Web Server in iX Developer 2.0

KI00320 2013-02

1 Function and area of use

This document can be seen as a compliment to iX –Help documentation.

The Web Server adds the possibility to use a server-side Application Programming Interface (API) to interact with a running project. The API operates over the HyperText Transfer Protocol (HTTP) and can be access from any compatible device. In addition, the web server can host web pages and related media to be consumed by web browsers.

The bundled Javascript Software Development Kit (SDK) makes it easy to build interactive web pages that utilize the API.

Login settings for the web server are available from Server group on the System ribbon tab.

This document can be seen as a complement to iX –Help documentation.

Software requirements:
- iX Developer 2.0

- Text editor for editing HTML files.

Example:

Notepad – Build in text editor in Windows.

Notepad++ - Free source code editor with syntax highlighting and more.

http://notepad-plus-plus.org/

Visual Studio Express 2012 for Web – May require payment depending on circumstances.

http://www.microsoft.com/visualstudio/eng/downloads#d-express-web

Beijer Electronics Automation AB – *a company in the Beijer Electronics Group*

Parent Company (Reg. office)
Beijer Electronics Automation AB
P.O. Box 426
SE-201 24 MALMÖ, SWEDEN
Telephone +46 40 35 86 00
Fax +46 40 93 23 01
Visitor's address Stora Varvsgatan 13a, Malmö

Subsidiaries
Norway, Drammen: Beijer Electronics AS, ❩ +47 32 24 30 00
Finland, Helsinki: Beijer Electronics Oy, ❩ +358 207 46 35 40
Denmark, Copenhagen: Beijer Electronics A/S, ❩ +45 70 26 46 46
Estonia, Tallin: Beijer Electronics Eesti Oü, ❩ +372 6 518140
Latvia, Riga: Beijer Electronics SIA, ❩ +371 7 842280
Lithuania, Vilnius: Beijer Electronics UAB, ❩ +370 5 2323101

Reg no. 556701-3965 VAT no SE556701-3965-01, Internet www.beijer.se, e-mail info@beijer.se

2 About this Start-Up document

This Start Up document should not be considered as a complete manual. It is an aid to be able to start up a normal application quickly and easily. For further information we refer to the manual for iX Developer 2.0. This document and other Start Up documents can be obtained from your closest distributor of operator terminals.

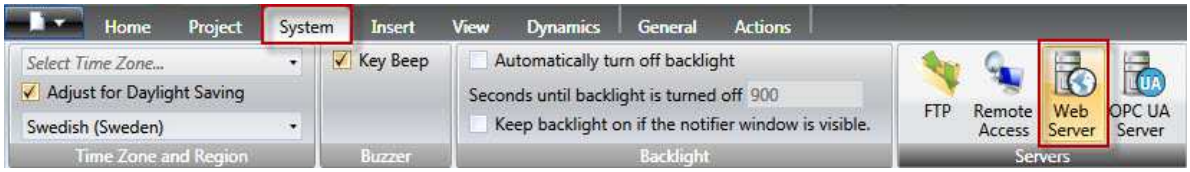
Please use the address *manual@ beijer.se* for feedback on our Start Up documents.

