PTC®

PTC® IoT Academic Program

Simple Weather Mashup

How to build a Simple Weather Mashup

Revision #	Date	ThingWorx Revision	Changes	Owner
1.0	03.11.2014	5.0.2.33		Adrian Petrescu
2.0	22.12.2014	5.0.2.33		Adrian Petrescu
3.0 – current	05.01.2015	5.0.2.33		Adrian Petrescu

SIMLPE WEATHER MASHUP 1

22.12.2014

Overview 2

Simple Weather Mashup

Overview

This document shows you how to build a simple application also known as a mashup, using the ThingWorx Composer development tool, and then how to access your mashup using any common browser or tablet.

This document also introduces you to some important ThingWorx Composer concepts such as mashups, tags, things, properties, widgets, events and services.

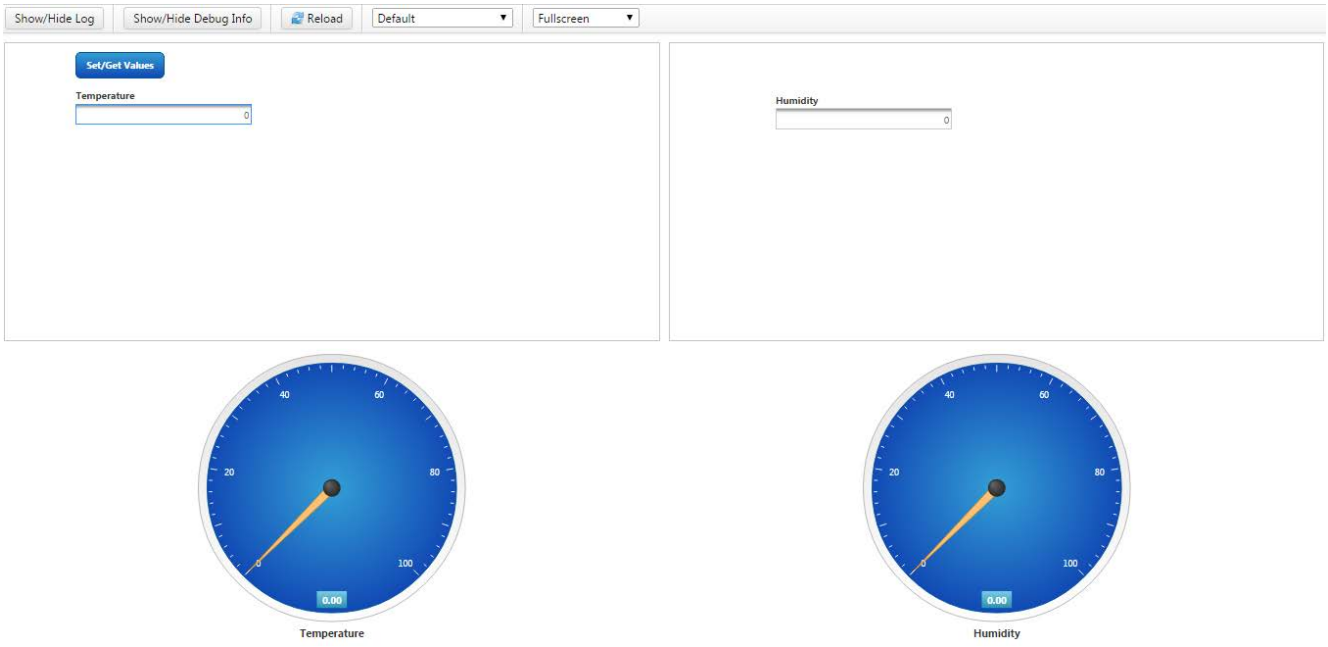
To be more precise, you will use the ThingWorx Composer to create a mashup also known as an application. You will begin by creating a "Model Tag" called "TempAndHumidity". Model Tags are used to identify all your mashup components.

The next step will be to create a "Thing" called TemperatureAndHumidity. The "Thing" will have two **properties,** one called Temperature the other called Humidity.

Next, you will create the actual mashup. The mashup will have dials to display your data and data entry boxes so you can enter your data. These are called widgets. The mashup will also have two services added to it. One service will has the purpose to set a value to the Temperature and Humidity properties, and the other has the purpose to retrieve the temperature and humidity property values.

Inside your mashup you will learn to bind the data between the widgets (dials and data entry boxes) and the services in order to set values on your Thing's temperature and humidity properties (using a button widget) and to get the values from your Thing's properties and display them on the dials.

The end result will be a mashup similar to this one that displays the temperature and humidity values that you enter:



The mashup will work in two steps:

1) You can set and view values for the Temperature and Humidity properties from your mashup. In order to achieve this you will need to use the SetProperties service from your

Thing. This service will be explained in future steps, but like the name points out, its purpose is to set the values inside the Things property or properties. In your mashup you will enter the temperature and humidity values inside the two numeric entry widgets and then click on the "Set/Get Values" button. The button signals an **event** that sets the values and triggers a **service** that gets the temperature property value and displays it on the Temperature gauge **widget** and the humidity property value and displays it on the Humidity gauge **widget**;

2) You can then view the values set in your Temperature and Humidity Thing's properties from your mashup. In order to achieve this you will use the GetProperties service from your Thing. This service will be explained in future steps, but like the name points out, its purpose is to get values from the Thing's property or properties. All you need to do is just refresh your mashup (use the "Reload" button from the mashup) and the values from the Temperature and Humidity gauge widgets will automatically change according to what you set in your Thing's properties.

This mashup does not require any programming skills. The following five steps outline how this simple mashup is built using the ThingWorx Composer:

1. **Create a Model Tag to identify all your mashup components;**

2. **Create a Simple TemperatureAndHumidity Thing;**

3. **Create Properties for the TemperatureAndHumidity Thing;**

4. **Create the TemperatureAndHumidityMashup Mashup;**

5. **Add Services and Bind the Data Inside the TemperatureAndHumidityMashup Mashup;**

In future mashups, you will learn how data, such as temperature and humidity, can come from a web service (such as Yahoo Weather), predefined simulated data, or from a temperature and humidity sensor connected to a microprocessor (such as a Raspberry Pi).

Step 1: Create a Model Tag to identify all your mashup components:

*ThingWorx **model tags** are a mechanism to label ThingWorx objects and data to assist in grouping, filtering, and locating ThingWorx objects and searching/finding data efficiently. Model tags provide metadata about an object/entity in ThingWorx or a row/record recorded as data inside of a ThingWorx object/entity.*

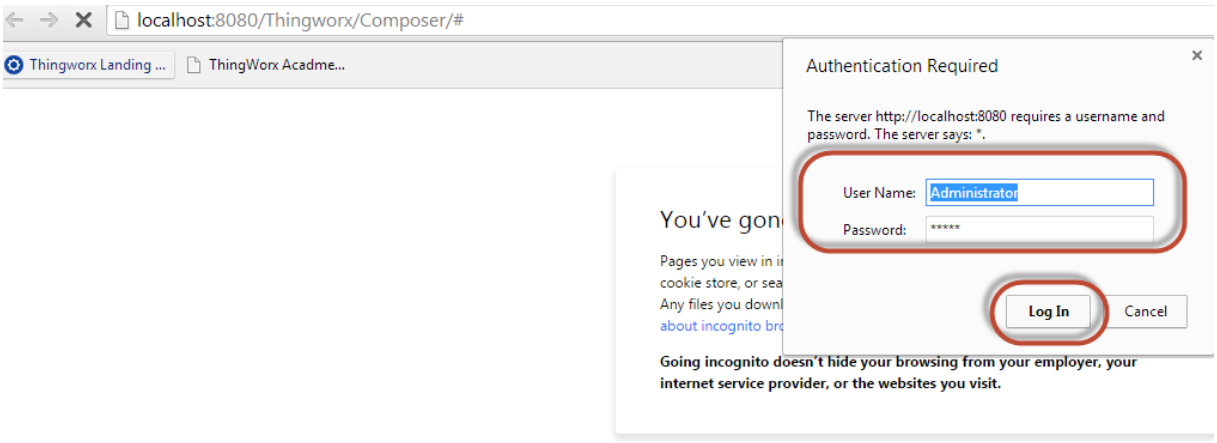
A tag is a combination of a vocabulary and a specific vocabulary term. A tag is shown as Vocabulary: VocabularyTerm. Almost every ThingWorx entity can be tagged. Tags can be used to create a relationship between many different ThingWorx entities.
A vocabulary is a list of words or identifiers used to describe a particular concept or thing. A vocabulary contains multiple vocabulary terms. An example of a vocabulary would be Colors or Locations.
A vocabulary term is a word or phrase used to describe a thing or to express a concept. Some examples of vocabulary terms of the Colors vocabulary are red, blue, and green. For Locations, it might be city names or plants.

ThingWorx has a naming restriction, in that when you set a name to a component in ThingWorx, you can't have a space inside the name box. This means that all of your components names must be one word (for example when you want to create the temp and humidity tag, you enter "tempandhumidity" as one word rather than, "temp and humidity"). Additionally, there can't be two components with the same name in the same ThingWorx instance. This applies to all objects in ThingWorx, not just model tags.

A) Log in to the ThingWorx Composer environment

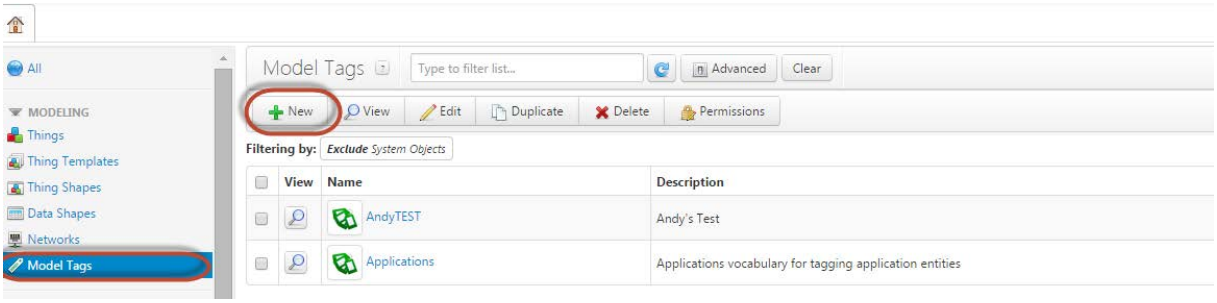
*The **Composer** is an integrated development environment (IDE) for the creation of ThingWorx applications. Both the data modeling and user interface development aspects of application development are performed using the Composer*

- *Log in to the Composer application with the URL and the credentials you were supplied with*



B) Create a Model Tag

- *Select the **Model Tags** option from the **Modeling** section in the **Home** tab and click the **New** button*



- *Enter the Name and Description (TempAndHumidity) for the new model tag in the **General Information** section and Save it (don't forget about the name restrictions)*

- *Press the **Edit** button to get back to editing mode*

- *Select the **Manage Terms** option. Just like a model tag called Color with terms such as red, blue and green, in this step we will create a term called "simple" for our TempAndHumidity model tag. Future terms will be named like complex or verycomplex.*

- *Add a new Term called simple by typing "simple" in the box and press **Add Term**. After this you are able to see the new term and then you can press **Done***

- *After this press "**Save**" again in order to save the model tag with the new term*

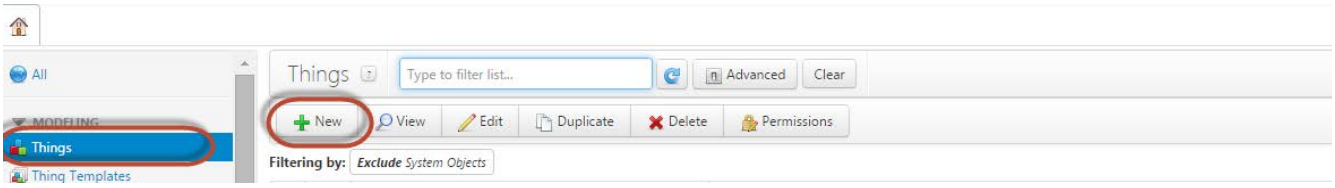
In Step 1, you've created a new Model Tag that you will use in order to tag all your mashup components. The main reason you are doing this is to always be able to identify a mashup's components and if you ever want to export a mashup from an instance of ThingWorx to another, you can do this with the export feature. The export feature works with Tags, so basically it will export everything from the ThingWorx platform that has a certain Tag.

