

# PHmi Quick start guide

---

## 1 Installation

For using PHmi following requirements should be satisfied:

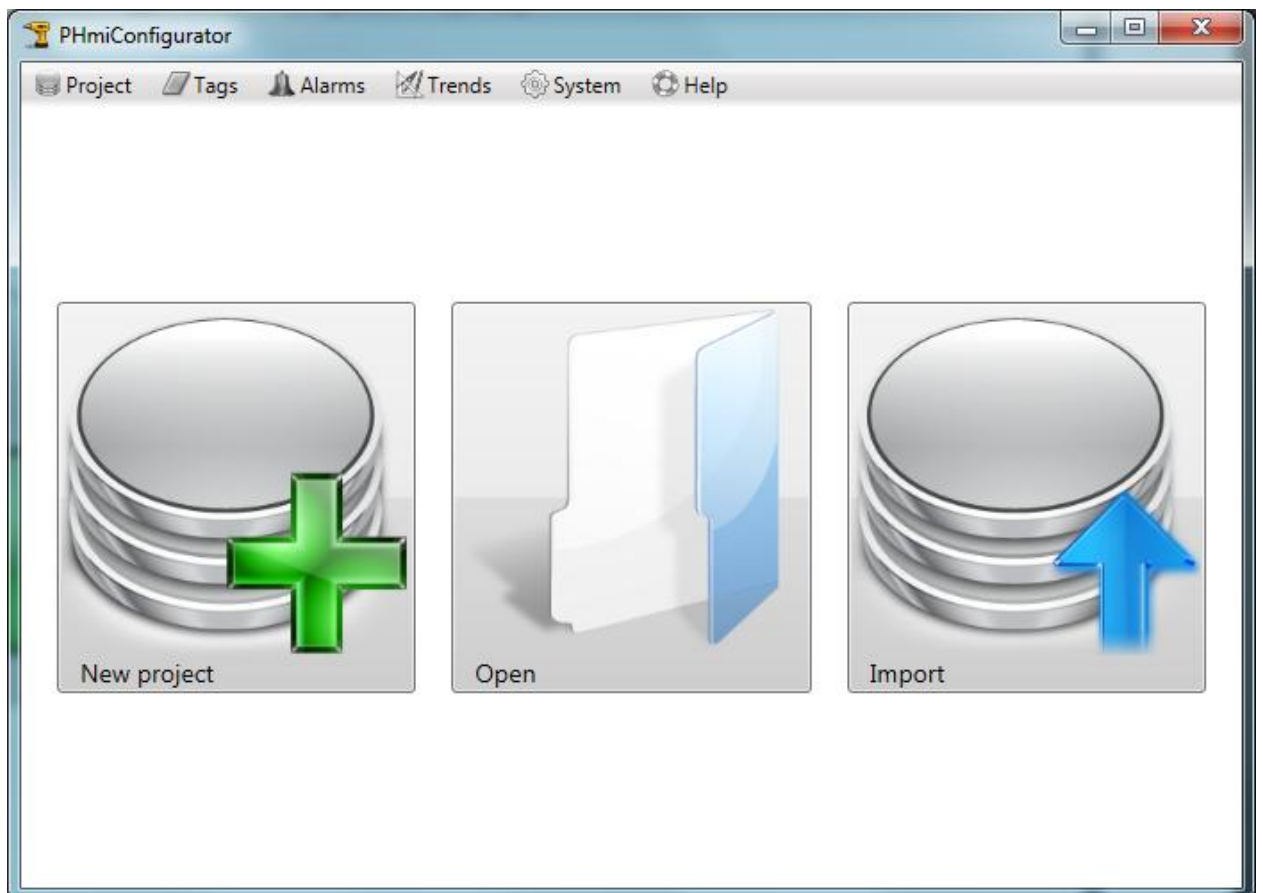
- 1) PHmi.
- 2) The world's most advanced open source database PostgreSQL v. 9.2 or higher (it can be downloaded at [www.postgresql.org](http://www.postgresql.org)).
- 3) Microsoft Visual Studio 2010 or higher (express version for windows desktop developing can be downloaded at [www.microsoft.com](http://www.microsoft.com)).

## 2 Developing project

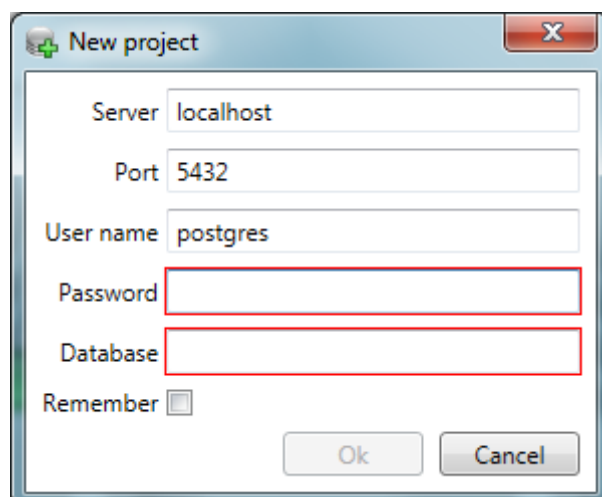
Project developing can be divided into two phases: configuring server and developing project.

### 2.1 Configuring server

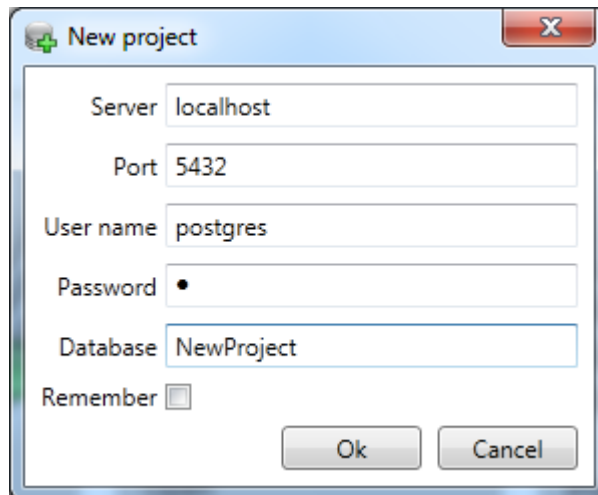
Let's start PHmiConfigurator.exe. The following window should appear.



Press the "New project" button. The "New project" window should appear.



Enter connection parameters to PostgreSQL and desired database name.

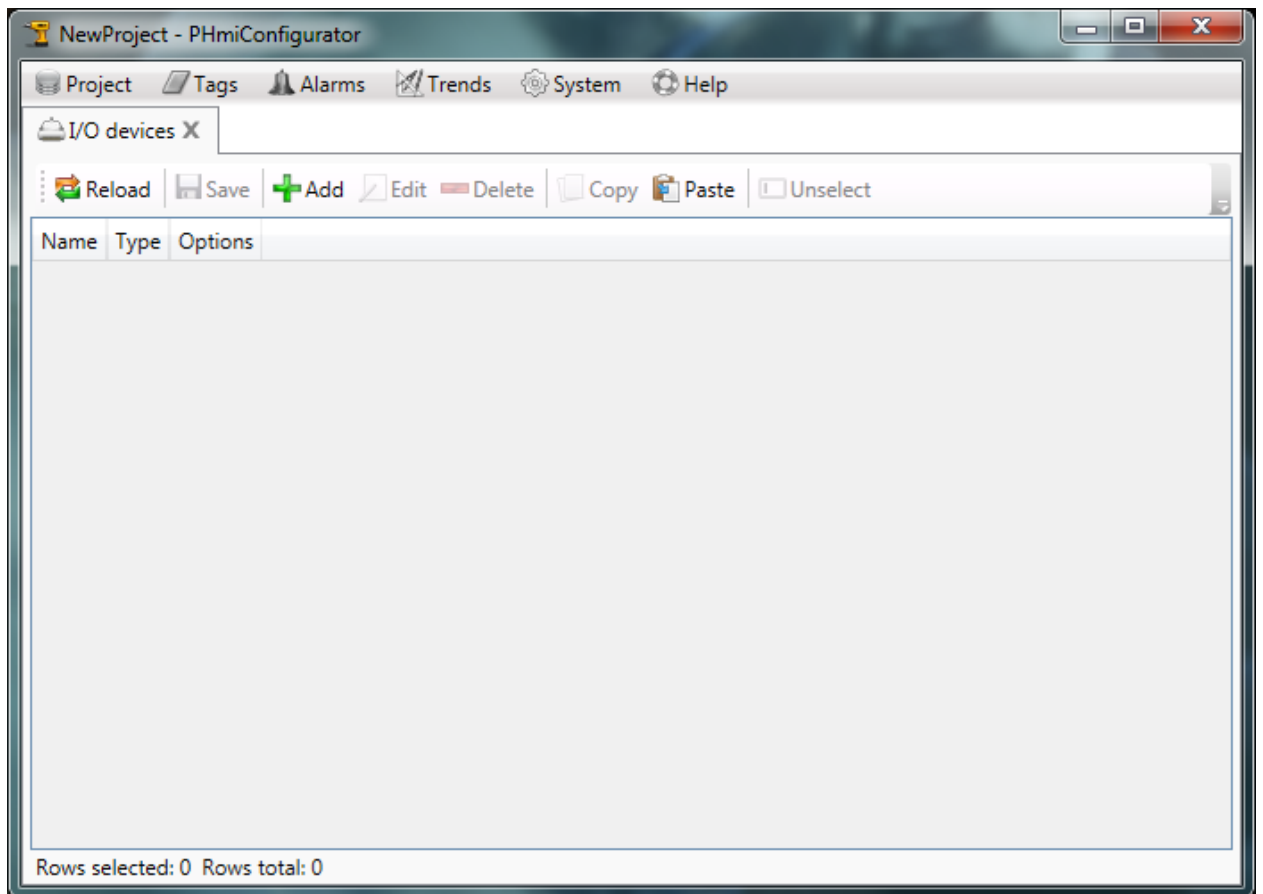


A dialog box titled "New project" with a close button (X) in the top right corner. It contains several input fields: "Server" with "localhost", "Port" with "5432", "User name" with "postgres", "Password" with a masked character (•), and "Database" with "NewProject". There is also a "Remember" checkbox which is unchecked. At the bottom right are "Ok" and "Cancel" buttons.

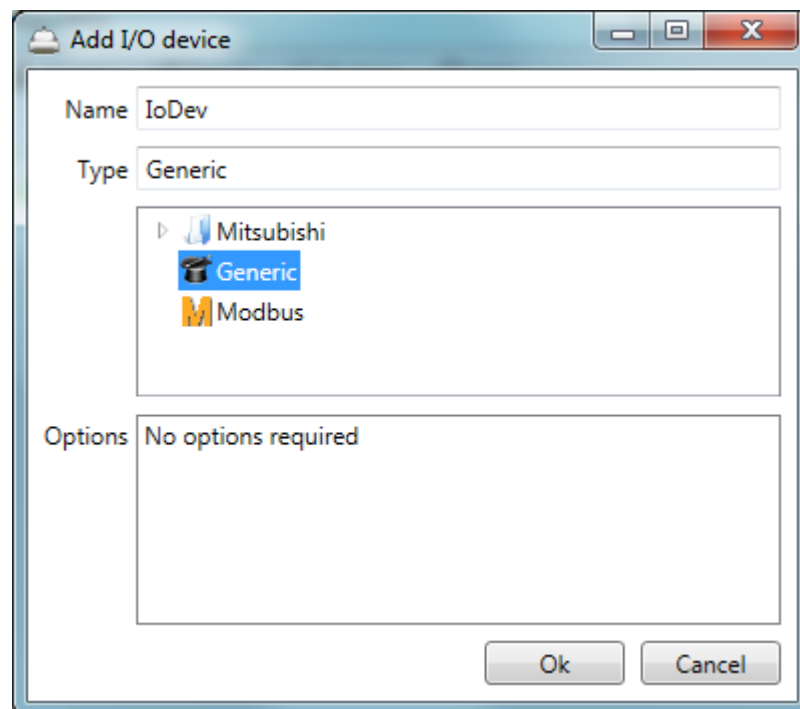
After pressing "Ok" new project will be created.