3 Setting up iX

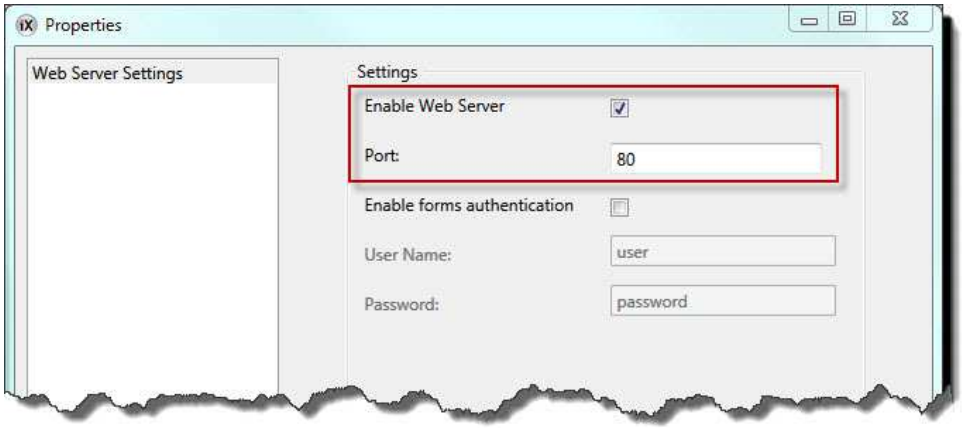
Create a Panel-project or a PC-project.
This document is based on a Panel-project but the same approach is applied to PC-project.

Let's first configure iX Developer and create a layout. In this example Demo-controller is used.

- Under the *System* ribbon click on *Web Server*.

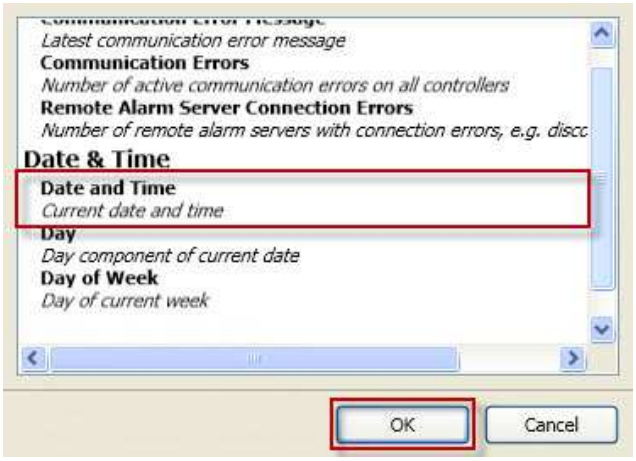


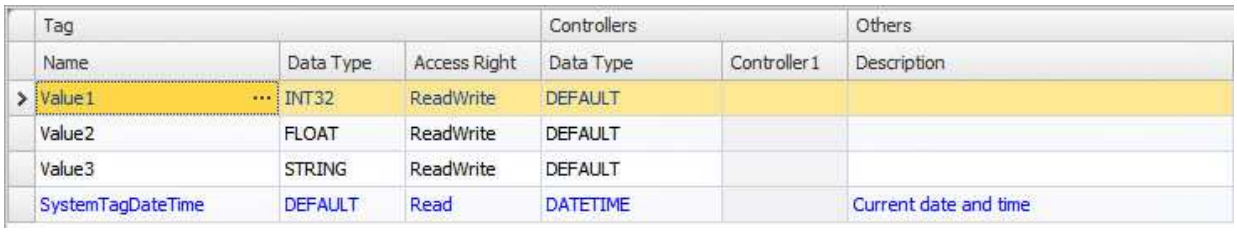
- Check *Enable Web Server* and enter a port number, default: 80.



When *Enable forms authentication* is checked; Free definable username and password can be set. When unchecked; Anonymous authentication is set.

- Go to *Tags*. Add 4 tags, one of which is a *System Tag*. These tags will be shown when the web server is accessed. The different formatting will be handled differently in the HTML-file.