Step 2: Create a simple TemperatureAndHumidity Thing:

The ThingWorx application development environment operates on an object or thing-driven model, which means that you represent your process in terms of things that have properties and business logic.
In addition to configuring things to represent your physical assets, you can also use things for a variety of other tasks. To facilitate this, all things are based on thing templates.
When you create a thing, you can base it off of your custom template, one of the built-in thing templates, or the generic template. Creating a thing using the generic template creates a thing with no inherited properties.

A) Create a TemperatureAndHumidity thing

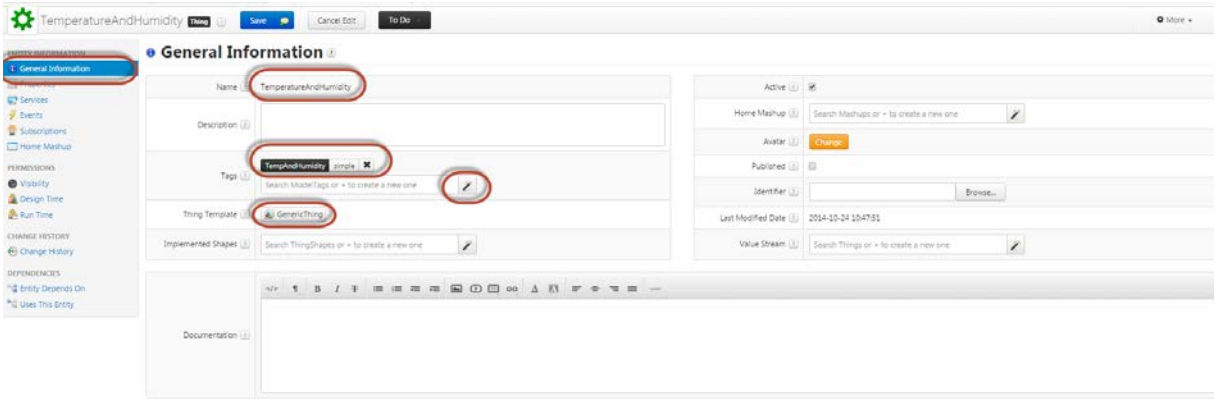
- *Select the **Things** option from the **Modeling** section in the **Home** tab and click the **New** button*



- *Name the new **Thing** "TemperatureAndHumidity" in the **General Information** section.*
- *Select the tag created in step 1: **TempAndHumidity simple** from Tags.*
- *Select **GenericThing** as the Base Thing Template (every Thing in ThingWorx must have assigned to it a Thing Template. If we don't need to create a new Thing Template with specific services, we will use the default, GenericThing template. This default template has all the necessary services that a Thing needs to work. A Thing gets all his characteristics from a Thing Template, so in our case we will have the minimal characteristics needed for a Thing to work.)*

*The ThingWorx Composer provides an Auto-Complete feature in its search forms. By typing "g" into the **Thing Template** text box it will filter the list to all templates that begin with "g" and make it easier to find the **GenericThing** Template. All the ThingWorx Composer search wizards make it easy to find selections or entities in this manner.*

- *The screen appears as shown below:*



- *Click the **Save** button to save the thing*

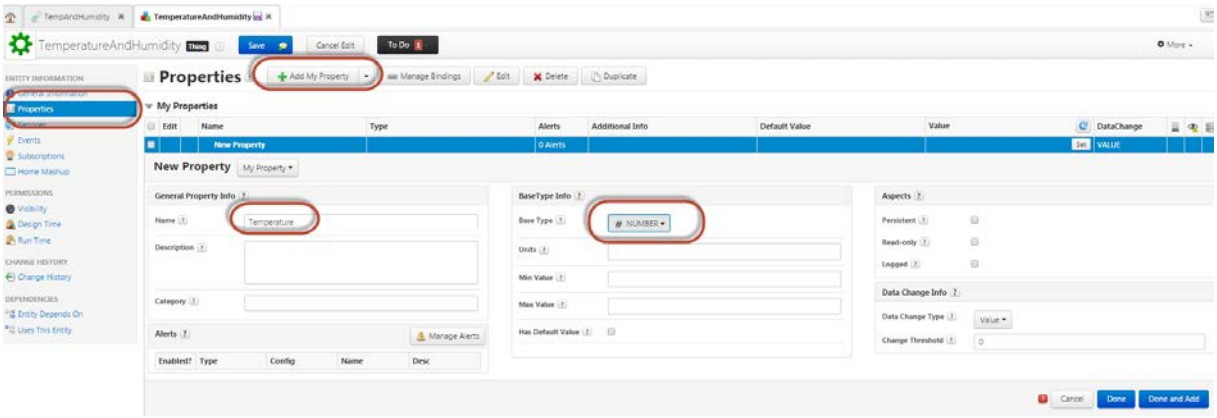
In Step 2, we've created a Thing that will help us log and display the information in and from our mashup. We will also use this Thing in order to do the bindings between the widgets from ThingWorx and the data from this Thing's services (GetProperties and SetProperties).

Step 3: Create properties for the TemperatureAndHumidity Thing:

You can define a property's description and category, which displays as a grouping when you use the Property/Service/Event browser. A property can originate from several base types, including the typical standard types and many Thingworx-specific types. Any property can have a default value and the property initializes with that value when it is instantiated in a thing.

A) Create a property called Temperature

- *Select **Properties** under the Entity Information section from the **TemperatureAndHumidity** Thing;*
- *Click the **Edit** button;*
- *Click the **Add My Property** button;*
- *Complete the fields as follow: Name: **Temperature**, Base Type: **Number;***
- *The screen appears as shown below:*



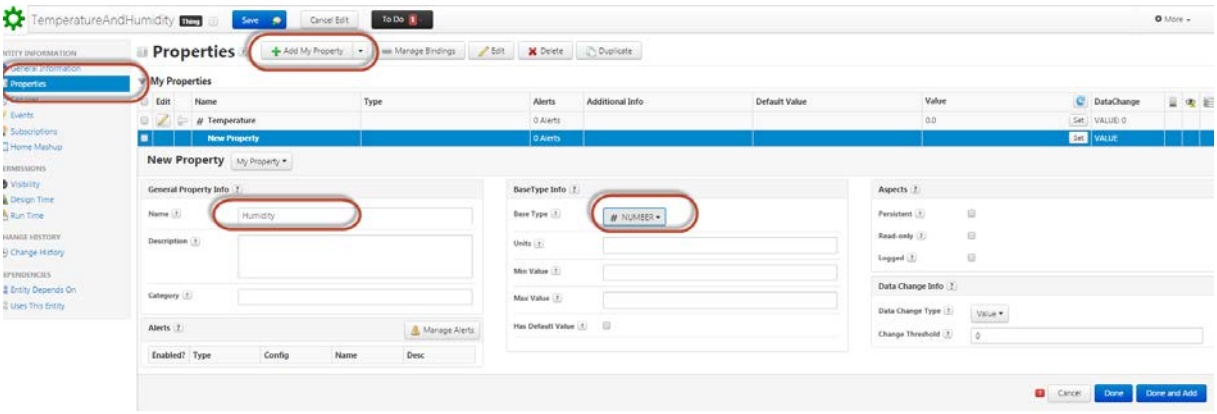
- *Click the **Done** button;*
- *Click the **Save** button in order to save your Thing with the newly added property;*

B) Create a property called Humidity

- *Select **Properties** under the Entity Information section from the **TemperatureAndHumidity** Thing;*
- *Click the **Add My Property** button;*

- *Complete the fields as follow: Name:* ***Humidity****, Base Type:* ***Number;***

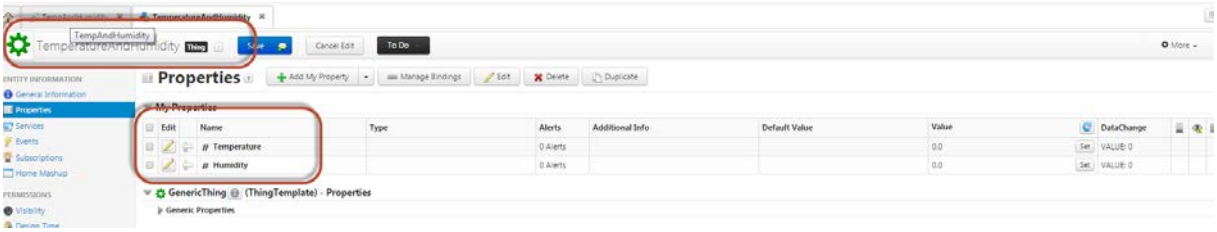
- *The screen appears as shown below:*



- *Click the* ***Done*** *button;*

- *Click the* ***Save*** *button in order to save your Thing with the newly added property;*

- *The screen appears as shown below:*



In Step 3, we've created 2 properties (Temperature and Humidity) for our Thing. We will use these properties in order to store and set values.

Step 4: Create the TemperatureAndHumidityMashup Mashup:

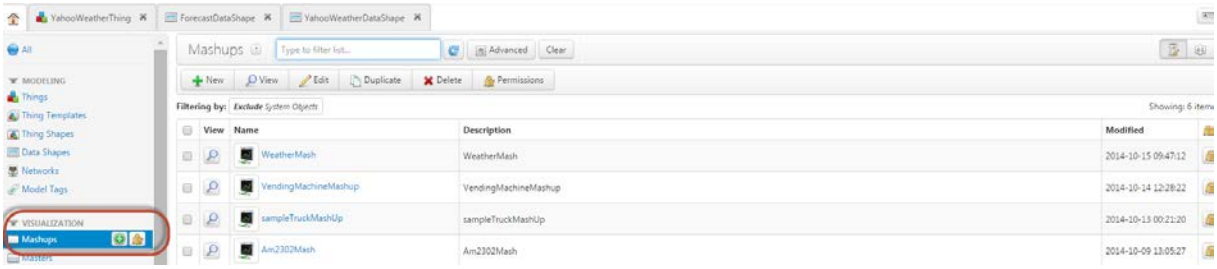
A) Create a TemperatureAndHumidityMashup Mashup:

Now that our TemperatureAndHumidity thing has been created, we can build a simple mashup to display the information stored in his properties.

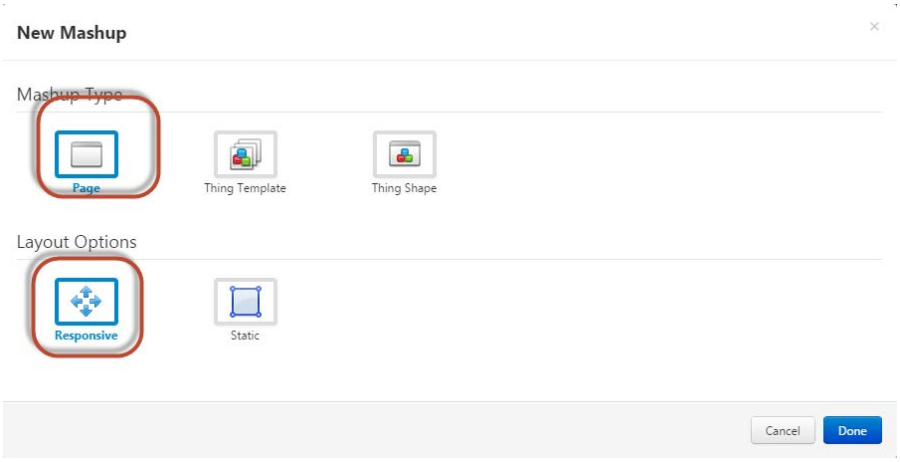
Mashup is the term used for a web page that allows you to visualize and interact with the model you have defined. Mashups usually integrate information from multiple sources bringing information together, potentially exposing relationships or giving new meaning to your model data.