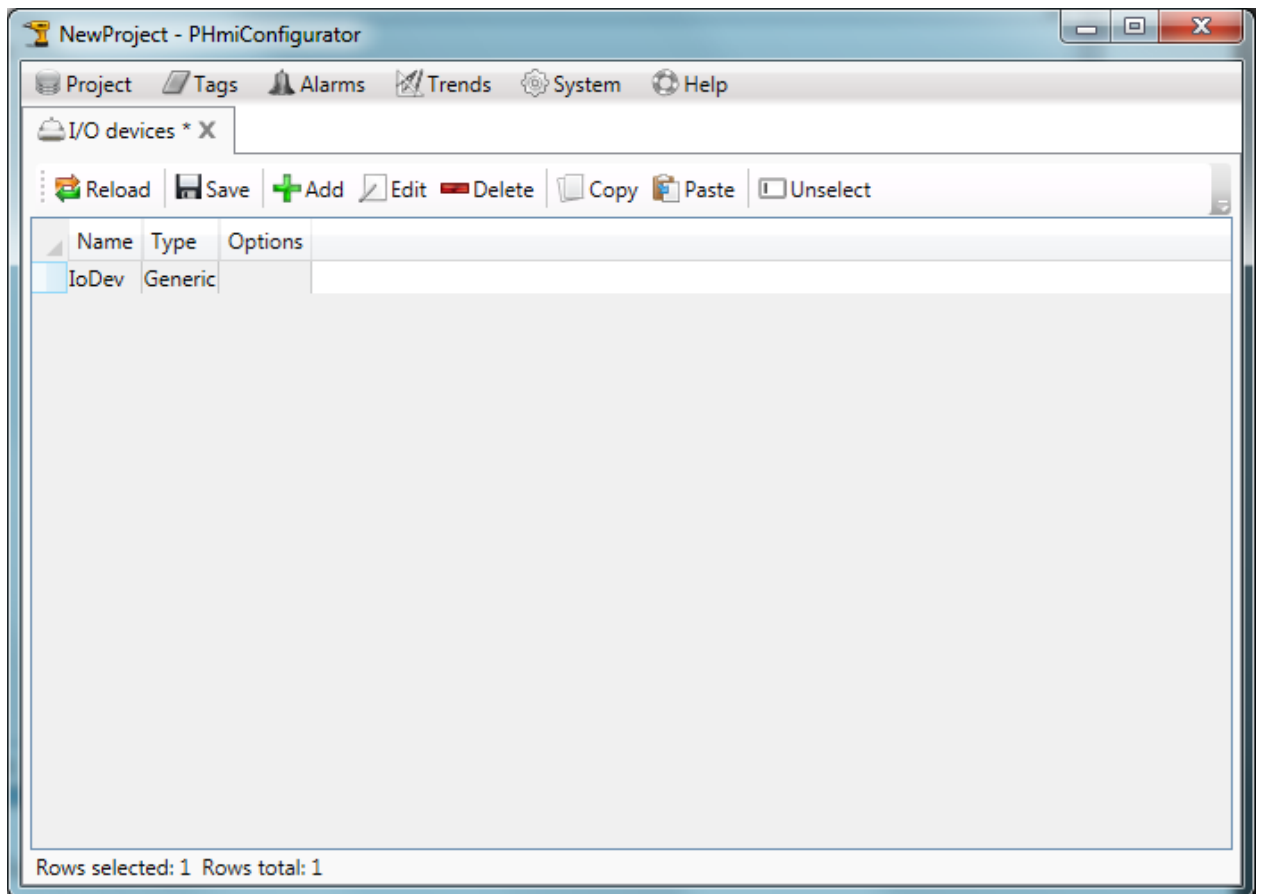


Press the "I/O devices" button. It will open "I/O devices" tab in the configurator.



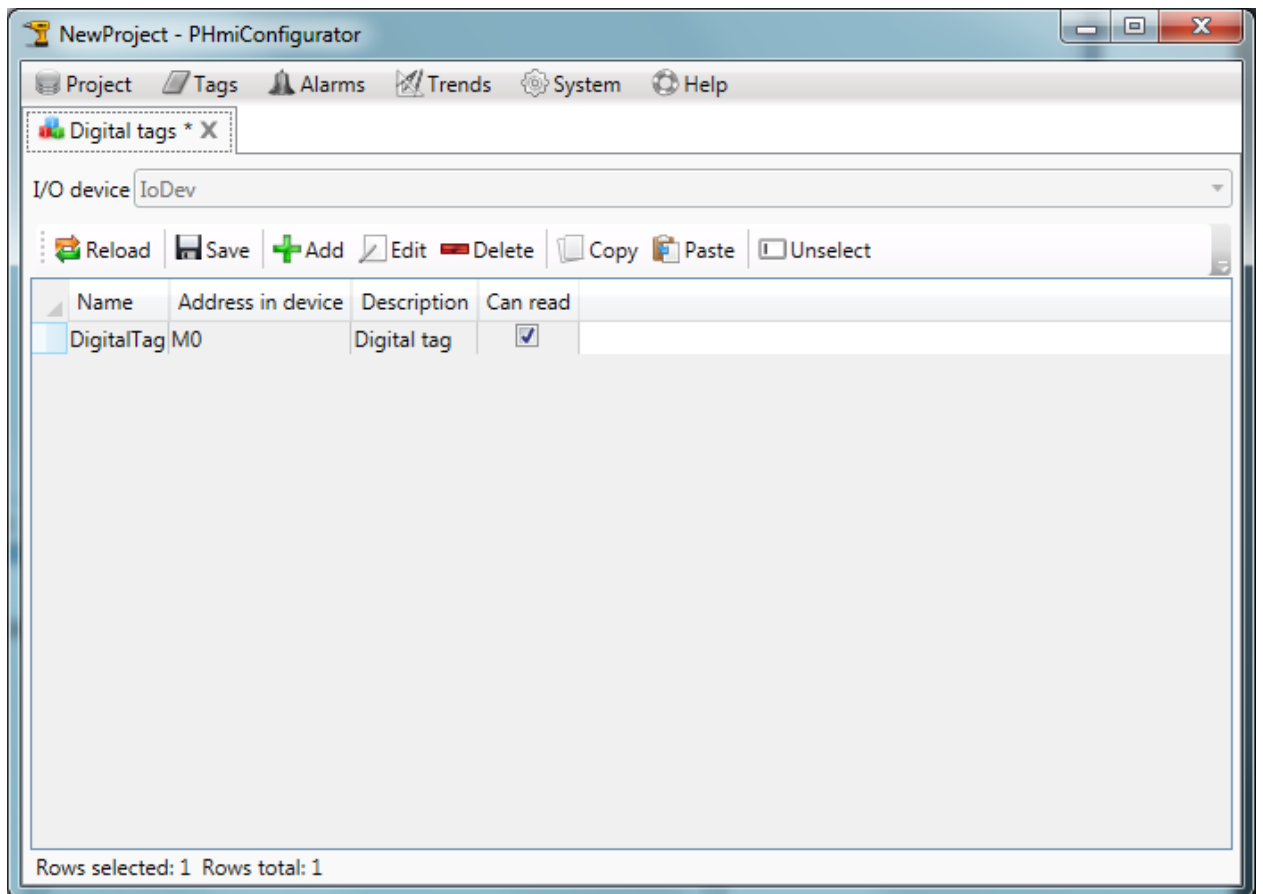
Add a new I/O device “IoDev” of type “Generic”. Generic is an I/O device for testing purposes. It’s used when it’s impossible to connect to a real device. It supports any tag names.





Press the save button and close the tab.

Let's open the "Digital tags" tab and create a new tag.



Let's open the "Numeric tags" tab and create a new tag.

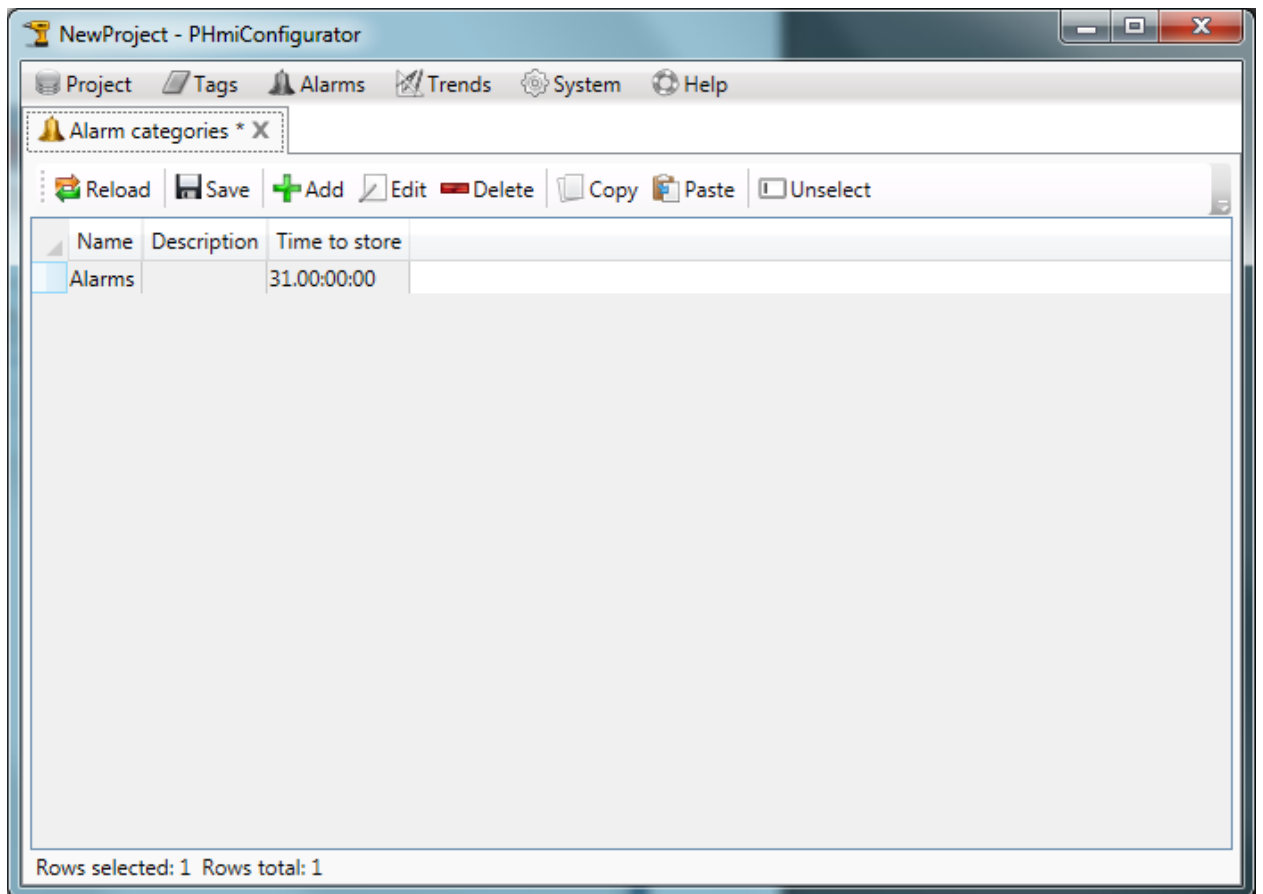
The 'Add numeric tag' dialog box is shown with the following fields and values:

- Name: NumericTag
- Address in device: D0
- Description: Numeric tag
- Can read: ☒
- Tag type: Int16
- Format: 0.0
- Engineer unit: A
- Raw min: 0
- Raw max: 10000
- Eng min: 0
- Eng max: 1000

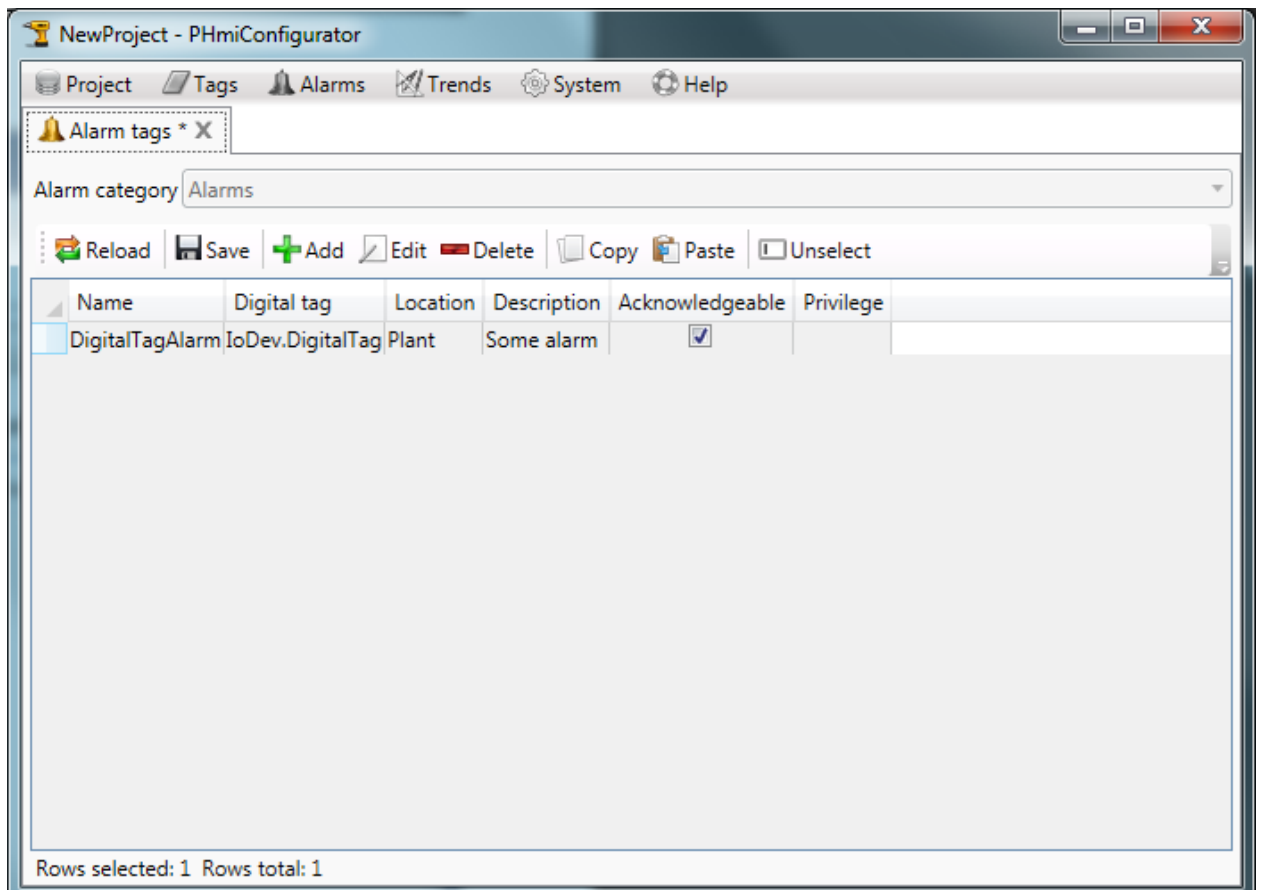
Buttons for 'Ok' and 'Cancel' are at the bottom right.

Note that description, format, engineer unit, scale boundaries are not mandatory and can be omitted.

Let's create a new alarm category.

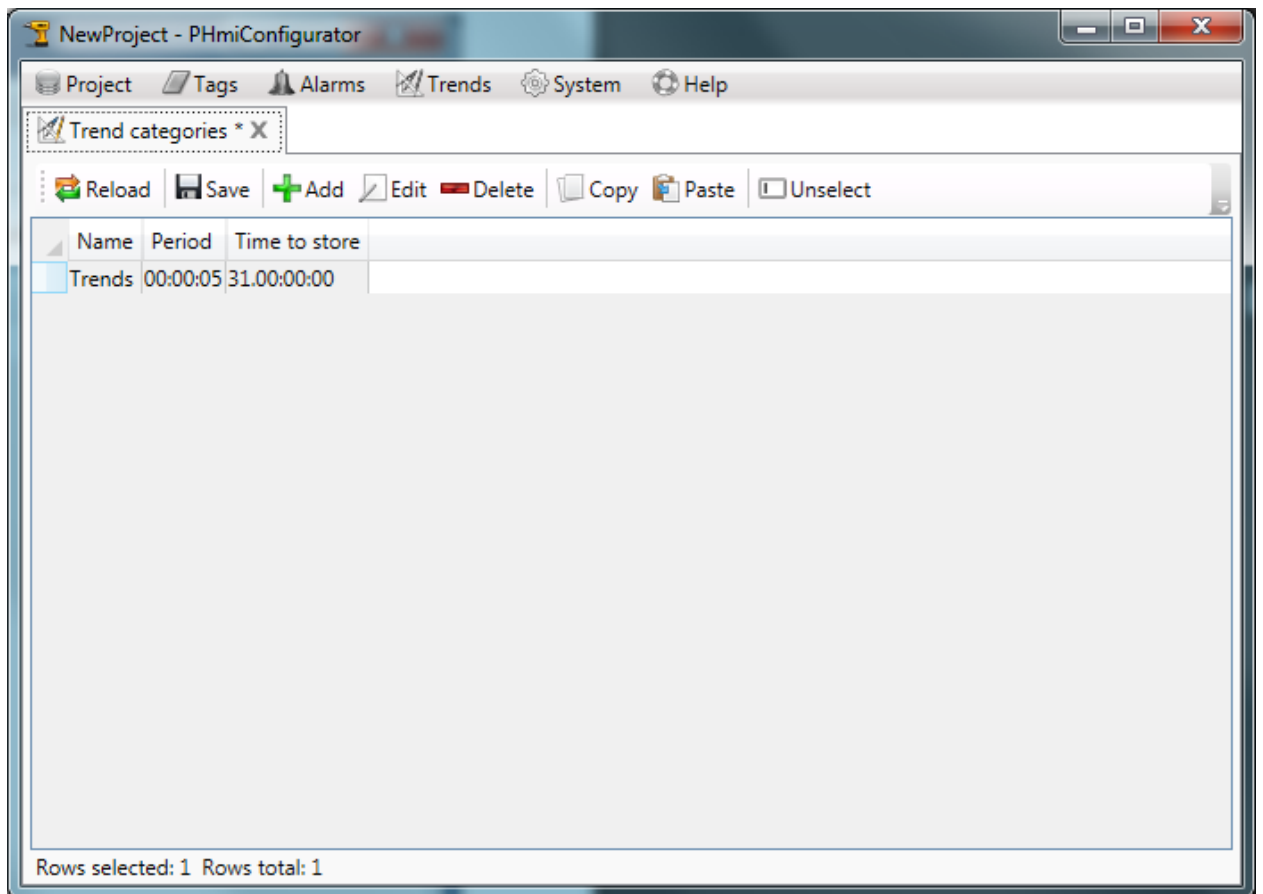


Let's create a new alarm tag.

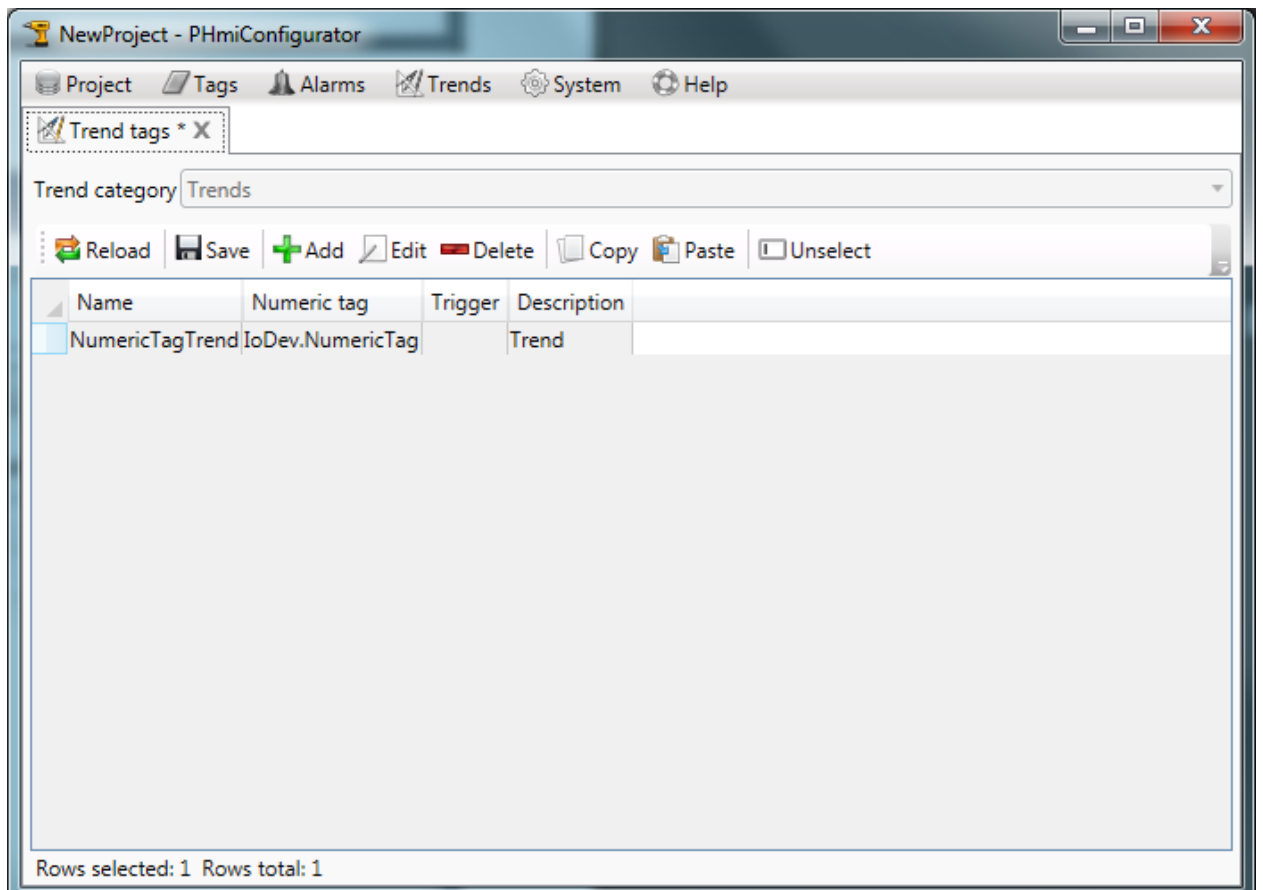


Let's create a new trend category.