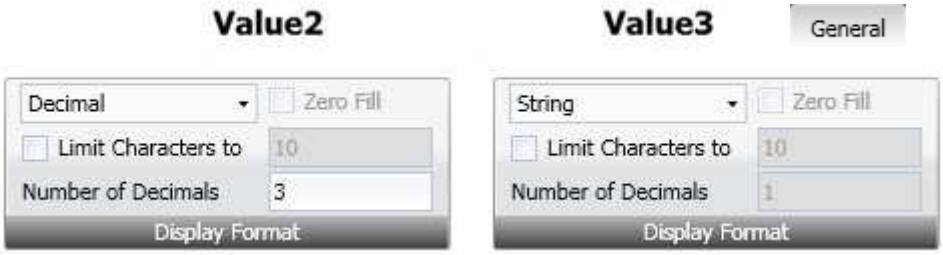


Tag			Controllers		Others
Name	Data Type	Access Right	Data Type	Controller1	Description
Value1	INT32	ReadWrite	DEFAULT		
Value2	FLOAT	ReadWrite	DEFAULT		
Value3	STRING	ReadWrite	DEFAULT		
SystemTagDateTime	DEFAULT	Read	DATETIME		Current date and time

Tag Name	Data Type
Value1	Int32
Value2	Float
Value3	String
SystemTagDateTime	DateTime

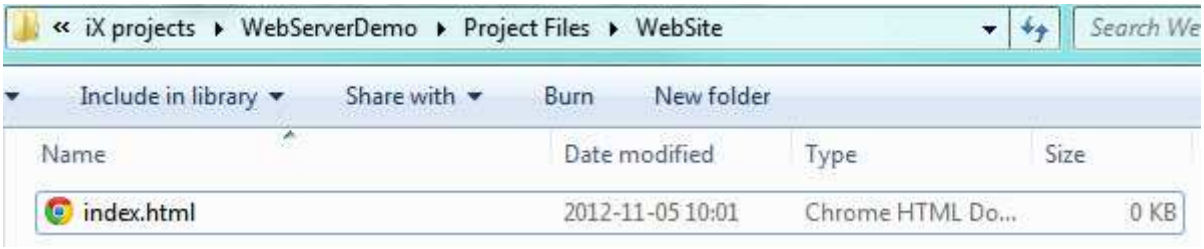
- Add 3 analogue numeric's to your screen. Connect the tags accordingly.

Note that you have to change the display format for Value2 and Value3, changed under *General* of the object. Value2 is displayed with 3 decimals and Value3 is displayed as string.



4 Set up web server interface

The web server files must be stored in the root folder: “ProjectName\Project Files\WebSite”. If *WebSite* does not exist, create it. The startpage must be called “index.html”.



- Create **index.html** in the folder **WebSite** as mentioned.
- Open the html file with your html-editor. In this document *Visual Studio Express for Web* is used.

There are two ways of working with the source files with iX panel.

One way is to work online and replace/modify the files in the root folder of the panel. This is done by enabling FTP server on the panel and using an FTP client to access the files/folders. The changes take effect immediately and you don’t need to restart the application.

Please see startup document article ID: DSSE-8ZBGH6 on how to set up FTP server.

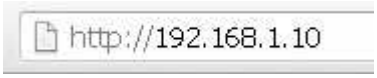
- Open up “index.html” with your HTML-editor.
- Create a simple code which will display “Hello World” to check that the setup is correct.

Example, *index.html* located in *WebSite*:

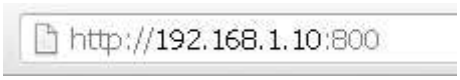
```
<!DOCTYPE html>
<html>
<head>
  <title>Webserver demonstration</title>
</head>

<body>
<p>Hello World</p>
</body>
</html>
```

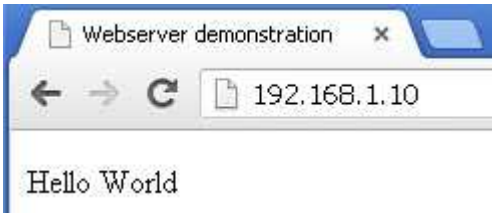
- Save the file and open up, or rebuild, your project.
- Download.

To access the web server you simply type in the IP address into your web browser.
If authentication is enabled, a login screen will first appear.



If you are not running on the default port 80 you have to enter this manually.
The port no. is separated with colon E.g. “192.168.1.10:800”.





