
      try
     {
         return Convert.ToString(readSnapshotValue(tagName));
     }
     catch { }
     return null;
 }
```

3.2. Database configuration

In addition to the web application, we need to make some modifications to the Finito database. The already existing tables **TagServerDriver** and **TagServer** need new entries in the database.

In our PI Server example, we use the following Insert Scripts.

```
INSERT INTO [dbo].[TagServerDriver]
           ([TagServerDriverId]
           ,[Label]
           ,[AssemblyName]
           ,[ReflectionType]
           ,[InsertBy]
           ,[InsertDate]
           ,[UpdateBy]
           ,[UpdateDate]
           ,[DeleteBy]
           ,[DeleteDate])

     VALUES
           (NEWID()
           ,'New Solutions Demo Driver'
           ,'Finito.Web.TagProvider.PI'
           ,'Finito.Web.TagProvider.PI.PITagProvider'
           ,'New Solutions'
           ,'2017-01-01'
           ,NULL
           ,NULL
           ,NULL
           ,NULL)
GO
```

TagServerDriverId: New ID which will also be needed in the TagServer table

Label: Caption of the driver

AssemblyName: Label of the namespace of the .cs-file, where the functionalities of the interfaces have been implemented

ReflectionType: This label is a concatenation of the namespace and the label of the created class

new solutions
business software

```
INSERT INTO [dbo].[TagServer]
           ([TagServerId]
           ,[Server]
           ,[Label]
           ,[TagServerDriverId]
           ,[ConnectionString]
           ,[AuthenticationData]
           ,[Password]
           ,[InsertBy]
           ,[InsertDate]
           ,[UpdateBy]
           ,[UpdateDate]
           ,[DeleteBy]
           ,[DeleteDate])
     VALUES
           (NEWID()
           ,'NewSolDemo'
           ,'Demo'
           ,TagServerDriverId
           ,ConnectionString
           ,NULL
           ,NULL
           ,'New Solutions'
           ,'2017-01-01'
           ,NULL
           ,NULL
           ,NULL
           ,NULL)
    GO
```

TagServerId: New ID

Server: Name of the server

Label: Caption of the server

TagServerDriverId: The same ID as the one created in the **TagServerDriver** table

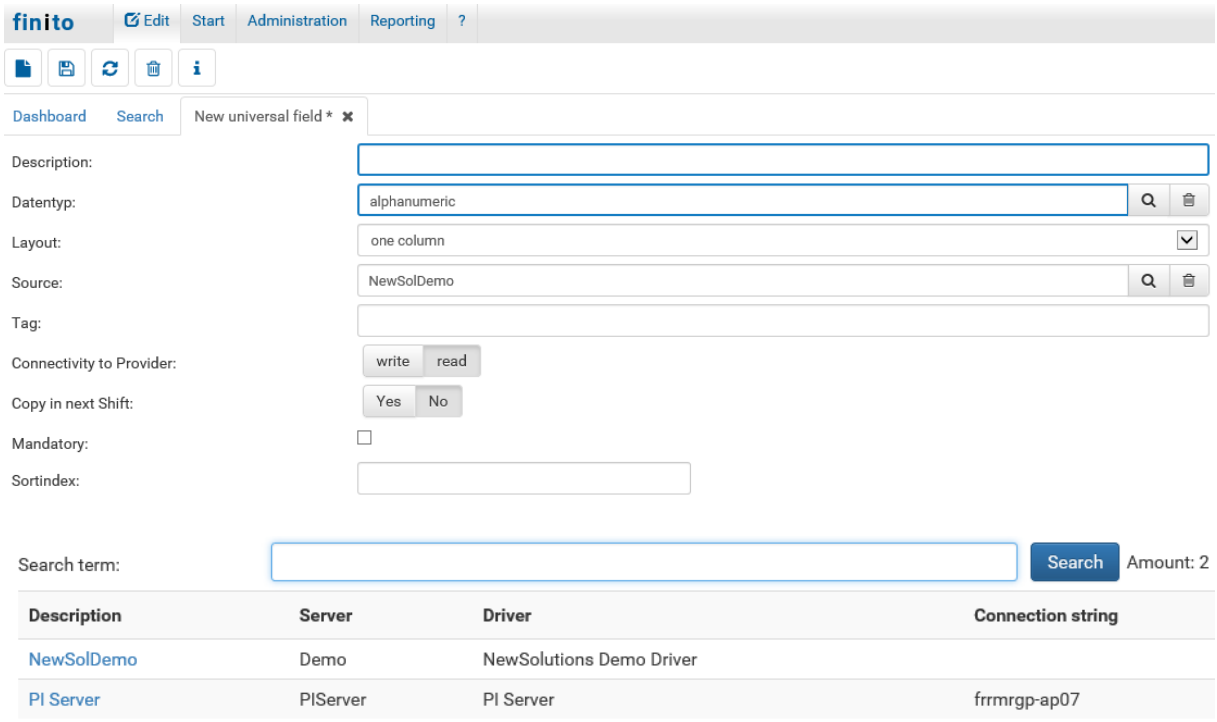
Connection String: Connection string to the respective server or SDK

Authentication Data: Credentials if needed, else NULL

Password: Password if needed, else NULL

4. Usage in Finito

In the Finito administration section, a new universal field will be created.
The new PI Server is located under Source.



When you're done configuring the universal fields, they will be filled automatically after clicking on the Receive Data button in the shift report.