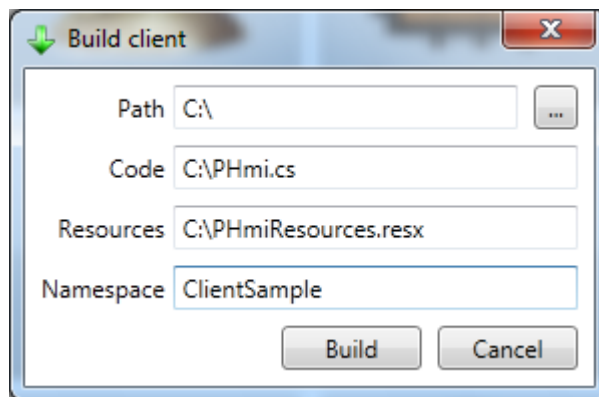


Let's create a new trend tag.



## 2.2 Developing project

In the configurator press the “Build client” button.

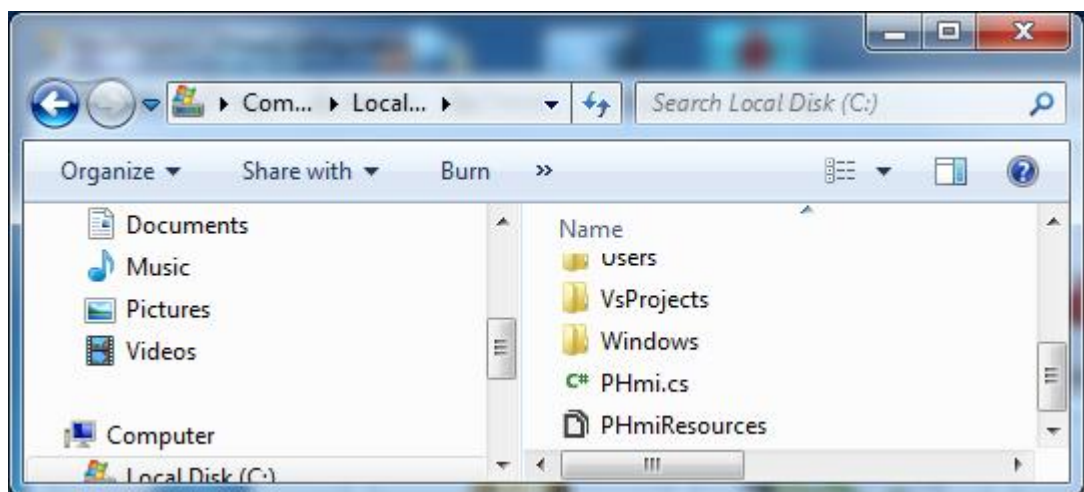


Choose the path for files to create.

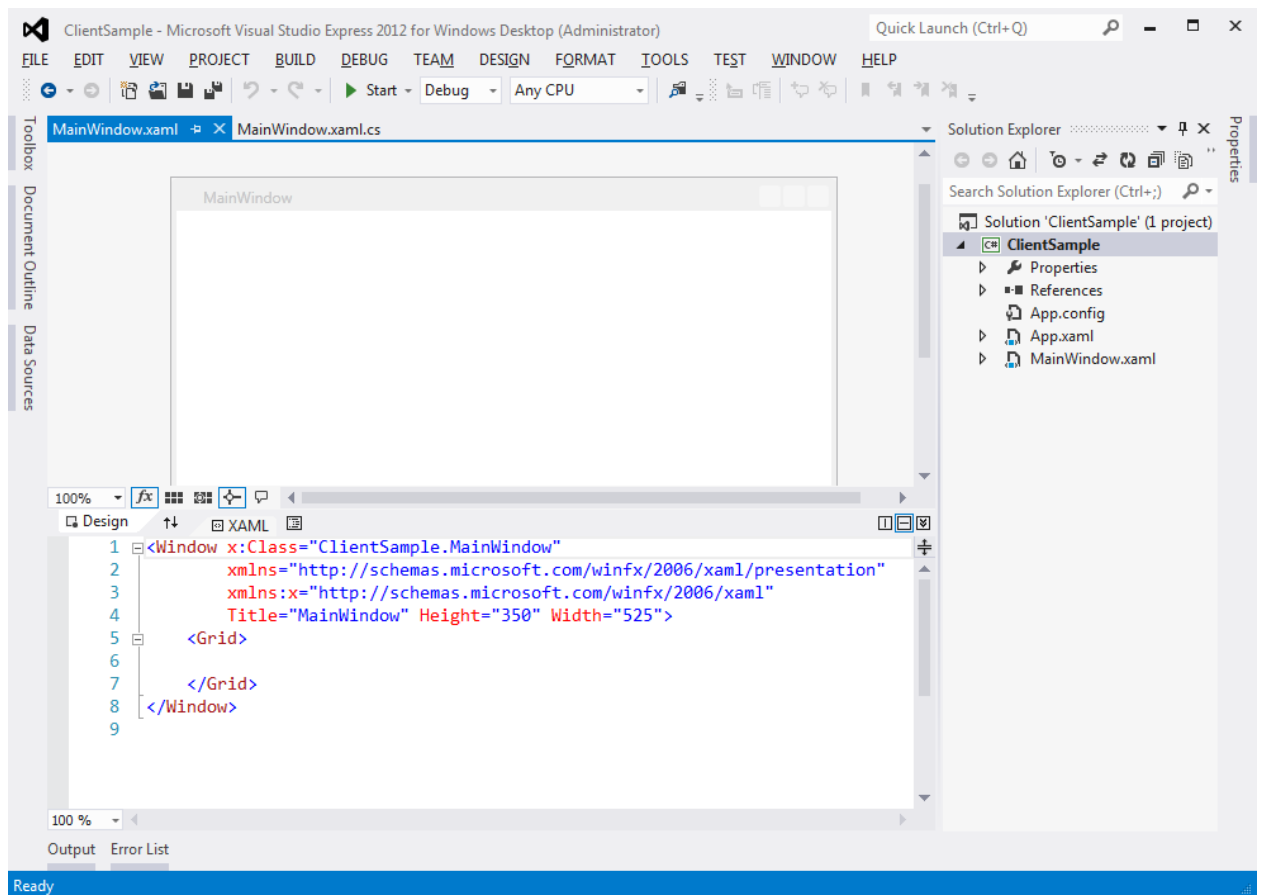
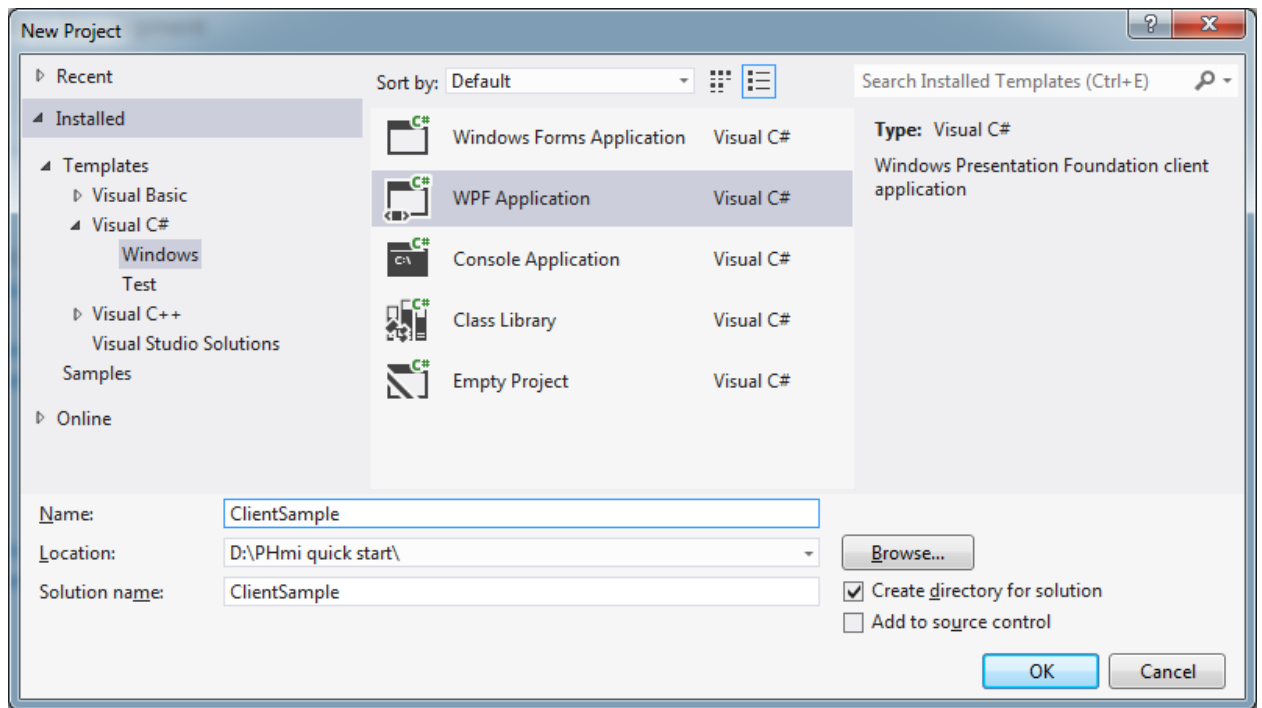
Namespace should be the name of the future client project.

Click the “Build” button.