Now we know that the web server is correctly set. Further, we will connect iX-tags to the web server.

5 Javascript SDK Overview

The JavaScript SDK provides a rich set of client-side functionality for accessing the server-side API calls. This enables the possibility to build highly interactive web pages that uses tags defined in the project.

5.1 Connect iX SDK to HTML document

1. The first step to build an interactive web page is to reference the SDK by inserting the following script into to the header of the html.

```
<script src="/assets/iX.js" type="text/javascript"></script>
```

2. Next, the SDK needs to be initialized. In its simplest form the following lines need to be added in the header after the reference to the SDK:

```
<script type="text/javascript"> iX.setup({});</script>
```

This will make it possible to use the defined tag metadata attributes to incorporate the project tags into the web page. The following line will, for example, insert the value of Tag1 into the html document:

```
<span data-ix-tag="Value1"</span>
```

Example of neccecary header tags:

```
<!DOCTYPE html>
<html>
<head>
        <title>Webserver demonstration</title>
        <script src="/assets/iX.js" type="text/javascript"></script>
        <script type="text/javascript"> iX.setup({});</script>
</head>

<body>
       <p>Hello World</p>
</body>
</html>
```

The header lines in the example above needs to be included in each HTML document that requires iX SDK functionality.

5.2 Tag Value or Properties

To insert the current value of a tag or tag property into an html element, the **data-ix-tag** attribute can be used. The value of the attribute should be the name of the tag as defined in the project and the property of a tag. Separate the names with a dot ".".

If no property is provided, the value of the tag is used.

```
<span data-ix-tag="Tag1">Replaced with Tag1 value</span>
<span data-ix-tag="Tag1.DataType">Replaced with the datatype of Tag1</span>
```

5.3 Formatting

Formatting of a tag value can be applied to a tag of numeric type or a datetime. The format is defined using the **data-ix-format** attribute.

```
<span data-ix-tag="SystemTagDateTime" data-ix-format="Y-M-d" </span>
<span data-ix-tag="NumericTag" data-ix-format="0.000" </span>
```

In the first example the Date will be shown as Year-Month-Day, where Year is a four digit number, month as the abbreviated name of the month and the day of the month as 2 digit number from 01 to 31. eg. 2012-Jan-01.

In the latter example the *NumericTag* will be shown with 3 decimals.

In iX Developer 2.0 help documentation (F1) you can find complete table of custom date and time format as well as custom numeric formats.

5.4 Refresh Modes

To control how tag values are refreshed, the **data-ix-refresh** attribute can be used with elements defined with **data-ix-tag** .

It is recommended to use a interval of 5 seconds.

```
<span data-ix-tag="Tag1" data-ix-refresh="onetime"></span>
<span data-ix-tag="Tag2" data-ix-refresh="none"></span>
<span data-ix-tag="Tag3" data-ix-refresh="interval"></span>
```

onetime = Refreshes only once. E.g. screen opened or refreshed.

none = Only refreshes when tag is invoked by refresh action.

interval = Refreshes at the interval set in iX.setup (default interval of 5s. This can be changed, see further down in this document)

Optional to refreshing the tags individually, global settings can be made when referencing iX SDK in the header of the HTML document.

Read more about refreshing tags in chapter 8.

5.5 Setting tag value

To set the value of tag, two elements must be defined.

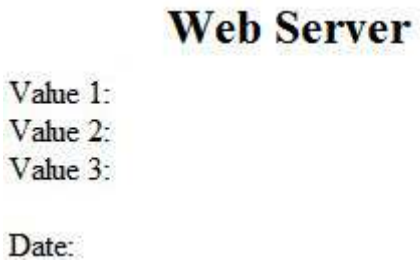
The first element has the **data-ix-setter** attribute on the position where the tag value can be entered. The second element has the **data-ix-submitbutton** attribute used to submit the new tag value.