*First navigate to the **Home** tab*

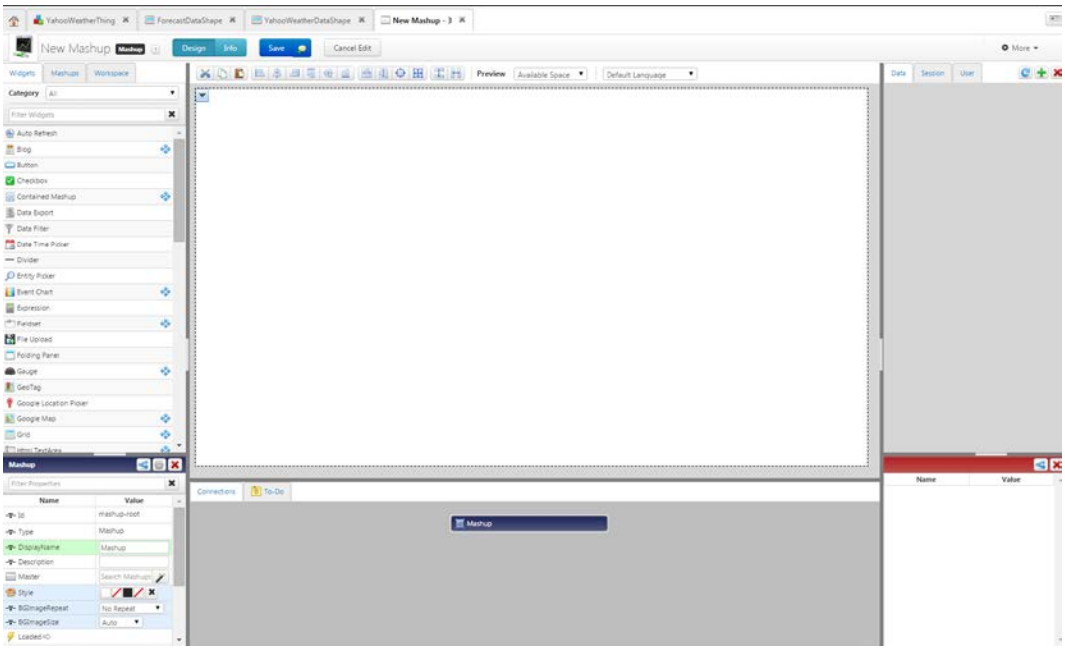
- *On the left panel, hover over the **Mashups** entry in the **Visualization** section and click the green **+** button as shown below:*



- *Leave the default **Mashup Type** of Page*
- *Select Responsive as the **Layout Option**.*



- *A mashup can be **Responsive** or **Static**. A responsive mashup means that it will always fill to the resolution of the display, never leaving any unused space around the mashup. A static mashup is statically sized to the dimensions that you define. When displayed in a lower resolution, it will have scroll bars, in a higher resolution it will leave unused space around the mashup;*
- *After clicking **Done** you will have a blank canvas on which to build your Mashup, as shown below:*



In this "Design" window we can see the following information:

- *On the left side we have the widgets side of the mashup. There are 3 tab's(Widgets, Mashups, Workspace) in which we can see all the available widgets, mashup's and every widget and their properties added in our workspace;*

- *On the middle we have our work area (workspace) where we drag and drop our widgets + services to bind the widgets.*

- *On the bottom we have the 2 tabs(Connections and To-Do) that helps you see if you did the correct bindings and if you forgot to bind something from the workspace;*

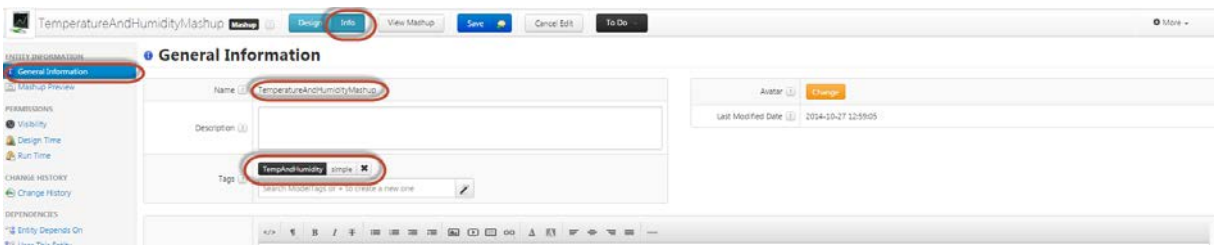
- *On the right side we have Services part of the mashup;*

- *Press the **Info** button, next to **Design;***

- *Name the new **Mashup** "TemperatureAndHumidityMashup" in the **General Information** section;*

- *Select the tag created in step 1: **TempAndHumidity simple** from Tags and save it;*

- *The screen appears as shown below:*



B) Add widgets to the TemperatureAndHumidityMashup Mashup:

Now that the TemperatureAndHumidityMashup has been created, you can add widgets to set and display the temperature and humidity values. For this, you need to add a few layout widgets, two panels, a button widget, two numeric entry widgets and two gauge widgets.

*A **Widget** is a visual element you can place in a **Mashup** that understands how to display information formatted in a useful way. Table **Widgets** know how to format the rows of data in your model as an HTML table. Map **Widgets** know how to display your position on a map from your latitude and longitude. **Mashups** are composed of multiple widgets that work together to display information.*

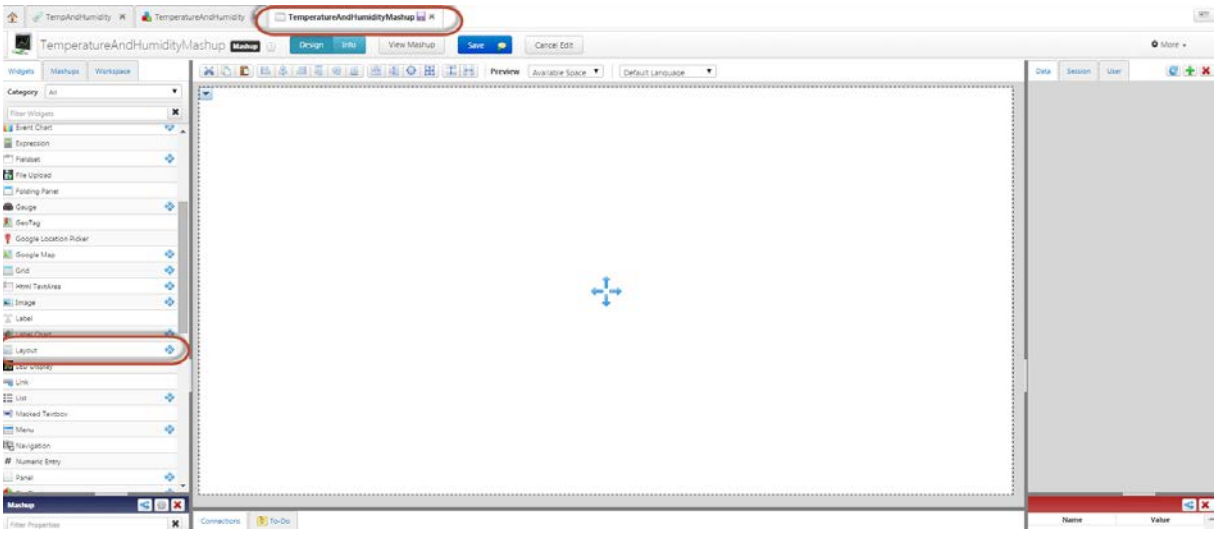
- *Access the newly created mashup by pressing the Home tab, on the left panel, go over the **Mashups** entry in the **Visualization** section and click the **TemperatureAndHumidityMashup** as shown below:*

- *Now, you add the layout widgets.*

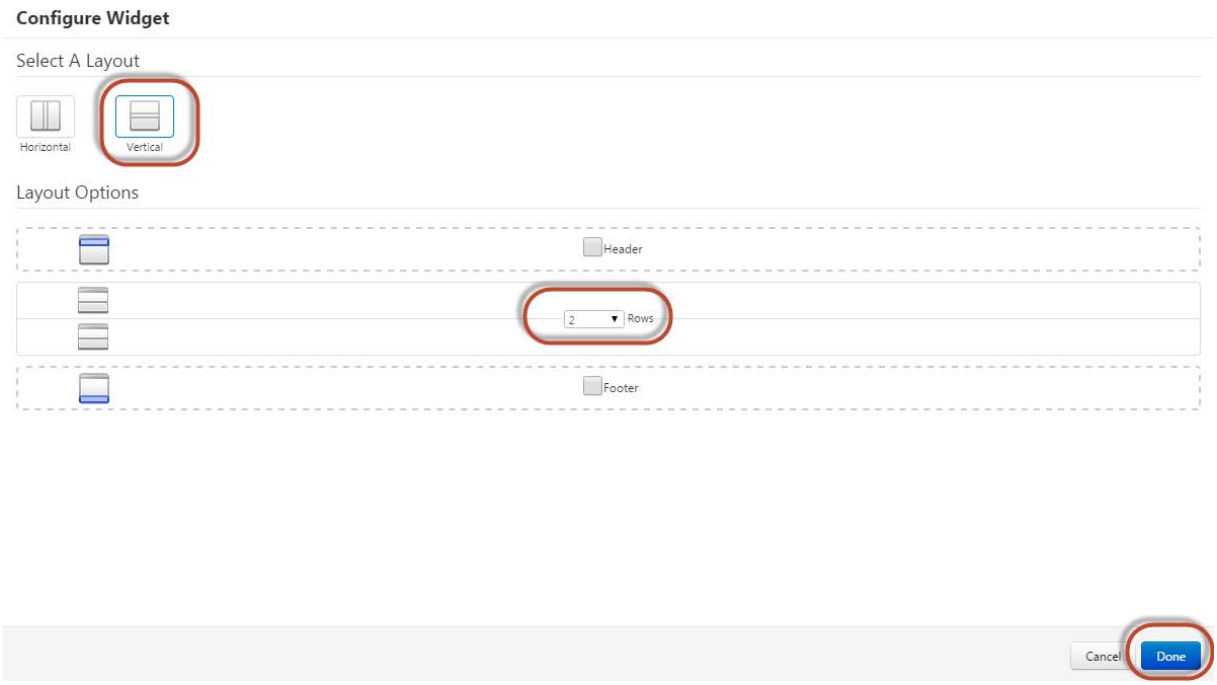
*The **Layout** Widget is a responsive container that allows you to separate a responsive container into sections. Layouts can be placed inside the responsive containers of other layouts to create more even sections.*
The Header and Footer have constant heights with responsive widths, while the sidebars have a constant width and responsive heights. The Columns and rows have both responsive widths and heights, but you can change the percent of available space they will occupy relative to the space not being occupied by headers, footers, and sidebars.