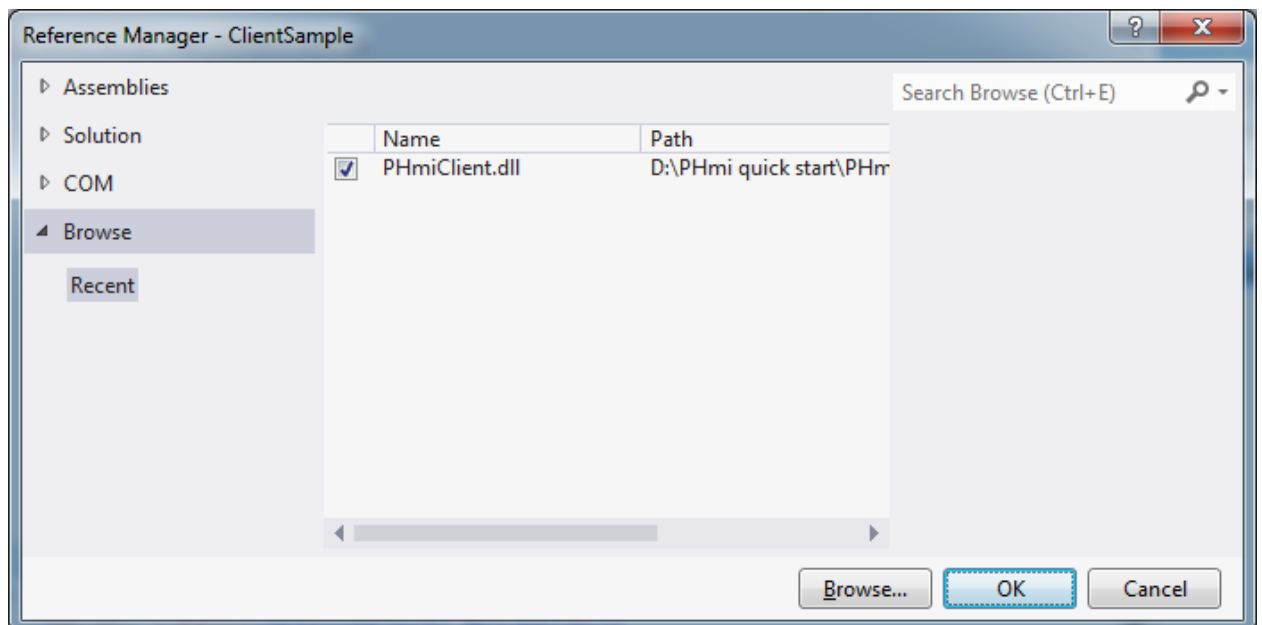
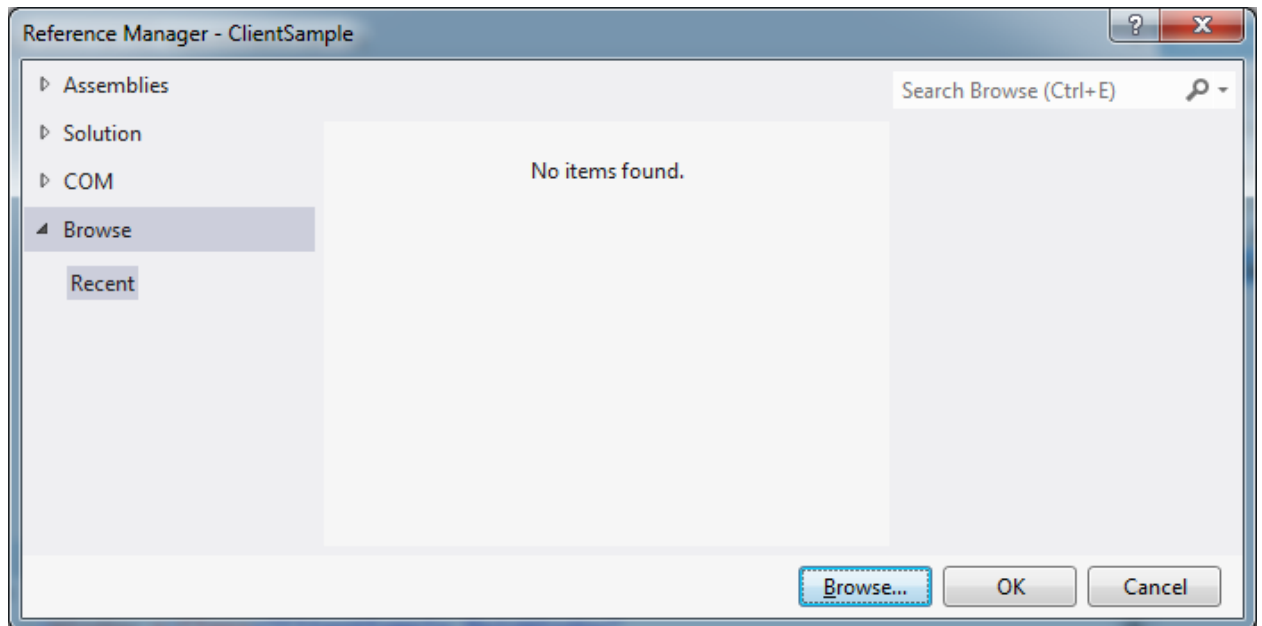
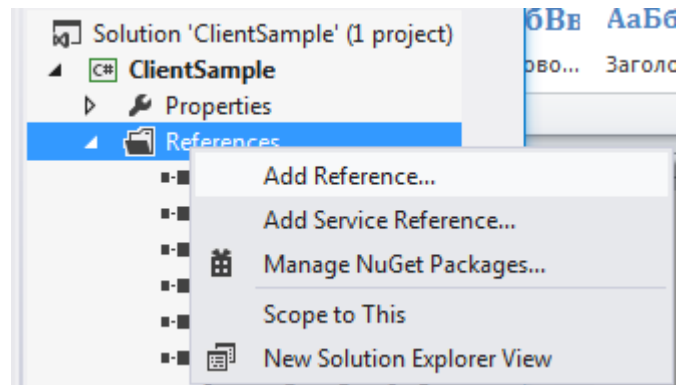
The files should be present at the specified path.



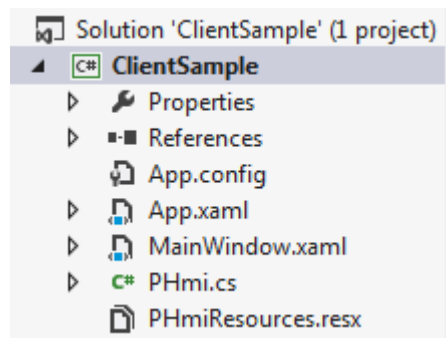
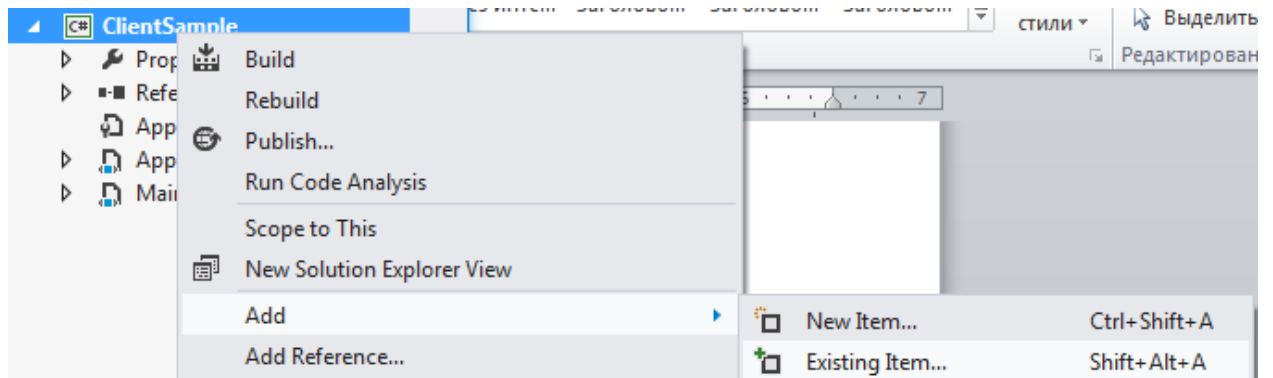
Start Microsoft Visual Studio and create a new WPF project. Target framework should be “.Net Framework 4.0” or higher.



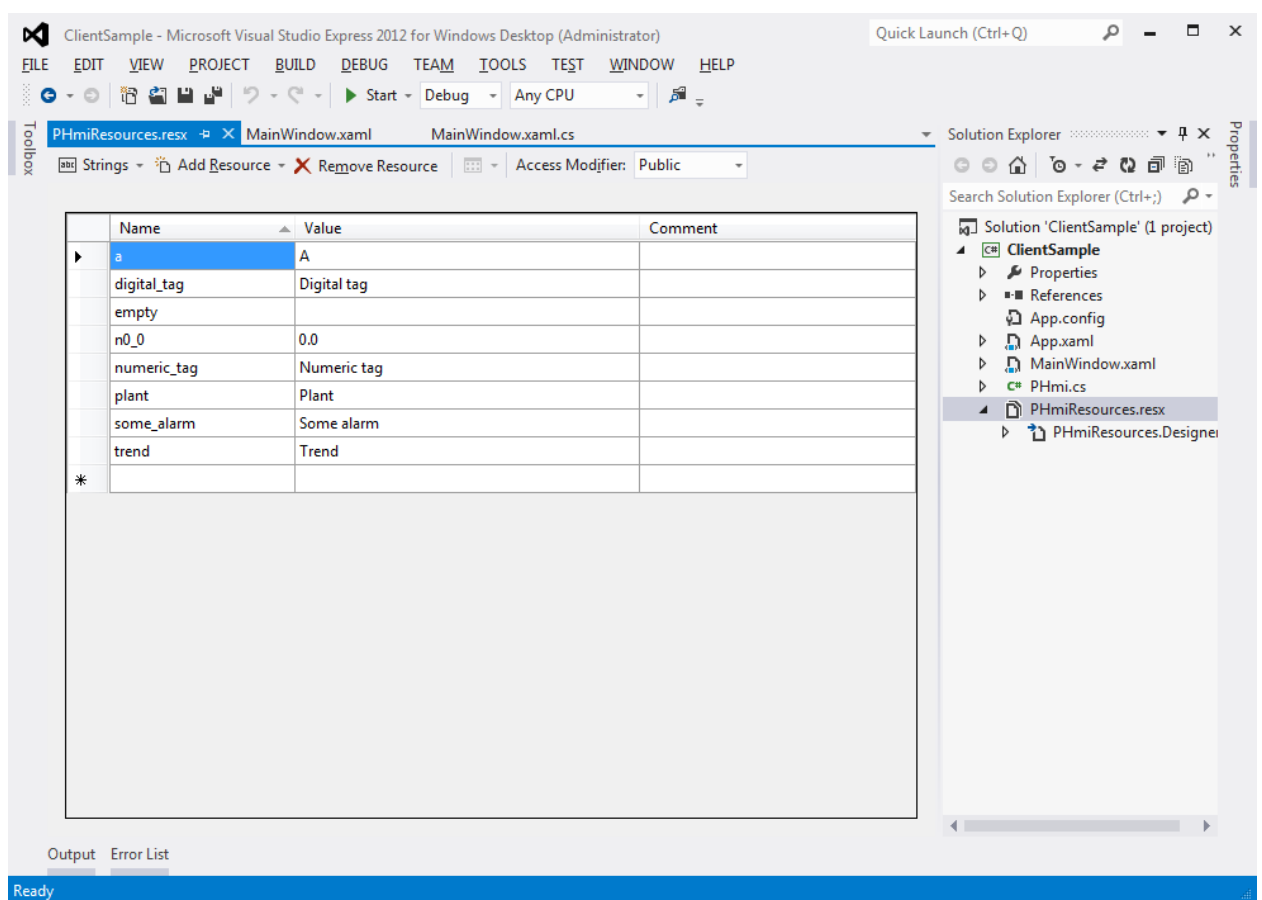
Add the reference to the PHmiClient.dll.



Add PHmi.cs and PHmiResources.resx created earlier.



Double click the PHmiResources.resx and change the “Access modifier” to Public.