The **data-ix-submitbutton** attribute takes the tag names, separated by comma “,”, to be submitted when the button is clicked. To submit all tags an asterisk “*” can be used.

```
<input type="number" data-ix-setter="Tag1" />
<input type="number" data-ix-setter="Tag2" />
<input type="number" data-ix-setter="Tag3" />
<input type="button" data-ix-submitbutton="Tag1" />      <!--Sets Tag1-->
<input type="button" data-ix-submitbutton="Tag1, Tag2" /><!--Sets Tag1 and Tag2-->
<input type="button" data-ix-submitbutton="*" />         <!--Sets Tag1,Tag2,Tag3-->
```

6 Complete example project

In this document we will first create a simple layout which will display the four tag values we created earlier.



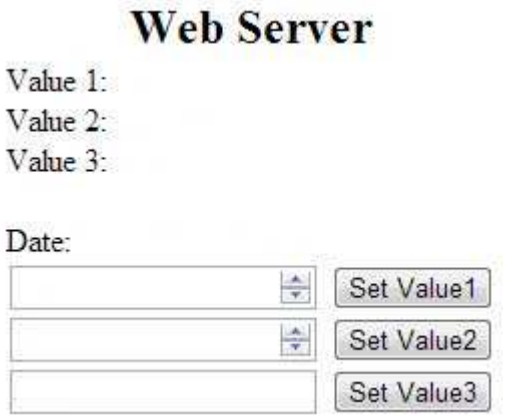
This is the code which is used for connecting the tags and to refresh the tags at an interval of 5 seconds (default). Value 2 and Date also has to be formatted.

```
<!DOCTYPE html>
<html>
<head>
  <title>Webserver demonstration</title>
  <script src="/assets/iX.js" type="text/javascript"></script>
  <script type="text/javascript"> iX.setup({});</script>
</head>

<body>
<span style="text-align:center";><h1>Web Server</h1></span>

Value 1: <span data-ix-tag="Value1"" data-ix-refresh="interval"></span><br />

Value 2: <span data-ix-tag="Value2" data-ix-format="0.000" data-ix-
refresh="interval"></span><br />

Value 3: <span data-ix-tag="Value3" data-ix-refresh="interval"></span><br /><br />

Date: <span data-ix-tag="SystemTagDateTime" data-ix-format="Y-m-d G:i" data-ix-
refresh="interval"></span><br>

</body>
</html>
```

You will now be able to read the tag value from the application.

- Save the html file/s.
- Download and test.

Let's add input and set-button to Value1-2-3. Something like this:



Add the following in the body under date:

```
<!DOCTYPE html>
<html>
<head>
  <title>Webserver demonstration</title>
  <script src="/assets/iX.js" type="text/javascript"></script>
  <script type="text/javascript"> iX.setup({});</script>
</head>
<body>
<span style="text-align:center";><h1>Web Server</h1></span>
Value 1: <span data-ix-tag="Value1"" data-ix-refresh="interval"></span><br />
Value 2: <span data-ix-tag="Value2" data-ix-format="0.000" data-ix-
refresh="interval"></span><br />
Value 3: <span data-ix-tag="Value3" data-ix-refresh="interval"></span><br />
<br />
Date: <span data-ix-tag="SystemTagDateTime" data-ix-format="Y-m-d G:i" data-ix-
refresh="interval"></span><br>
<!—Add following:-->
<input type="number" data-ix-setter="Value1" />
<input type="button" value="Set Value1" data-ix-submitbutton="Value1" style="width: auto" />
<br />

<input type="number" data-format="0,000" data-ix-setter="Value2" />
<input type="button" value="Set Value2" data-ix-submitbutton="Value2" style="width:auto" />
<br />

<input type="text" data-ix-setter="Value3" />
<input type="button" value="Set Value3" data-ix-submitbutton="Value3" style="width: auto" />

</body>
</html>
```