- *Use the layout widget to "split" the mashup into more responsive areas.*

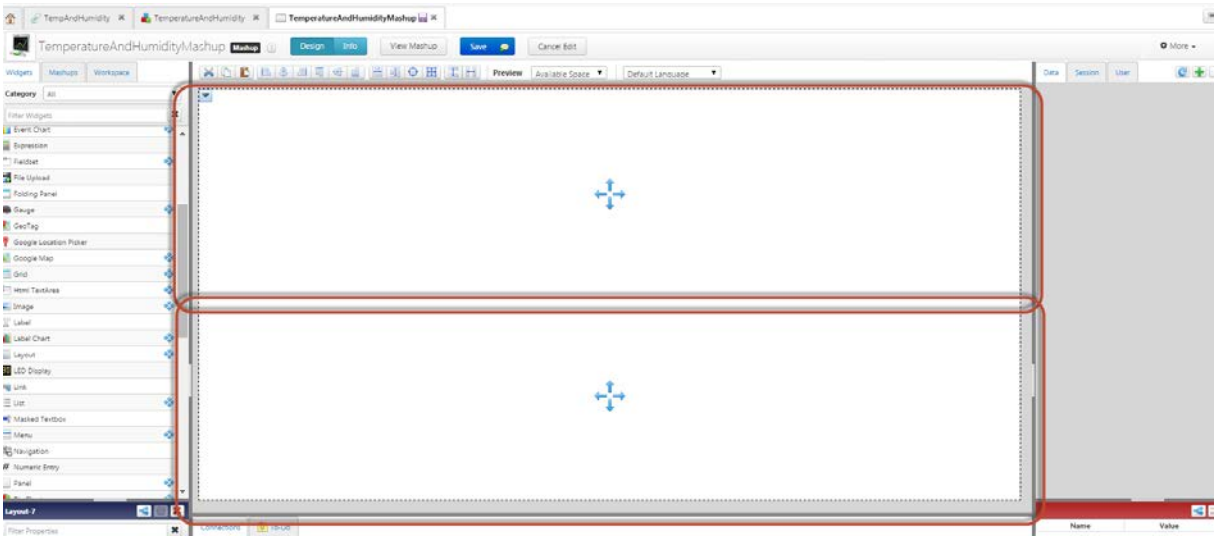
- *From the **Widgets** panel on the top left scroll down to the **Layout** widget and drag and drop it onto the Mashup*



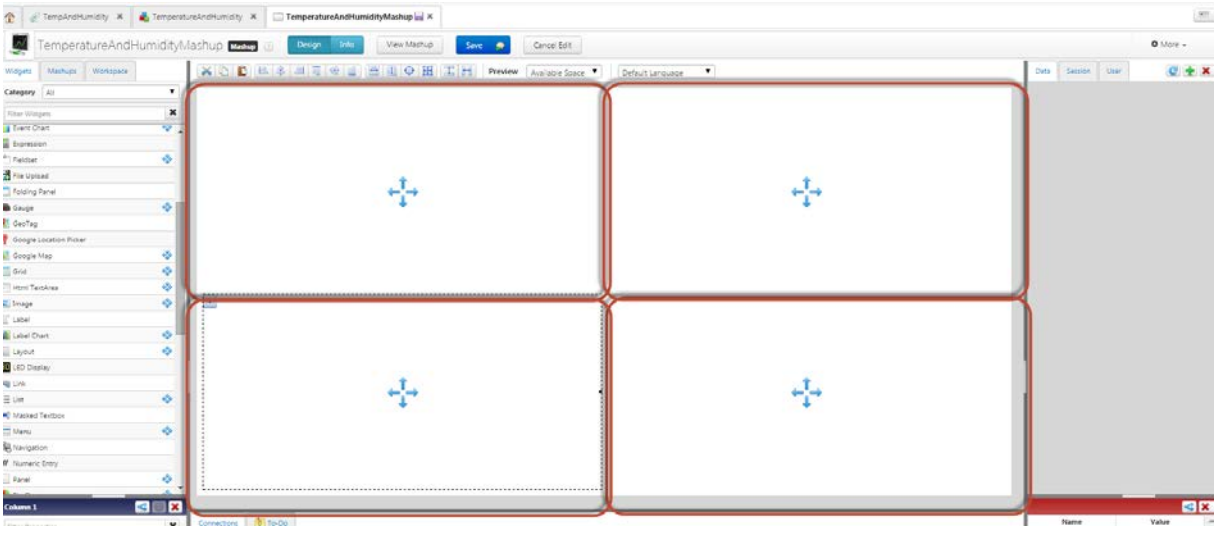
- *After you've added your layout widget, a screen pops-up in which you will configure the widget. From the "Select A layout" select* ***Vertical****, and from the "Layout Options" select two rows and press* ***Done,*** *like in the screen below:*



- *After this, the mashup has a layout widget divided in 2 rows:*



- *Add another two layout widgets, one in the upper row and the other in the bottom row. Both widgets must be Horizontal and have two rows. After you've added the widgets, the mashup looks like in the screen below:*

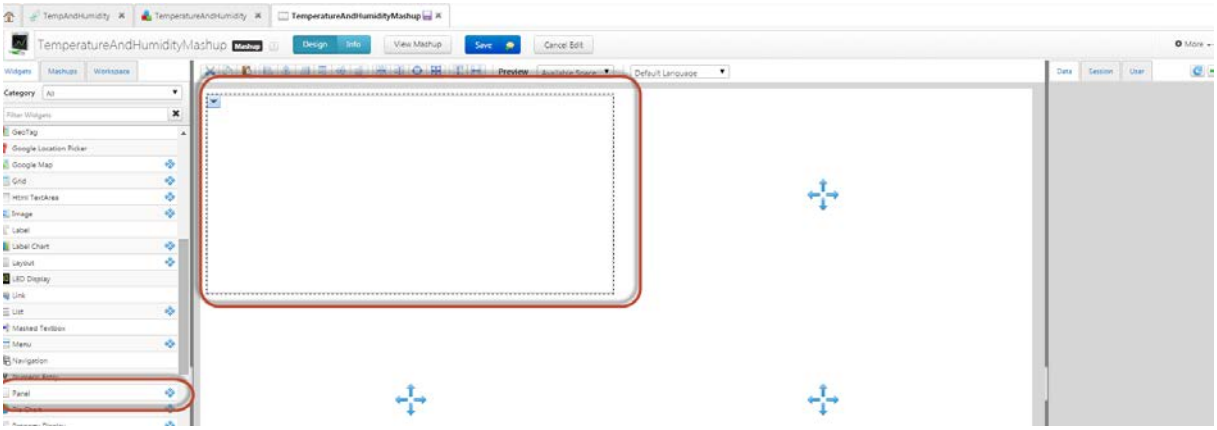


- *Now, let's add the panel widgets.*

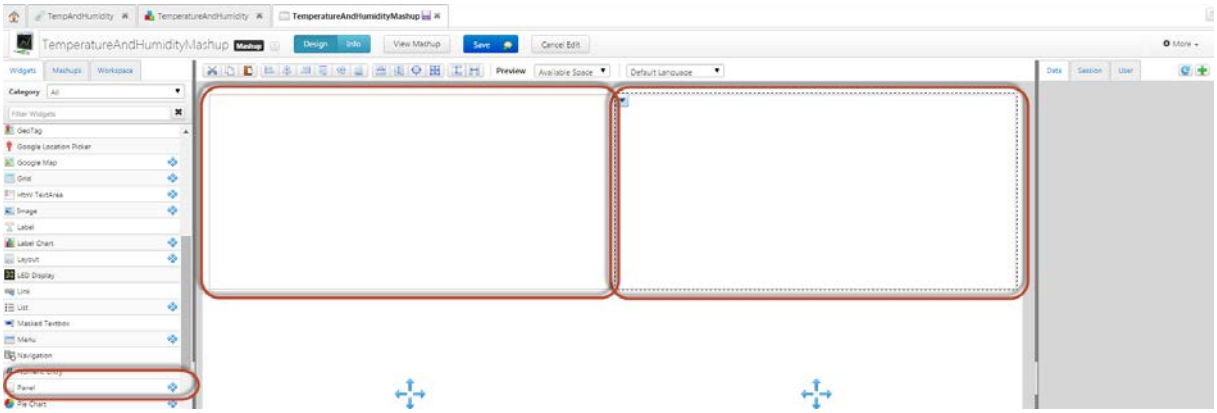
*The **Panel** Widget is a container. You can place other Widgets into a Panel and move them around as a group. You can also use a Style to change the background color of a Panel to add contrast to your Mashup. All Widgets placed in the Panel are relative to the Panel location.*

- *So, you use the panel widget in order to place the numeric entry and button widgets in it.*

- *Adding the first panel widget – you will use this to store a numeric entry widget and a button widget. The panel is responsive, so this means that it will take all the space from the column. From the **Widgets** panel on the top left scroll down to the **Panel** widget and drag and drop the widget onto the first column from the upper layout inside the mashup as shown below:*



- *Adding the second panel widget – you use this to store a numeric entry widget. The panel is responsive, so this means that it will take all the space from the column. From the **Widgets** panel on the top left scroll down to the **Panel** widget and drag and drop the widget onto the second column from the upper layout inside the mashup as shown below:*



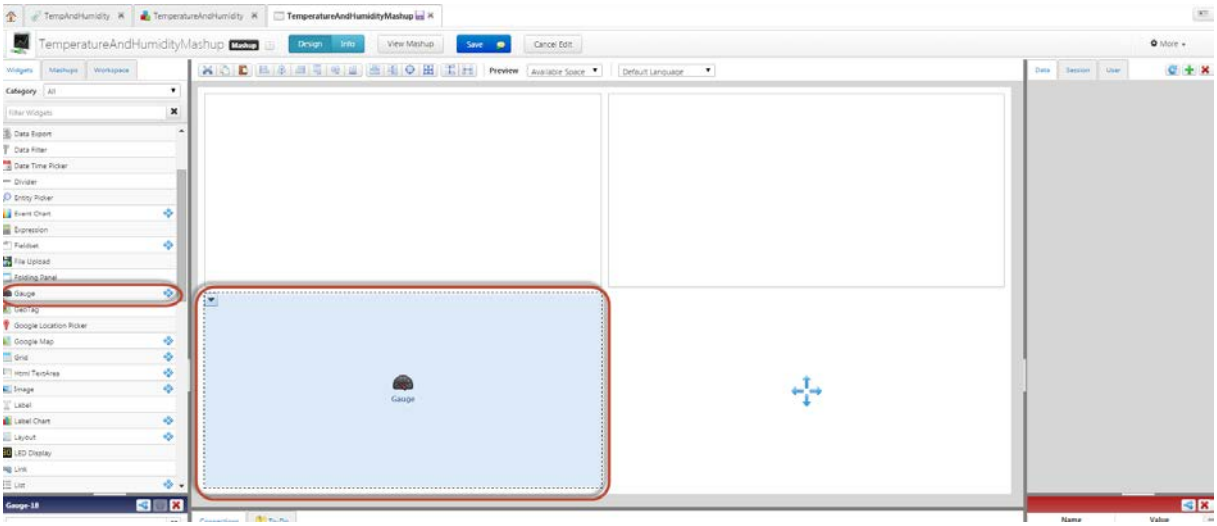
- *Now, you add the gauge widgets.*

The Gauge Widget is a single needle gauge. You can define how many sections, or intervals, the Gauge has and different color schemes for each interval.

- *You use the gauge widgets in order to show the values for the Temperature and Humidity properties.*

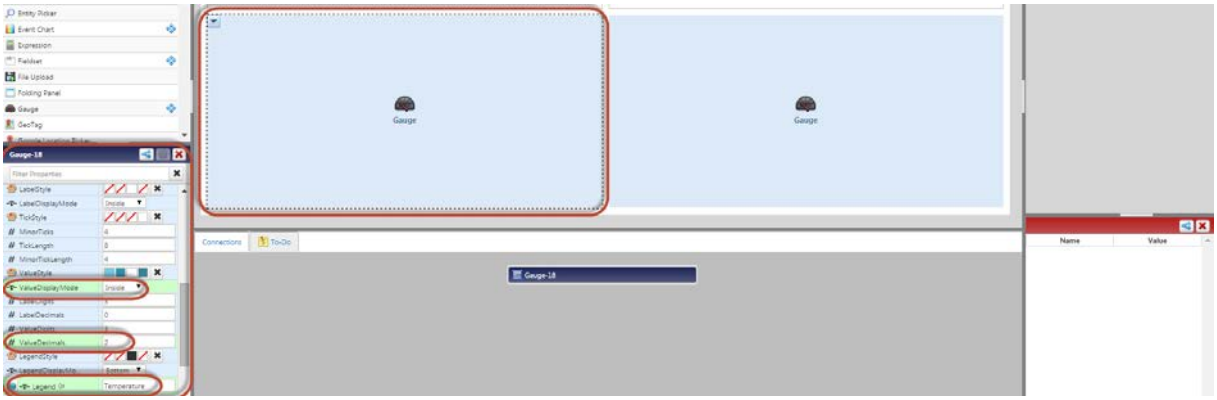
- *Adding the first **Gauge** widget – you use this widget to show the value for the Temperature property. The Gauge is responsive, so this means that it will take all the space from the column. From the **Widgets** panel on the top left, scroll down to the **Gauge** widget and drag*

and drop the widget onto the first column from the bottom layout inside the mashup as shown below:

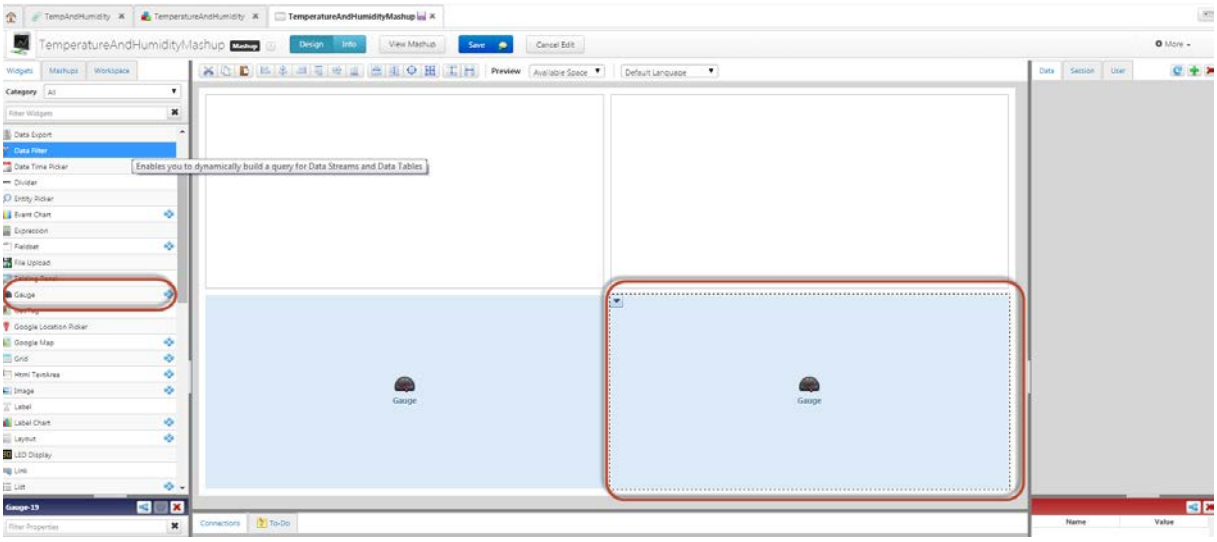


- *Next, you have to modify the properties of the first gauge widget as follows: ValueDisplayMode = Inside, ValueDecimals = 2, Legend = Temperature.*

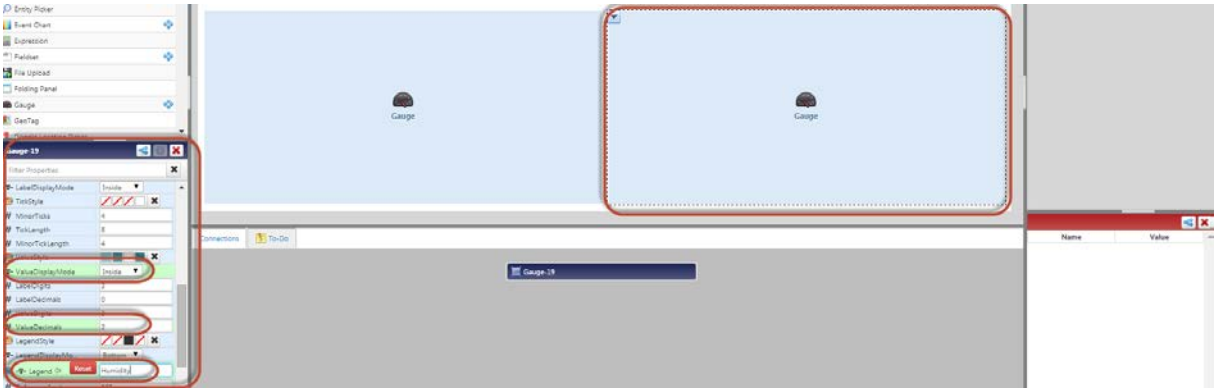
- *You modify these properties in order to better see the values displayed in the gauge (the dial) widget. The ValueDisplayMode property makes it possible to see the value that the gauge widget has inside, on the top or at the bottom of the gauge widget. The ValueDecimals property displays the two-digit temperature property value from the "TemperatureAndHumidity" thing. The Legend widget provides you with an option to describe (label) the gauge widget, that's why you set "Temperature" as value for the Legend property. You see the widget's property on the bottom left corner of the screen as shown below:*



- *Adding the second **Gauge** widget – you use this to show the value for the Humidity property. The Gauge is responsive, so this means that it will take all the space from the column. From the **Widgets** panel on the top left, scroll down to the **Gauge** widget and drag and drop the widget onto the second column from the bottom layout inside the mashup as shown below:*



- *Next, you to modify the properties of the second Gauge widget as follows: ValueDisplayMode = Inside (this displays the value inside the dial), ValueDecimals = 2 (this shows two digits for the dials value), Legend = Humidity (this labels the dial as Humidity). You see the widget's property on the bottom left corner of the screen as shown below:*



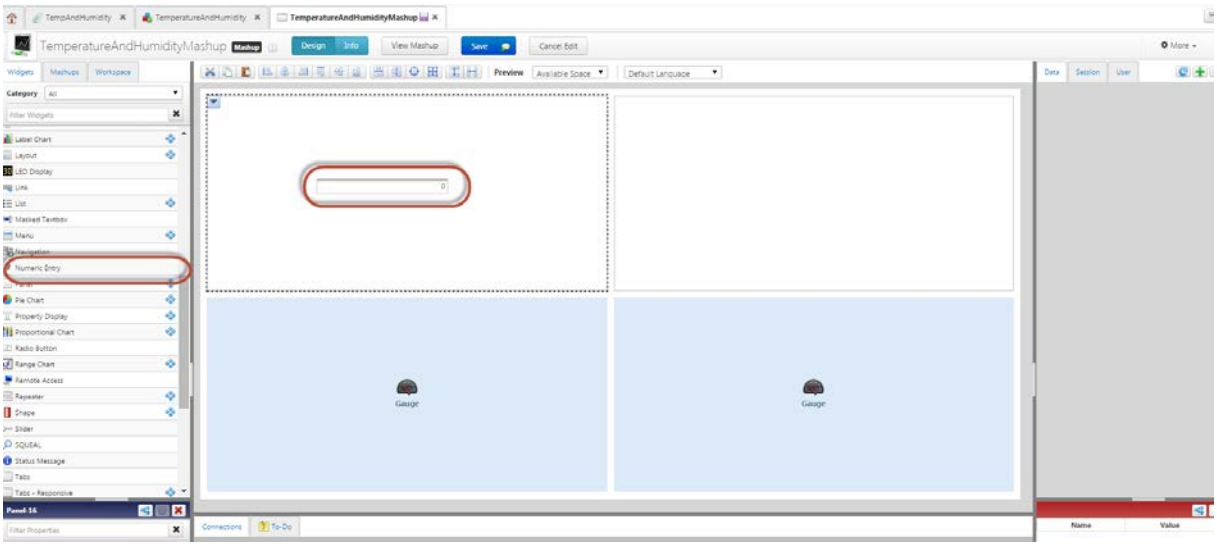
- *Now, you add the numeric entry widgets.*

The Numeric Entry widget limits the user's input to numeric values only. The values will be range limited to numbers between the minimum and maximum values if the ConstrainValue property is true. The decimal places will be limited as well to the number of FixedDigits, if the value of FixedDigits is greater than zero. There is an option to allow or disallow negatives using the AllowNegatives property.

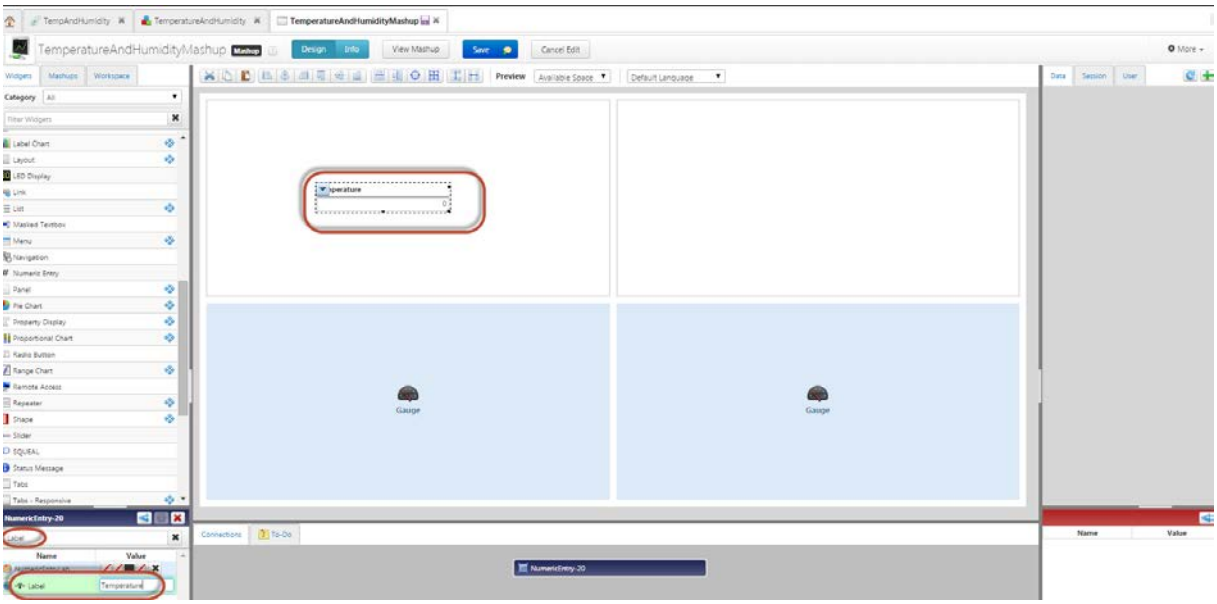
- *You use the numeric entry widgets in order to set the values for the Temperature and Humidity properties inside the thing and to set the values inside the 2 gauge widgets that you added earlier.*

- *Adding the first numeric entry widget – you use this to set the Temperature property inside the thing and to set the value for the first gauge widget. From the* ***Widgets*** *panel on the top*

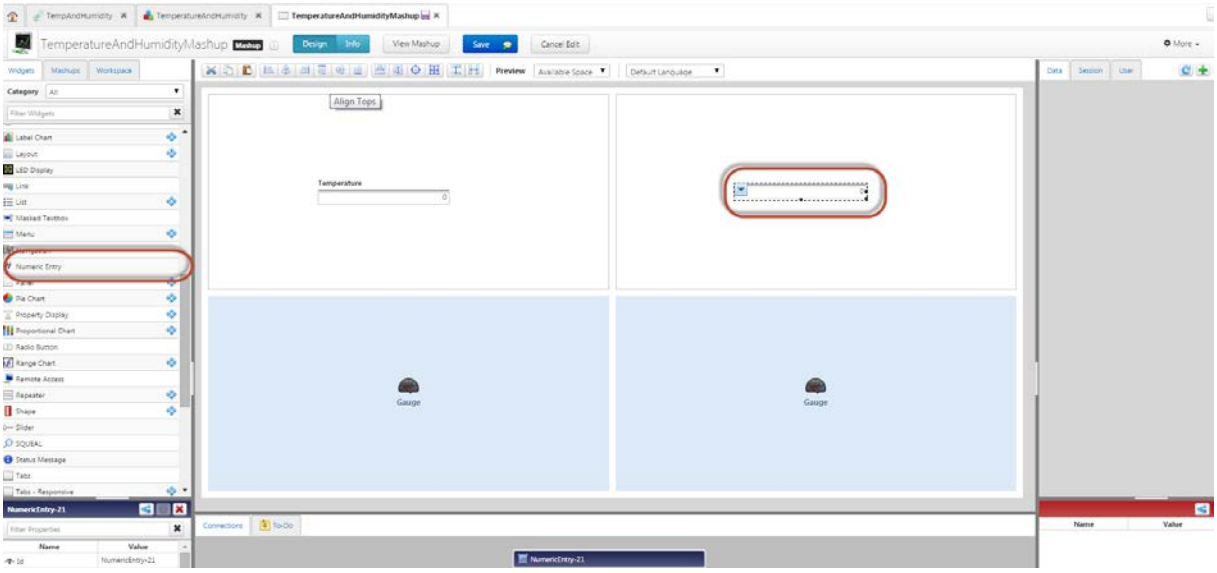
*left, scroll down to the **Numeric Entry** widget and drag and drop the widget onto the first panel as shown below:*