Let's edit the MainWindow.xaml. Add the Root control to the Grid.

```

<Window x:Class="ClientSample.MainWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        xmlns:pHmiControls="clr-namespace:PHmiClient.Controls;assembly=PHmiClient"
        Title="MainWindow" Height="350" Width="525">
    <Grid>
        <pHmiControls:Root/>
    </Grid>
</Window>

```

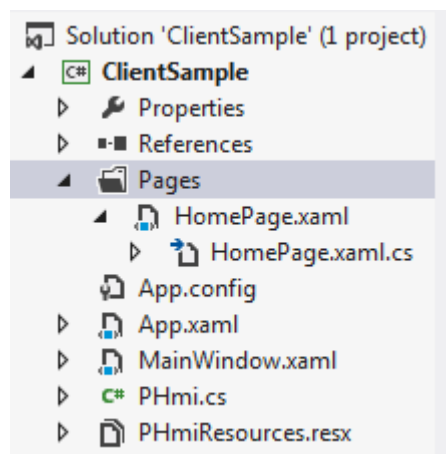
Bind PHmi to the Root's DataContext. For that create new PHmi object in the window resources.

```

<Window x:Class="ClientSample.MainWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        xmlns:pHmiControls="clr-namespace:PHmiClient.Controls;assembly=PHmiClient"
        xmlns:clientSample="clr-namespace:ClientSample"
        Title="MainWindow" Height="350" Width="525">
    <Window.Resources>
        <clientSample:PHmi x:Key="PHmi"/>
    </Window.Resources>
    <Grid>
        <pHmiControls:Root DataContext="{StaticResource PHmi}"/>
    </Grid>
</Window>

```

Add Pages folder for storing pages. Add HomePage UserControl.



HomePage should implement IPage interface. There's HomePage.xaml.cs listing below.

```

using PHmiClient.Controls.Pages;
using System;
using System.Windows.Controls;

namespace ClientSample.Pages
{
    /// <summary>
    /// Interaction logic for HomePage.xaml
    /// </summary>
    public partial class HomePage : UserControl, IPage
    {
        public HomePage()
        {
            InitializeComponent();
        }

        public object PageName
        {
            get { return "Home page"; }
        }

        public PHmiClient.Controls.IRoot Root
        {
            get; set;
        }
    }
}

```

PageName will be displayed at the Root as a page name.

Markup is displayed bellow.

```

<UserControl x:Class="ClientSample.Pages.HomePage"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
    xmlns:clientSample="clr-namespace:ClientSample"
    mc:Ignorable="d"
    d:DesignHeight="300" d:DesignWidth="300"
    d:DataContext="{d:DesignInstance clientSample:PHmi,
IsDesignTimeCreatable=True}">
    <Grid>
        <StackPanel>
            <TextBlock Text="{Binding Path=IoDev.DigitalTag.Value}"/>
            <TextBlock Text="{Binding Path=IoDev.NumericTag.ValueString}"/>
        </StackPanel>
    </Grid>
</UserControl>

```

We added TextBlocks to view values of tags.

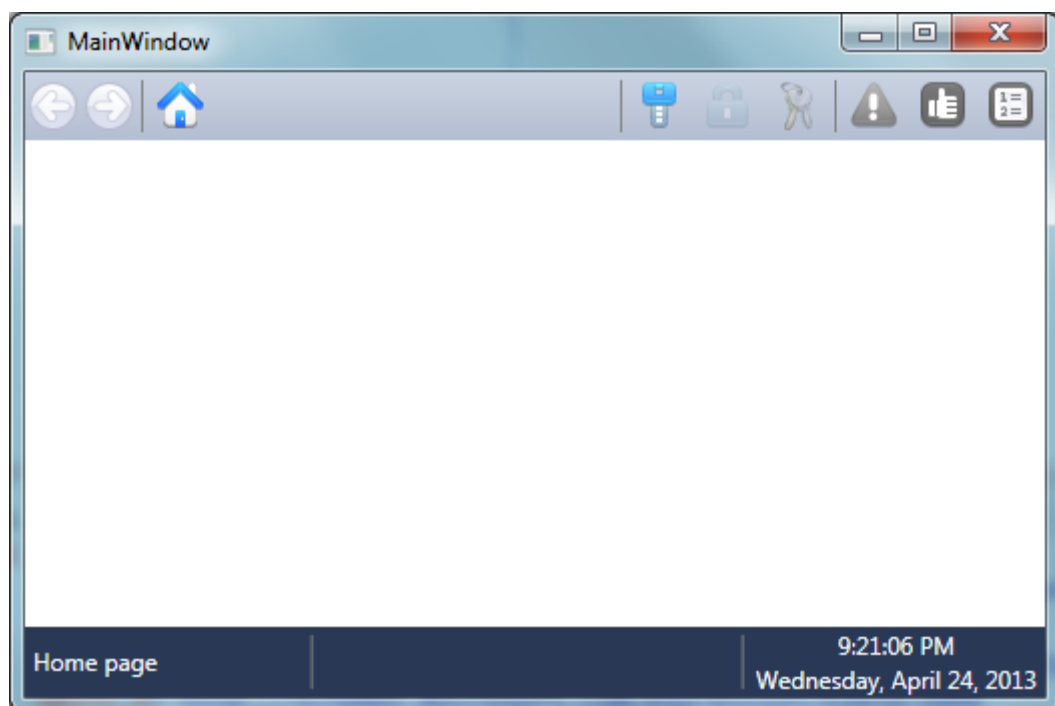
Let's bind home page type to the Root:

```

<Window x:Class="ClientSample.MainWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        xmlns:pHmiControls="clr-namespace:PHmiClient.Controls;assembly=PHmiClient"
        xmlns:clientSample="clr-namespace:ClientSample"
        xmlns:pages="clr-namespace:ClientSample.Pages"
        Title="MainWindow" Height="350" Width="525">
    <Window.Resources>
        <clientSample:PHmi x:Key="PHmi"/>
    </Window.Resources>
    <Grid>
        <pHmiControls:Root
            DataContext="{StaticResource PHmi}"
            HomePage="{x:Type pages:HomePage}"/>
    </Grid>
</Window>

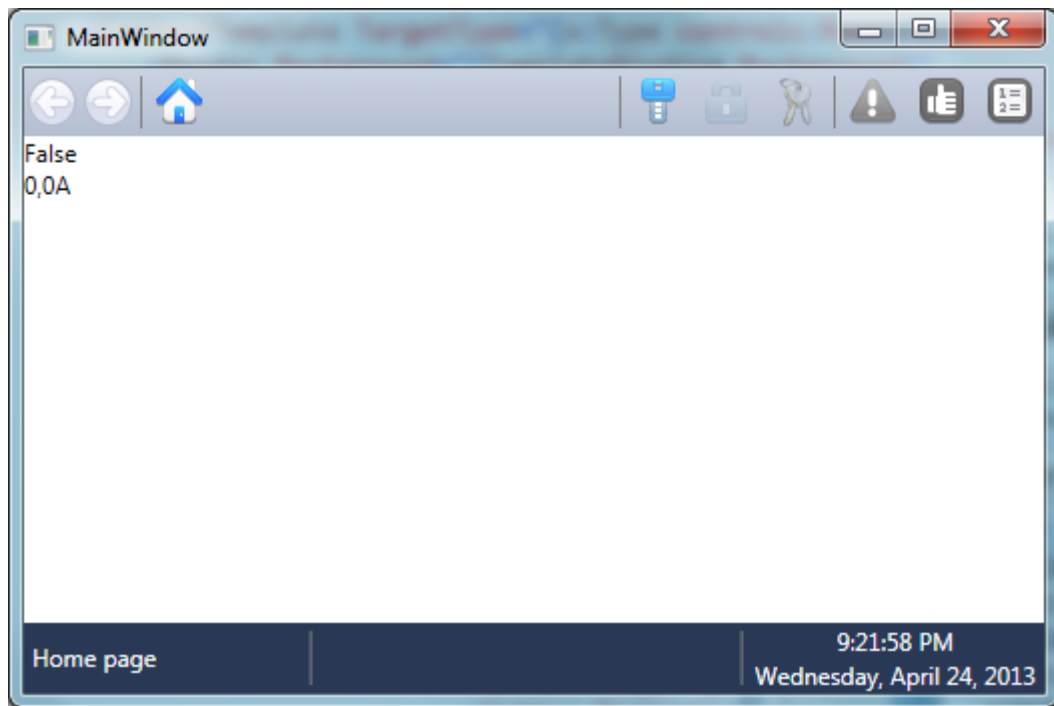
```

Run the application and see what is done.



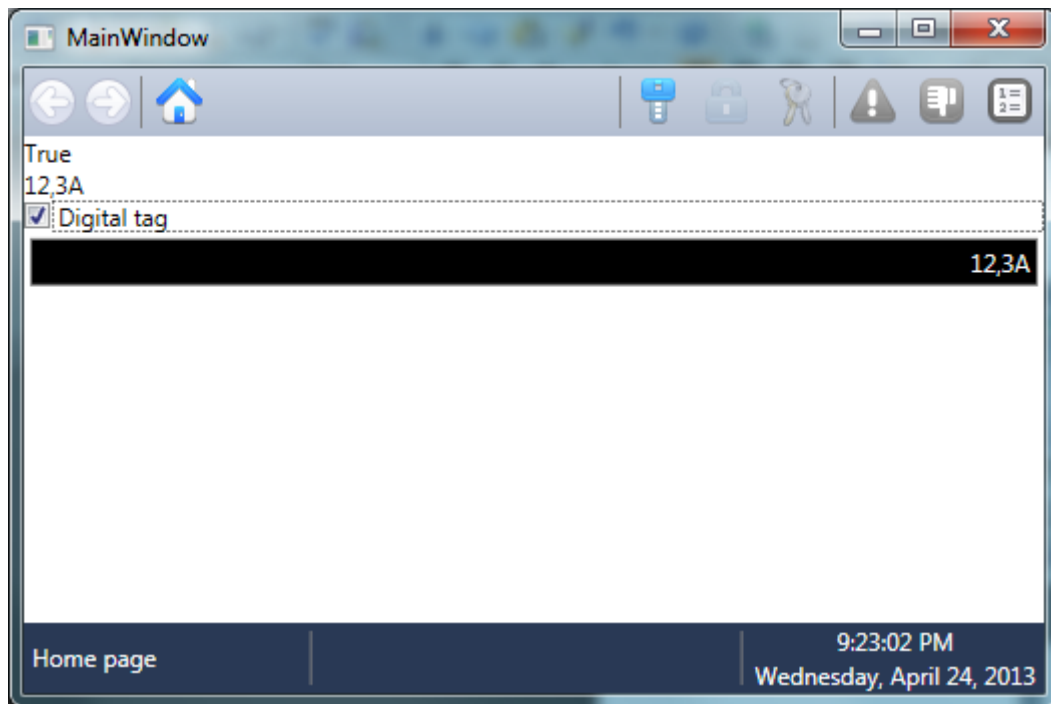
No tag values are displayed. That is because PHmiRunner.exe is not running. Press the “Run” button in the configurator.





Now add control elements to change tag's values.