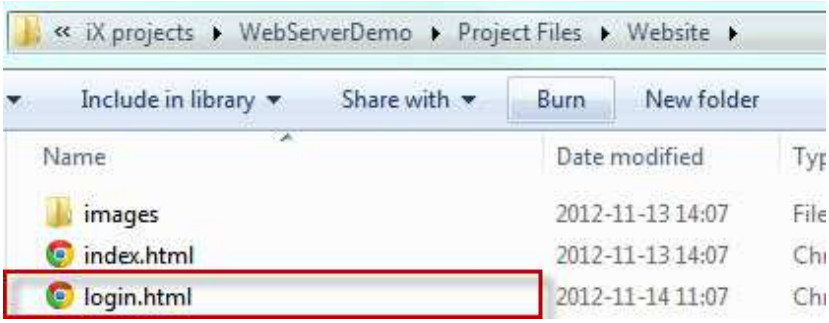
You should now be able to set values 1-3.

7 Change login

If *forms authentication* is enabled the default login screen will first appear and provide a login authentication using any code.

Security notice! Username and Password are sent in plain text. It's recommended to use encrypted tunnelling (VPN) if considering to use web browser connecting over unsafe networks.

It's possible to override the default login page by creating your own "login.html" and place it in the *Website* folder. Authentication must be enabled.



The Javascript SDKprovides tools for creating the form using a few conventions. Note that this only applies if you have authentication enabled in the web server settings.

Reference to the authentication script must be made in *head*:

```
<script src='/assets/authentication.js' type='text/javascript'></script>
```

Assign id:s to *Username* and *Password* field with the predefined id's "**ix-username**" and "**ix-password**". In *body*:

```
Username:<input id="ix-username" type="text" maxlength="20" required="true" />
Password:<input id="ix-password" type="password" maxlength="20" required="true" />
```

Assign id to login button with the predefined id "**ix-login**". Any element can be used. In *body*:

```
<button id="ix-login">LOGIN</button>
```

Optionally add an element with the predefined id "**ix-invalid-credentials**" that will be shown if the login fails. Style is located in the header.

In html *head*:

```
<style>
     #ix-invalid-credentials
     {
         display:none;
     }
</style>
```

In html *body*:

```
<span id="ix-invalid-credentials">Failed to login, try again.</span>
```

Simple login example:

```
<!DOCTYPE html>
<html>
<head>
 <title>Web Server Login</title>
     <script src='/assets/authentication.js' type='text/javascript'></script>
      <style>
            #ix-invalid-credentials
            {
                display:none;
            }
    </style>
</head>
<body>
<h1>Login</h1>
Username: <input id="ix-username" type="text" maxlength="20" required="true" />
<br />
Password: <input id="ix-password" type="password" maxlength="20" required="true" />
<br />
<button id="ix-login">Login</button> <span id="ix-invalid-credentials">Login failed, try
again.</span>

</body>
</html>
```

8 Appendix: iX-Object configuration

After loading the JavaScript SDK, call iX.setup to initialize the framework. This will wire the tag metadata attributes that you have defined in html and will start periodical polling.

iX.setup has a set of parameters that can be modified.

Options:

Property	Type	Description	Argument	Default
refreshInterval	Int	The refresh interval of tag values in ms. For performance reasons it is recommended not to use low values for this setting.	Optional	5000
refreshMode	String	Overrides the default refresh mode of tag values. Accepted modes are: none, onetime, interval.	Optional	onetime
refreshModeSetter	String	Overrides the default refresh mode of tag setters. Accepted modes are: none, onetime, interval.	Optional	onetime