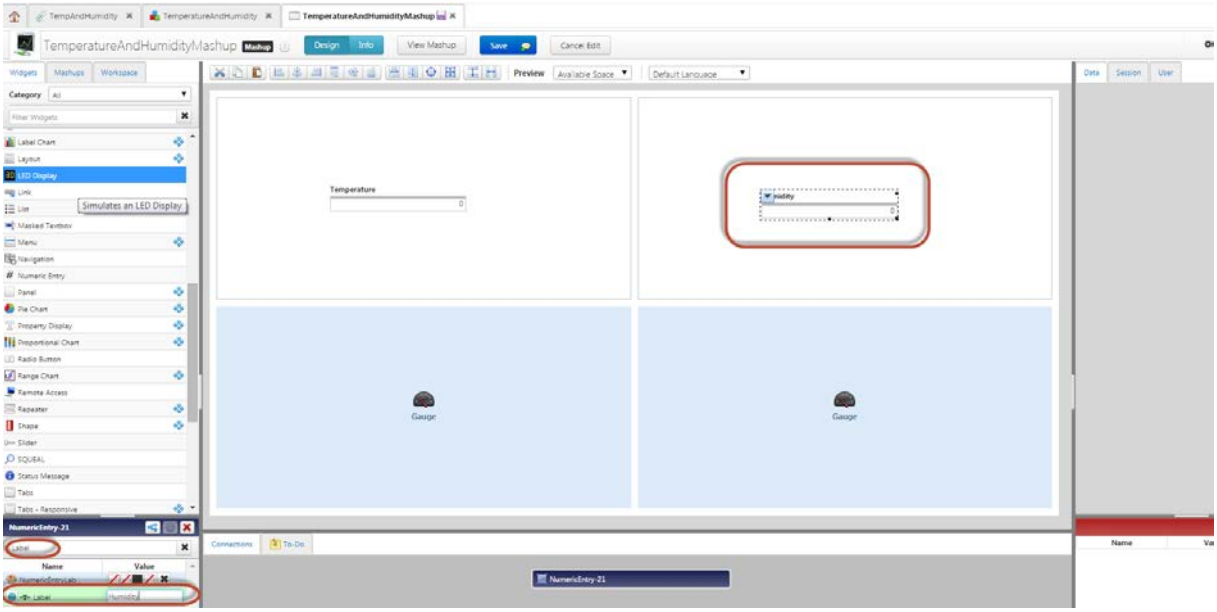
- *Next, you modify the properties of the numeric entry widget as follows: Label = Temperature. You see the widget's properties on the bottom left corner of the screen. You either scroll to the Label property, or you just type label inside the filter properties as shown below:*



- *Adding the second numeric entry widget – you use this to set the Humidity property inside the thing and to set the value for the first gauge widget. From the **Widgets** panel on the top left, scroll down to the **Numeric Entry** widget and drag and drop the widget onto the first panel as shown below:*



- *Next, you modify the properties of the second numeric entry widget as follows: Label = Humidity. You see the widget's properties on the bottom left corner of the screen. You either scroll to the Label property, or you just type label inside the filter properties as shown below:*

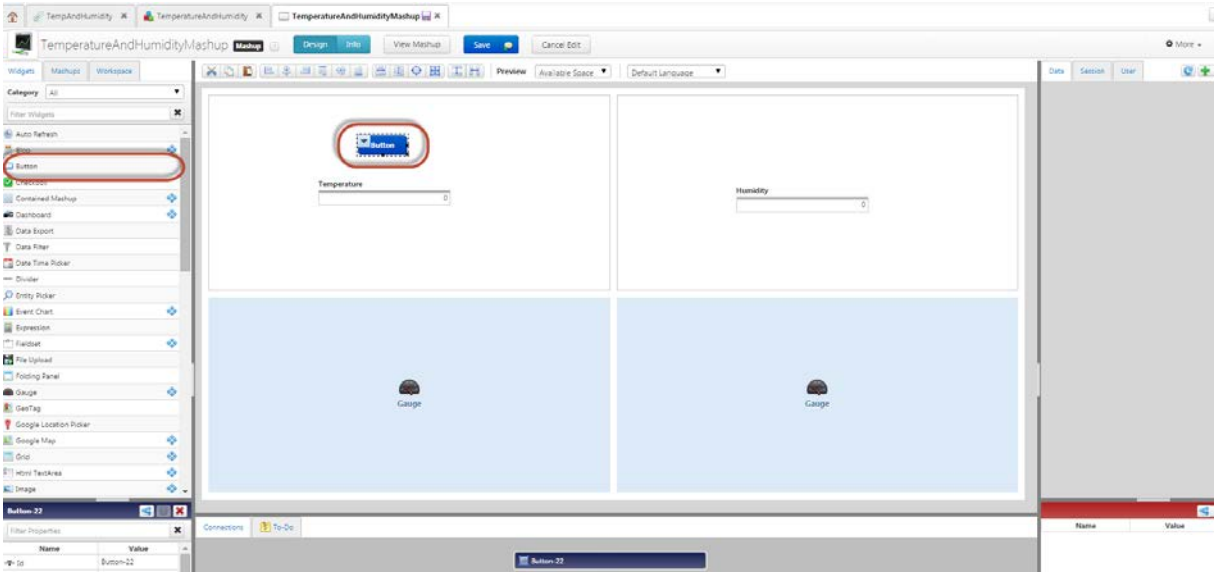


- *Now, you add the button widgets.*

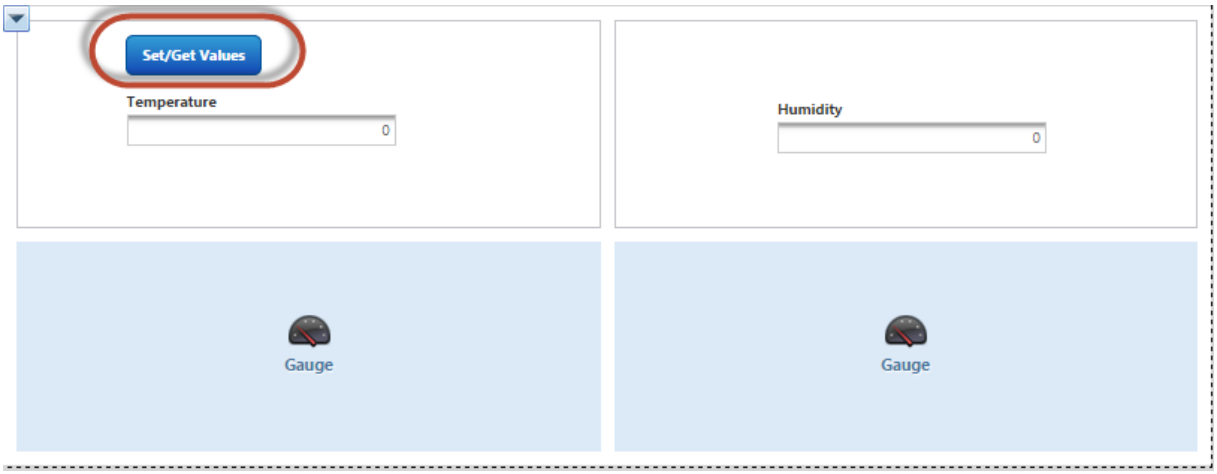
*The Button widget raises an **Event** when clicked.*

- *So, you use the button widget in order to modify the values from the Gauges with the values set in the numeric entry. The steps will be as follow: first, you set a value in either of the two numeric entry widgets followed by pressing the button widget. After you press the button widget, you can see the value(s) of the Gauge(s) modifies according to what you've set in the numeric entry.*

- *Adding a button widget – from the **Widgets** panel on the top left, scroll down to the **Button** widget and drag and drop the widget onto the first panel as shown below:*



- *Next, you modify the properties of the button widget as follows: Label = Set/Get Values. You see the widget's properties on the bottom left corner of the screen. You either scroll to the Label property, or you just type label inside the filter properties as shown below:*



- *Now there are some widgets on the Mashup but no data to bind them to. In next step you add some data to bind with these widgets.*

Step 5: Add Services and Bind the Data inside the TemperatureAndHumidityMashup Mashup

A) Add Services to the TemperatureAndHumidityMashup Mashup:

ThingWorx Data Services are what you call to get data into a mashup and then render the data using a visualization widget. To render any data in a mashup, you need to add one or more Data Services to the mashup. One Data Service can feed multiple visualization widgets. This minimizes the amount of data calls to the server. Most widgets are configurable so that only the columns of data from the Data Service that you wish to see in that widget are shown.

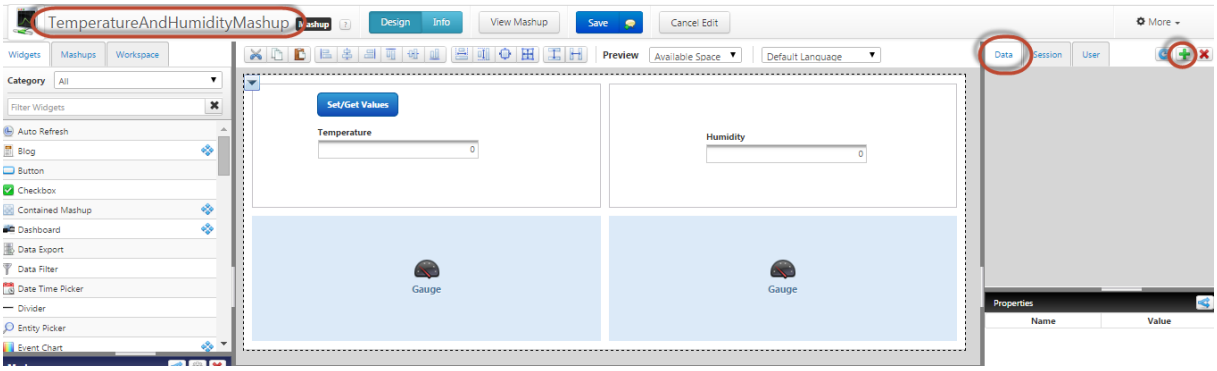
- *You use this mashup to set values in the thing's properties and to also get values from the thing's properties. To achieve this, you need to add 2 services (GetProperties and SetProperties). The services are generic services that come along from the Generic Template associated with the Thing.*

- *SetProperties service – this service, like the name points out, sets the value on a thing's property. In this case, it sets the temperature and humidity value. To use the SetProperties service for a thing, you must have at least one property defined on your thing. To bind a widget with the SetProperties service of a thing, you must have in the SetProperties data only information about that thing's property or properties. In this case, there are only two parameters: Temperature and Humidity, the two properties that you set on the "TemperatureAndHumidity" thing. In our mashup, the binding is made between the numeric temperature and humidity data entry widgets where and the two parameters (Temperature and Humidity) from the SetProperties service. So, the SetProperties service binds the Temperature numeric entry widget to the Temperature thing property (through the Temperature parameter of the service), and the Humidity numeric entry widget to the Humidity thing property (through the Humidity parameter of the service). In order for the values to be set inside the two properties you also need to bind a button to the SetProperties service. The button triggers the set event and updates the values inside the properties.*