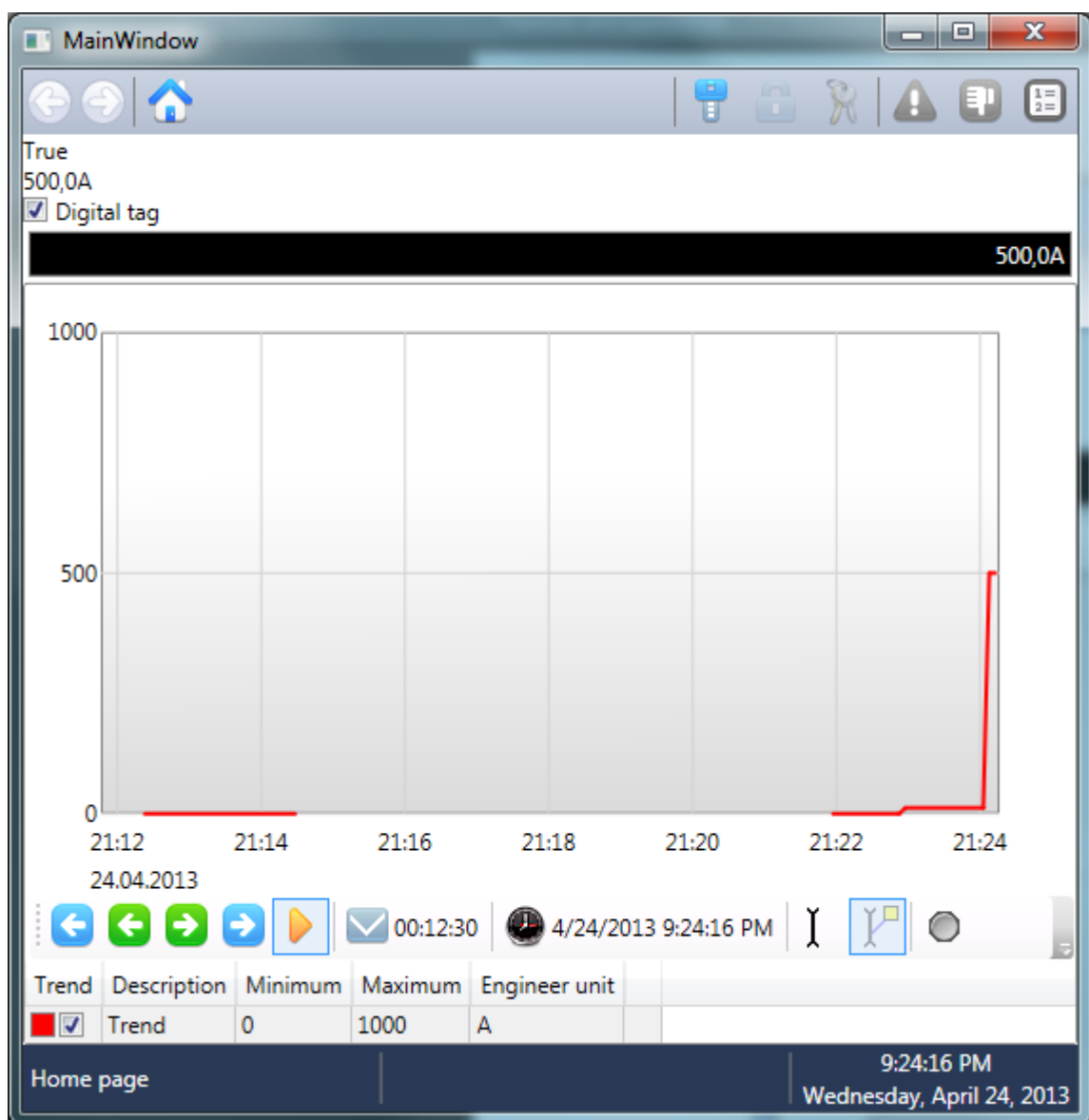
```
<UserControl x:Class="ClientSample.Pages.HomePage"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
    xmlns:clientSample="clr-namespace:ClientSample"
    xmlns:pHmiControls="clr-namespace:PHmiClient.Controls;assembly=PHmiClient"
    mc:Ignorable="d"
    d:DesignHeight="300" d:DesignWidth="300"
    d:DataContext="{d:DesignInstance clientSample:PHmi,
IsDesignTimeCreatable=True}">
    <Grid>
        <StackPanel>
            <TextBlock Text="{Binding Path=IoDev.DigitalTag.Value}"/>
            <TextBlock Text="{Binding Path=IoDev.NumericTag.ValueString}"/>
            <CheckBox
                IsChecked="{Binding Path=IoDev.DigitalTag.Value, Mode=TwoWay}"
                Content="{Binding Path=IoDev.DigitalTag.Description}"/>
            <pHmiControls:NumericInput NumericTag="{Binding Path=IoDev.NumericTag}"/>
        </StackPanel>
    </Grid>
</UserControl>
```



When we trigger the “Digital tag” alarm occurs.

Add a trend viewer to the page.

```
<UserControl x:Class="ClientSample.Pages.HomePage"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
    xmlns:clientSample="clr-namespace:ClientSample"
    xmlns:pHmiControls="clr-namespace:PHmiClient.Controls;assembly=PHmiClient"
    xmlns:trends="clr-namespace:PHmiClient.Controls.Trends;assembly=PHmiClient"
    mc:Ignorable="d"
    d:DesignHeight="300" d:DesignWidth="300"
    d:DataContext="{d:DesignInstance clientSample:PHmi,
IsDesignTimeCreatable=True}">
    <Grid>
        <StackPanel>
            <TextBlock Text="{Binding Path=IoDev.DigitalTag.Value}"/>
            <TextBlock Text="{Binding Path=IoDev.NumericTag.ValueString}"/>
            <CheckBox
                IsChecked="{Binding Path=IoDev.DigitalTag.Value, Mode=TwoWay}"
                Content="{Binding Path=IoDev.DigitalTag.Description}"/>
            <pHmiControls:NumericInput NumericTag="{Binding Path=IoDev.NumericTag}"/>
            <trends:TrendViewer Height="400">
                <trends:TrendPen
                    TrendTag="{Binding Path=Trends.NumericTagTrend}"
                    Brush="Red"/>
            </trends:TrendViewer>
        </StackPanel>
    </Grid>
</UserControl>
```



## 2.3 Setting window language

In order to display strings with regional settings (date, time etc.) it is necessary to update XamlLanguage in window constructor:

```
using System.Threading;
using System.Windows;
using System.Windows.Markup;

namespace ClientSample
{
    /// <summary>
    /// Interaction logic for MainWindow.xaml
    /// </summary>
    public partial class MainWindow : Window
    {
        public MainWindow()
        {
            Language = XamlLanguage.GetLanguage(
                Thread.CurrentThread.CurrentUICulture.IetfLanguageTag);
            InitializeComponent();
        }
    }
}
```

Or use extension method from PHmiClient.Utils namespace:

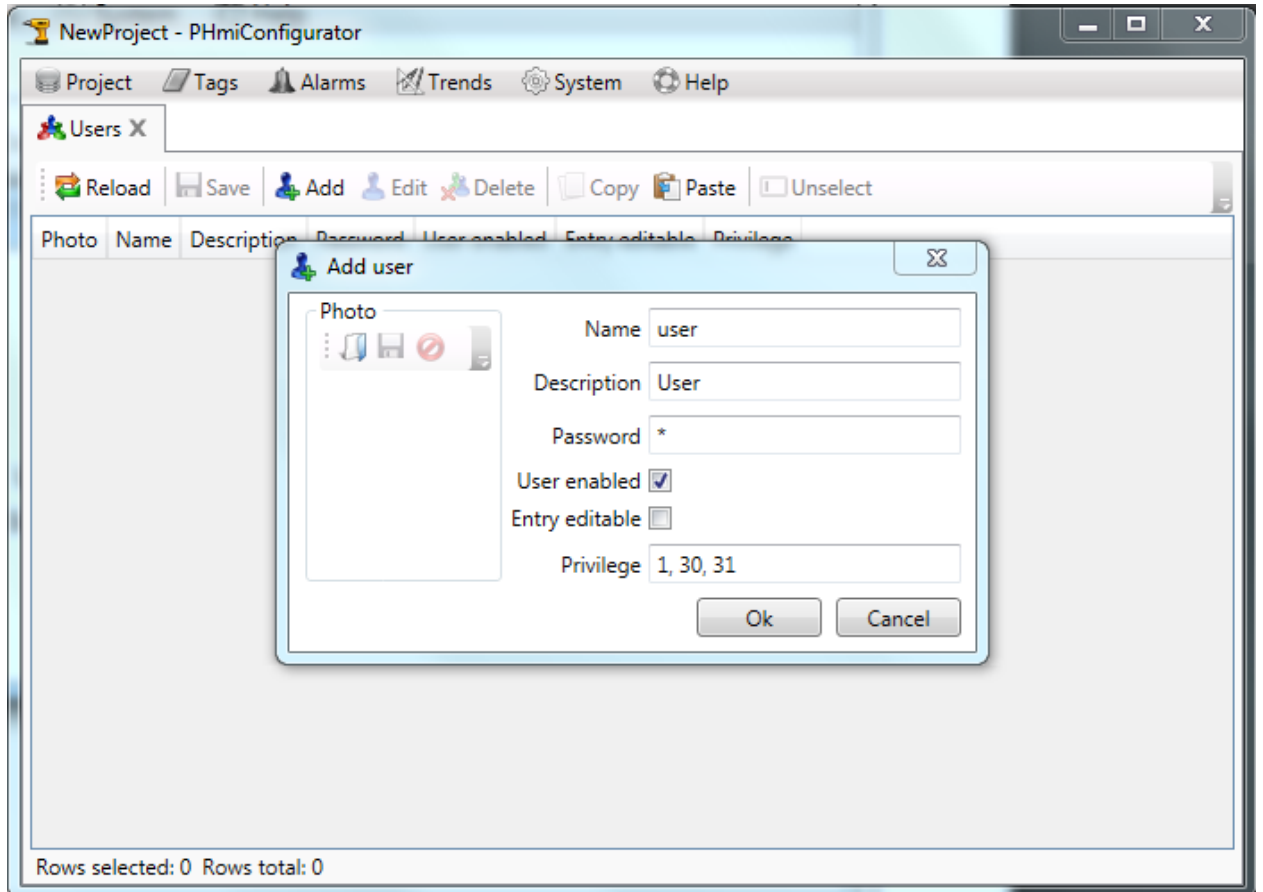
```
using System.Windows;
using PHmiClient.Utils;

namespace ClientSample
{
    /// <summary>
    /// Interaction logic for MainWindow.xaml
    /// </summary>
    public partial class MainWindow : Window
    {
        public MainWindow()
        {
            this.UpdateLanguage();
            InitializeComponent();
        }
    }
}
```

## 2.4 Users

Users are used to restrict certain actions.

Let's create a user.



Users should have name, password and privilege. Privilege is set as comma separated numbers from 1 to 32. Table of predefined privileges is below.

Privilege	Description
32	System Messages Long Description viewer
31	Users viewer
30	Users editor

Restart PHmiRunner. It will create user in the database. Note that if you later change user in the configurator it will not be changed in the runner database. It should be deleted if you want changes in the configurator to take effect (after restarting PHmiRunner). It can be done in pgAdmin application (it is installed with PostgreSQL). PHmiRunner database is named NewProject\_data.