This example below will set periodical refresh interval to 10 seconds, set the refresh mode of html elements marked with the **data-ix-tag** attribute to interval and set the refresh mode of html elements marked with the **data-ix-setter** attribute to refresh only once. Note upper and lower case!

```
<!DOCTYPE html>
<html>
<head>
  <title>Webserver demonstration</title>
  <script src="/assets/iX.js" type="text/javascript"></script>
<script type="text/javascript"> iX.setup({ refreshInterval: 10000, refreshMode: 'interval',
refreshModeSetter: 'onetime'});</script>
</head>

<body>

<input type="number" data-ix-tag="Tag1" /> <!--Tag will be updated every 10 seconds-->
<input type="number" data-ix-setter="Tag2" /> <!--Tag will be updated once-->
<input type="number" data-ix-setter="Tag3" data-ix-refresh="interval" /> <!--Tag will be
updated every 10 seconds-->
<input type="number" data-ix-tag="Tag4" data-ix-refresh="onetime" /> <!--Tag will be updated
once-->

</body>
</html>
```

8.1 iX invalid style

Basic validation is done on input values. If the value cannot be assigned to the Tag then the value will not be set. To indicate an invalid value the framework will set the CSS class “ix-invalid” on the element. This can then be used to provide the user with visual cues of the invalid state. For example, the following CSS snippet adds a red border around the **input** when the **input** is incorrect:

Located in head:

```
<style type="text/css">
      .ix-invalid {
          border: 1px solid red;
                  }
</style>
</head>
```

Locaded in body:

```
<body>
<input type="number" data-ix-tag="Tag1" class="ix-invalid" />
</body>
```

8.2 iX refreshElements

Refresh tag values manually. Values can be manually updated on event E.g. button.

In the example below, the html elements with id "Value1Id" to “Value3Id” will be updated when the element with id "refreshButton" is clicked. This example uses jQuery (bundled with iX.js) to bind to the click event of the "refreshButton".

In html *head*:

```
<script type="text/javascript">iX.setup({});
       $(document).ready(function() {
          $('#refreshButton').click(function() { iX.refreshElements(['Value1Id', 'Value2Id',
'Value3Id']); });});
</script>
```

In html *body*:

```
Value 1: <span id="Value1Id" data-ix-tag="Value1" ></span><br />
Value 2: <span id="Value2Id" data-ix-tag="Value2" data-ix-format="0.000" ></span><br />
Value 3: <span id="Value3Id" data-ix-tag="Value3" ></span><br />
<button id="refreshButton">Refresh</button>
```

9 Complete example html code

```
<!DOCTYPE html>
<html>
<head>
  <title>Webserver demonstration</title>
  <script src="/assets/iX.js" type="text/javascript"></script>
  <script type="text/javascript">iX.setup({ refreshInterval: 10000, refreshMode: 'interval',
refreshModeSetter: 'onetime' });
      $(document).ready(function() {
          $('#refreshButton').click(function() { iX.refreshElements(['Value1Id', 'Value2Id',
'Value3Id']); });
      });
  </script>

  <style type="text/css">
      .ix-invalid {
          border: 1px solid red;
                  }
  </style>
</head>
<body>

<span style="text-align:center";><h1>Web Server</h1></span>
Value 1: <span id="Value1Id" data-ix-tag="Value1" ></span><br />
Value 2: <span id="Value2Id" data-ix-tag="Value2" data-ix-format="0.000" ></span><br />
Value 3: <span id="Value3Id" data-ix-tag="Value3" ></span><br />
<br />
Date: <span data-ix-tag="SystemTagDateTime" data-ix-format="Y-m-d G:i" ></span><br>
<input type="number" data-ix-setter="Value1" />
<input type="button" value="Set Value1" data-ix-submitbutton="Value1" style="width: auto" />
<br />
<input type="number" data-format="0,000" data-ix-setter="Value2" />
<input type="button" value="Set Value2" data-ix-submitbutton="Value2" style="width:auto" />
<br />
<input type="text" data-ix-setter="Value3" />
<input type="button" value="Set Value3" data-ix-submitbutton="Value3" style="width: auto" />
<br />
<button id="refreshButton">Refresh</button>

</body>
</html>
```