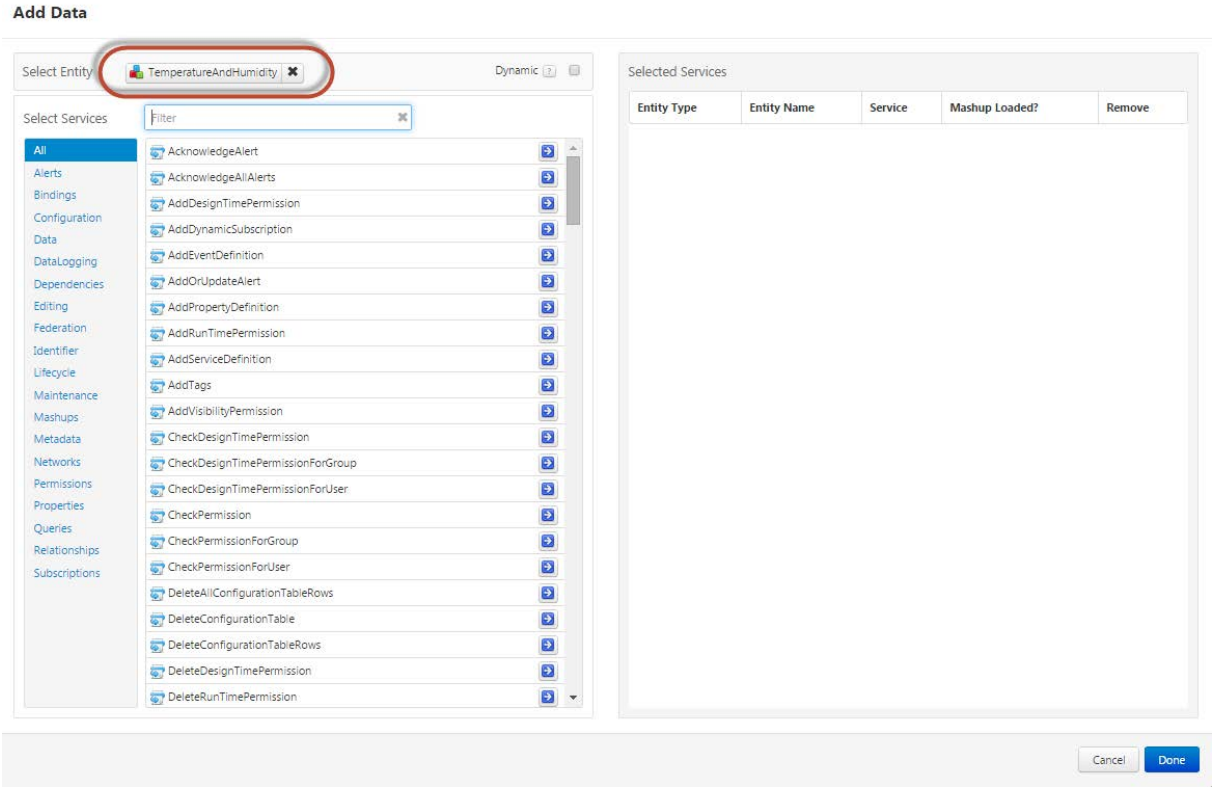
- *GetProperties service - this service, like the name points out, retrieves data value from a thing's property or data values from multiple thing properties. In this case it gets the temperature and humidity data values. In order to use the GetProperties service for a thing, you need to have at least one property and data value defined on your thing. In this mashup, the binding is made between the Temperature and Humidity properties values in the data entry boxes and the corresponding Temperature and Humidity Gauges.*

- *Refresh the mashup by using the **Reload** button from the mashup and the values from the Temperature and Humidity gauge widgets automatically change according to what is set in your thing's properties.*

- *In the mashup, click the green **+** button next to the **Data** tab on the top right hand side of the Mashup Editor as shown below:*



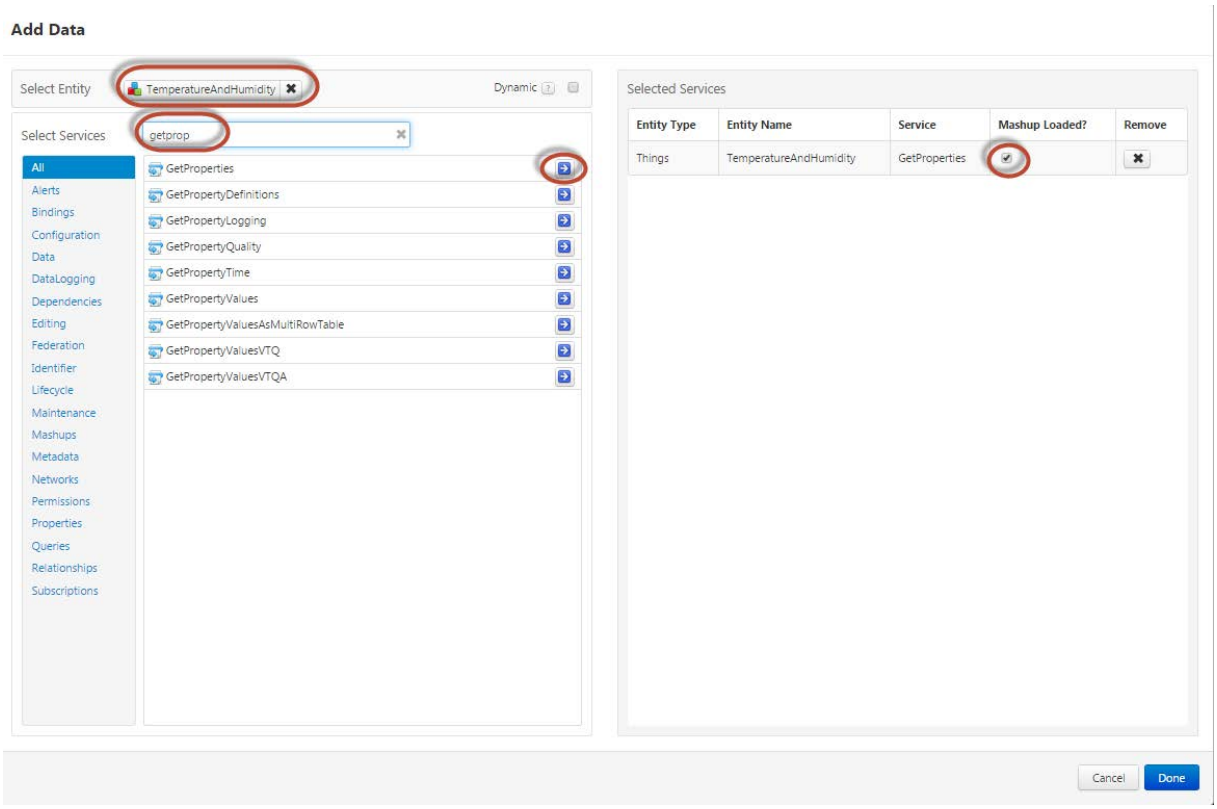
- *In the **Search Entities** textbox of the window that opens, enter "Temperature" and the TemperatureAndHumidity thing you just created appears in the filtered list. Select it.*



- *In the **Select Services** textbox, enter "getprop" to filter the exposed services for the thing. Click the blue arrow button on the right hand side of the GetProperties service to add it to the Selected Services list.*

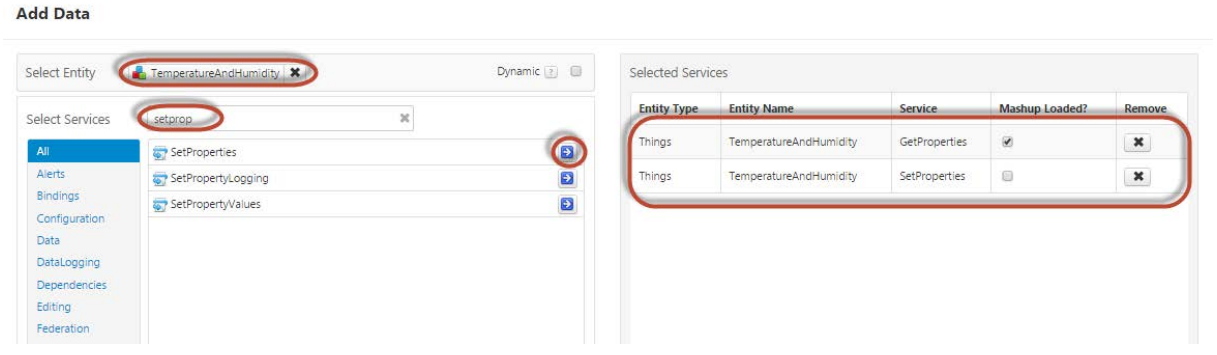
*Why do I have to choose a **Service** to access the data of my **Thing**? At runtime, your **Thing's** data must be transported to your web browser. Your web browser makes a service call to request the current service values of your **Thing** to display in your **Widgets** and if you modify that data, you will have to make another service call to save it as well. Any service you define on a Thing can be bound to and called from a **Widget**.*

- *Check the "**Mashup Loaded?"** checkbox. This causes the GetProperties Service to be called when your mashup page is first opened.*



*What does "Mashup Loaded?" really do? It allows you to make a **Service** call as soon as all of the Widgets on your page have finished loading. Usually, this service calls job is to fetch the data to initially populate your Widgets.*

- *In the **Select Services** textbox, enter "setprop" to filter the exposed services for the thing. Click the blue arrow button on the right hand side of the SetProperties service to add it to the Selected Services list. The screen appears as shown below:*



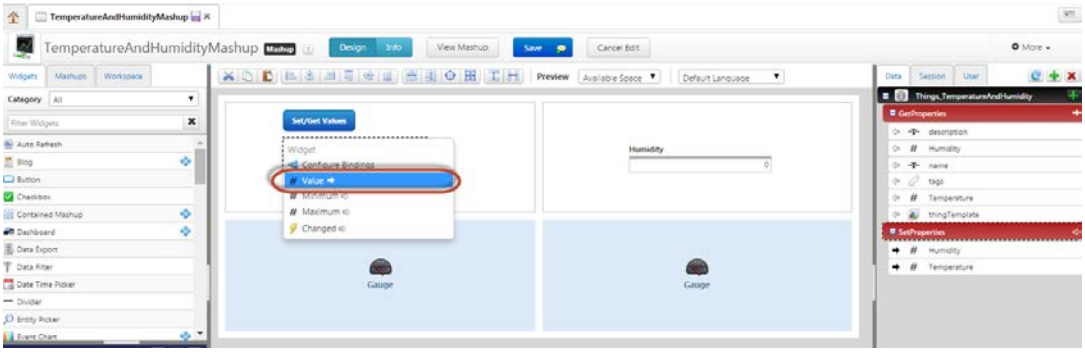
- *Click on the **Done** button to see this service listed in the **Data** tab of the Mashup Editor.*

- *You now have all the data that you need to bind with the widgets from the mashup. In next step you bind some data from the "TemperatureAndHumidity" thing to these widgets.*

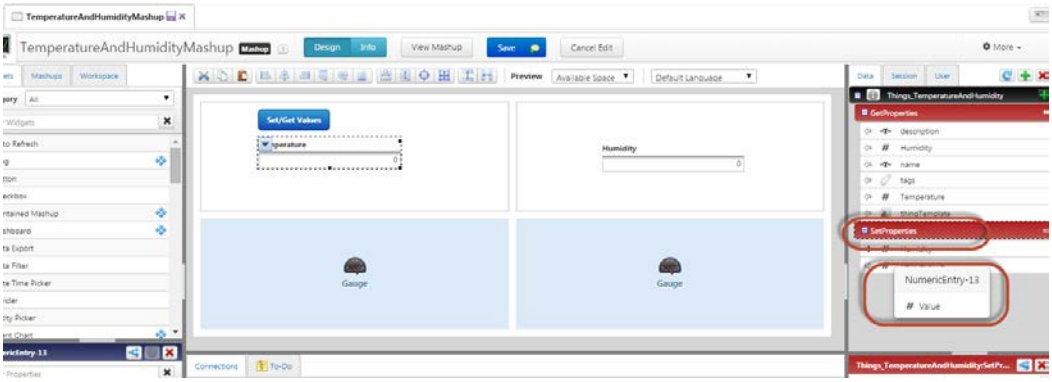
B) Bind the Data between the Widgets and the Services:

1) ***Set the Temperature and Humidity properties from the mashup*** *– to do this, you need to bind the two numeric data entry box widgets to the Things properties (Temperature and Humidity) and then bind the "Set/Get Values" button to the SetProperties service:*