Set to root Grid of HomePage attached property PHmiControls.Privilege in namespace PHmiClient.Controls.

```
<UserControl x:Class="ClientSample.Pages.HomePage"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
```

```

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
xmlns:clientSample="clr-namespace:ClientSample"
xmlns:pHmiControls="clr-namespace:PHmiClient.Controls;assembly=PHmiClient"
xmlns:trends="clr-namespace:PHmiClient.Controls.Trends;assembly=PHmiClient"
mc:Ignorable="d"
d:DesignHeight="600" d:DesignWidth="800"
d:DataContext="{d:DesignInstance clientSample:PHmi,
IsDesignTimeCreatable=True}">
    <Grid pHmiControls:PHmiControls.Privilege="1">
        <StackPanel>
            <TextBlock Text="{Binding Path=IoDev.DigitalTag.Value}"/>
            <TextBlock Text="{Binding Path=IoDev.NumericTag.ValueString}"/>
            <CheckBox
                IsChecked="{Binding Path=IoDev.DigitalTag.Value, Mode=TwoWay}"
                Content="{Binding Path=IoDev.DigitalTag.Description}"/>
            <pHmiControls:NumericInput NumericTag="{Binding Path=IoDev.NumericTag}"/>
            <trends:TrendViewer Height="400">
                <trends:TrendPen
                    TrendTag="{Binding Path=Trends.NumericTagTrend}"
                    Brush="Red"/>
            </trends:TrendViewer>
        </StackPanel>
    </Grid>
</UserControl>

```

If you run the project you will see that numeric input is disabled. If user enters it will be enabled.

But checkbox is enabled anyway. To get similar behavior we should bind it's `IsEnabled` property to the current user's privilege.

```

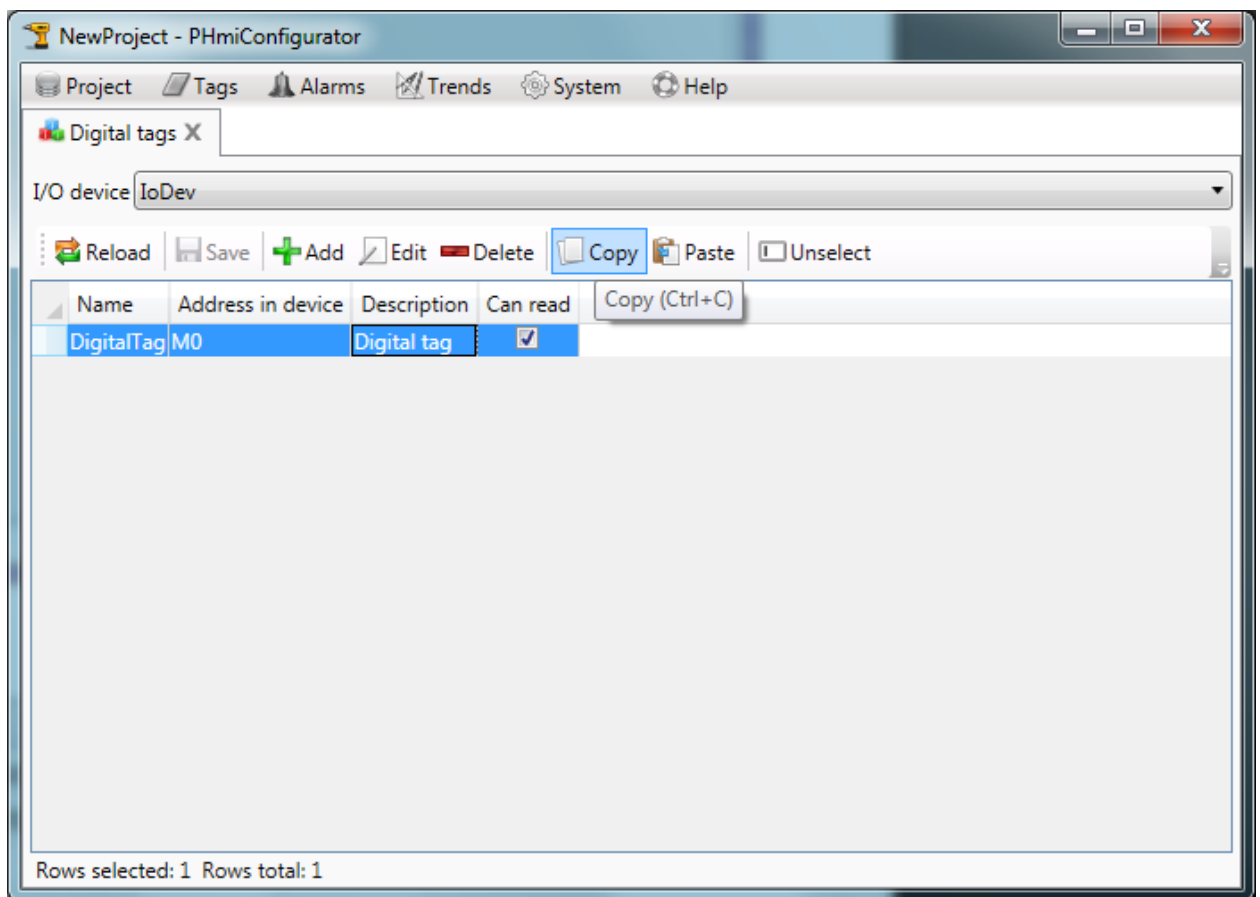
<UserControl x:Class="ClientSample.Pages.HomePage"
xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
xmlns:clientSample="clr-namespace:ClientSample"
xmlns:pHmiControls="clr-namespace:PHmiClient.Controls;assembly=PHmiClient"
xmlns:trends="clr-namespace:PHmiClient.Controls.Trends;assembly=PHmiClient"
xmlns:pHmiMultiValueConverters="clr-
namespace:PHmiClient.Converters.MultiValueConverters;assembly=PHmiClient"
mc:Ignorable="d"
d:DesignHeight="600" d:DesignWidth="800"
d:DataContext="{d:DesignInstance clientSample:PHmi,
IsDesignTimeCreatable=True}">
    <Grid pHmiControls:PHmiControls.Privilege="1">
        <StackPanel>
            <TextBlock Text="{Binding Path=IoDev.DigitalTag.Value}"/>
            <TextBlock Text="{Binding Path=IoDev.NumericTag.ValueString}"/>
            <CheckBox
                IsChecked="{Binding Path=IoDev.DigitalTag.Value, Mode=TwoWay}"
                Content="{Binding Path=IoDev.DigitalTag.Description}"
                <CheckBox.IsEnabled>
                    <MultiBinding>
                        <MultiBinding.Converter>
                            <pHmiMultiValueConverters:PrivilegedToTrueConverter/>
                        </MultiBinding.Converter>
                        <Binding Path="(pHmiControls:PHmiControls.Privilege)"
RelativeSource="{RelativeSource Self}" />
                        <Binding Path="Users.Current" />
                    </MultiBinding>
                </CheckBox.IsEnabled>
            </CheckBox>
            <pHmiControls:NumericInput NumericTag="{Binding Path=IoDev.NumericTag}"/>
            <trends:TrendViewer Height="400">

```

```
        <trends:TrendPen
            TrendTag="{Binding Path=Trends.NumericTagTrend}"
            Brush="Red"/>
    </trends:TrendViewer>
</StackPanel>
</Grid>
</UserControl>
```

### 3 Using MS Excel for tables editing

Open tags editor. Select a row and press “Copy” button.



Launch MS Excel and press “Paste”.

	A	B	C	D	E	F
1	id	Name	Address in device	Description	Can read	
2	1	DigitalTag M0		Digital tag	True	
3						

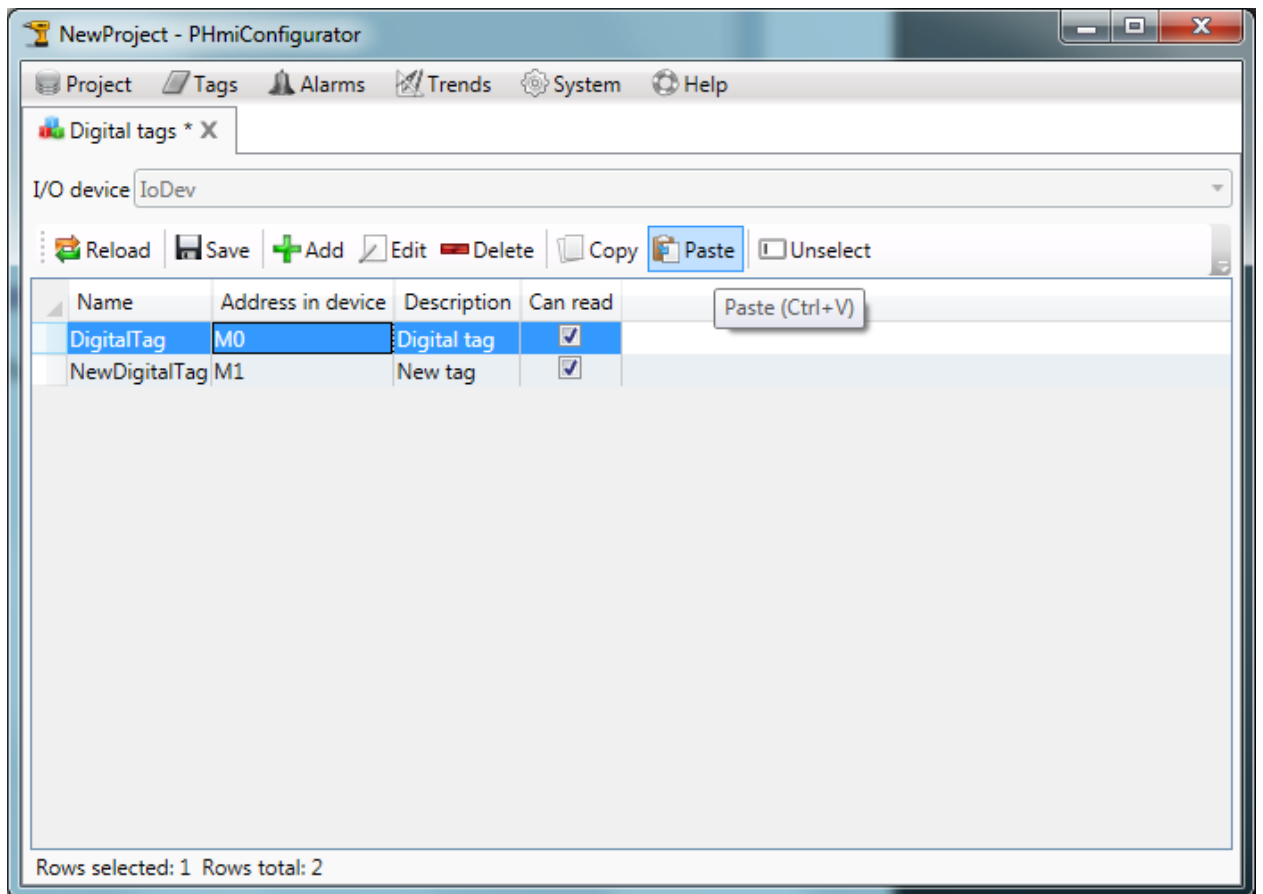
Now you can add new tags (or edit existing).

A1		fx		id	
	A	B	C	D	E
1	id	Name	Address in device	Description	Can read
2	1	DigitalTag	M0	Digital tag	True
3		NewDigitalTag	M1	New tag	True
4					

Select the rows as shown at the figure above and press “Copy”.

Then paste it to the PHmiConfigurator.





Rows where “id” is not specified will be added to the table. Others will be updated.