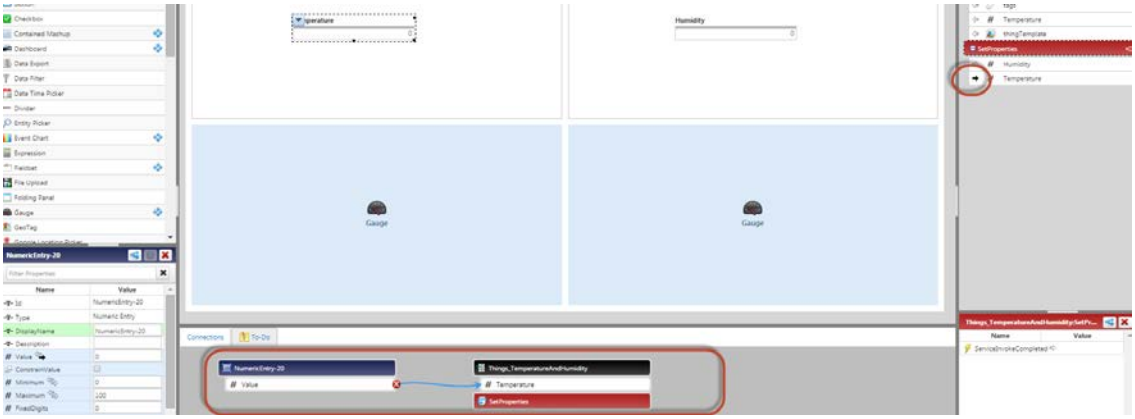
   a) Binding the first numeric entry (Temperature) with the Temperature property of the thing *– click the Temperature numeric entry, hover over the left arrow, this opens a menu from which you select* ***Value*** *as shown below:*



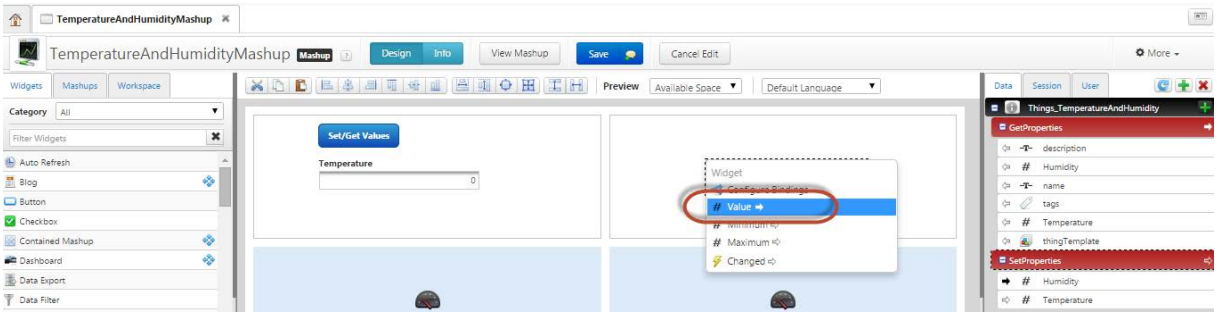
- *Next, drag and drop the* ***Value*** *to the Temperature property from the "SetProperties" service as shown below:*



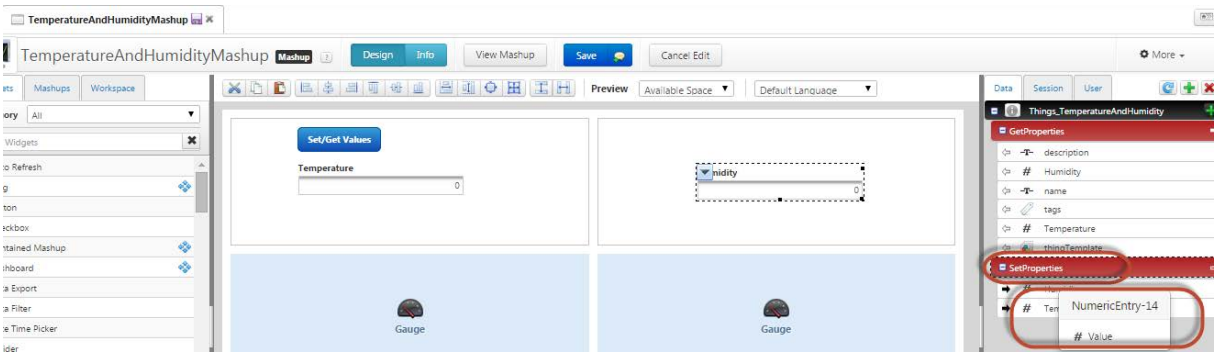
- *If the binding is correct, the arrow from Temperature is black and you can see in the* ***Connections*** *tab from the bottom of the mashup the bindin as shown below:*



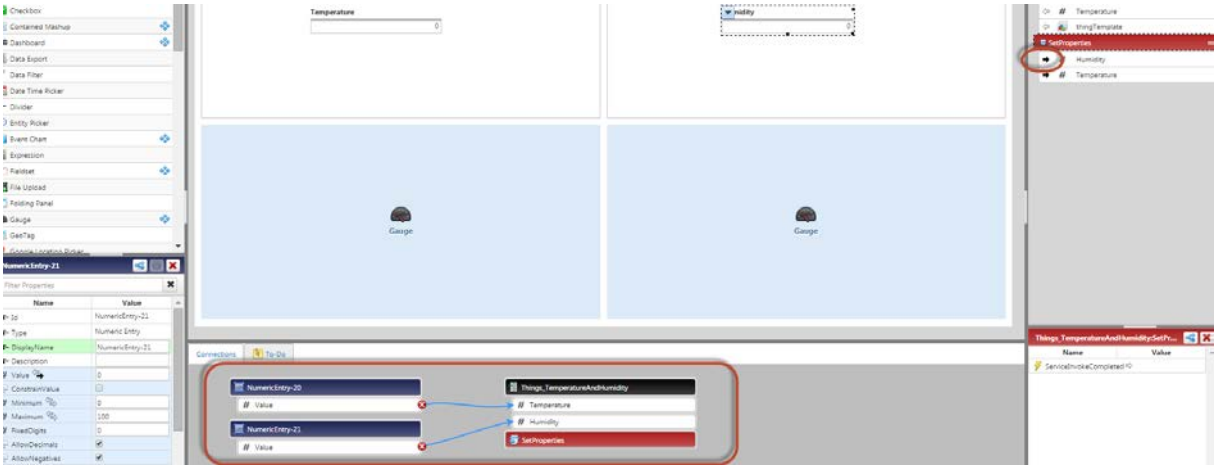
b) *Binding the second numeric entry (Humidity) with the Humidity property of the thing* – *click the Humidity numeric entry, hover over the left arrow, this opens a menu from which you select **Value** as shown below:*



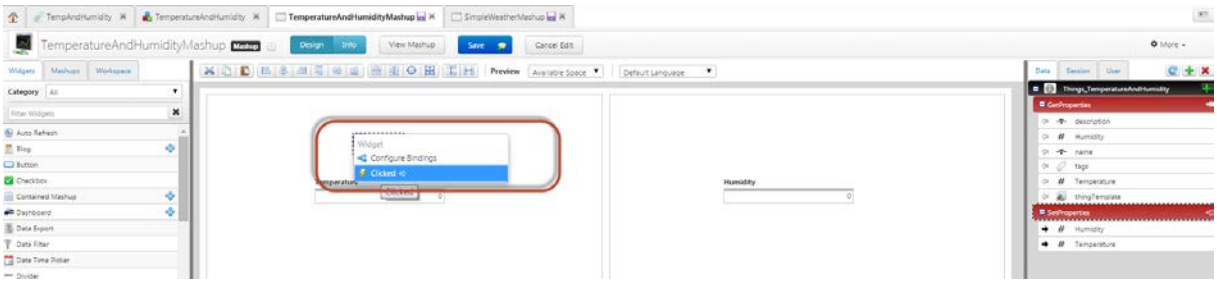
- *Next, drag and drop the **Value** to the Humidity property from the "SetProperties" service as shown below:*



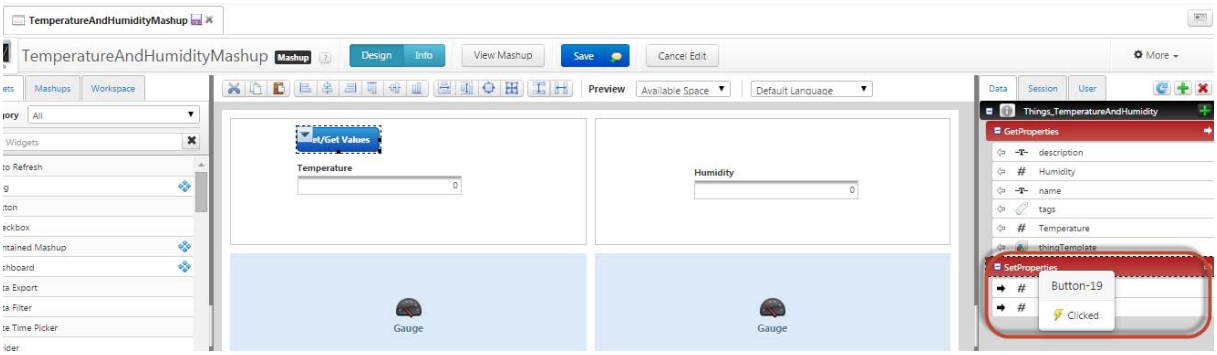
- *If the binding is correct, the arrow from Humidity is black and you can see in the **Connections** tab from the bottom of the mashup the binding as shown below:*



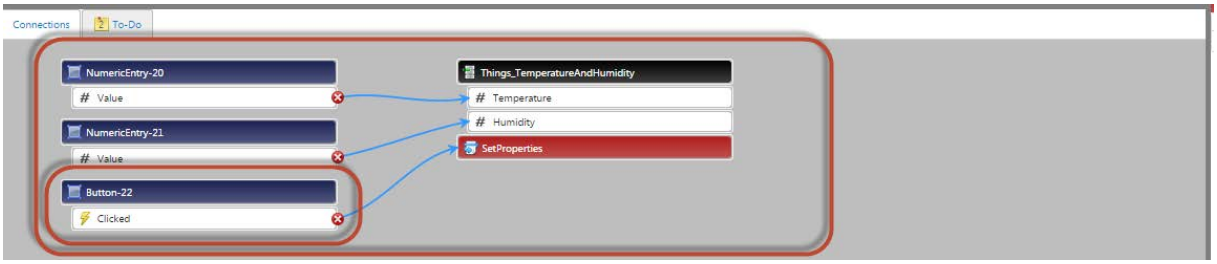
c) *Binding the Button (Set/Get Values) with the SetProperties service* – *click the Set/Get Values button, hover over the left arrow, this opens a menu from which you select **Clicked** as below:*



- *Drag and drop the **Clicked** value to the "SetProperties" service as shown below:*

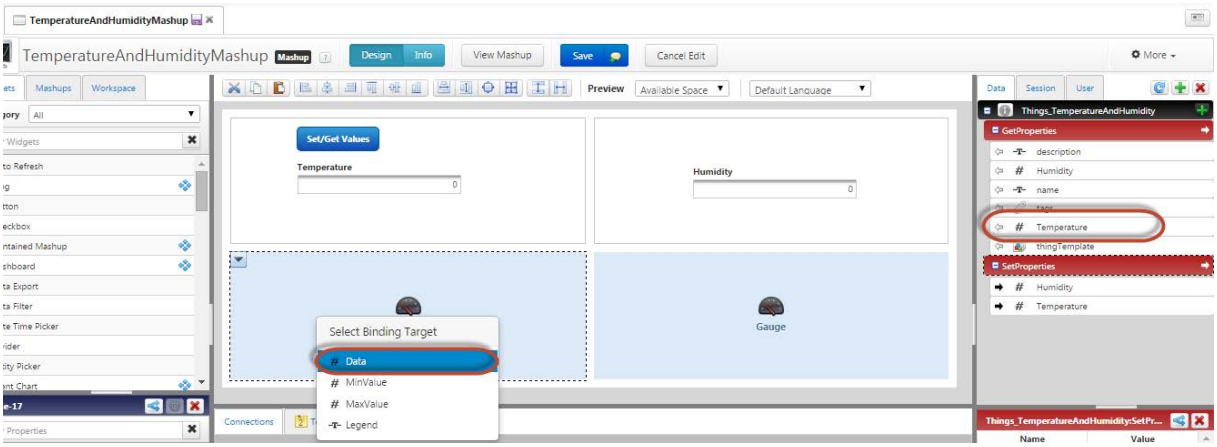


- *If the binding is correct, you can see in the **Connections** tab from the bottom of the mashup the binding as shown below:*

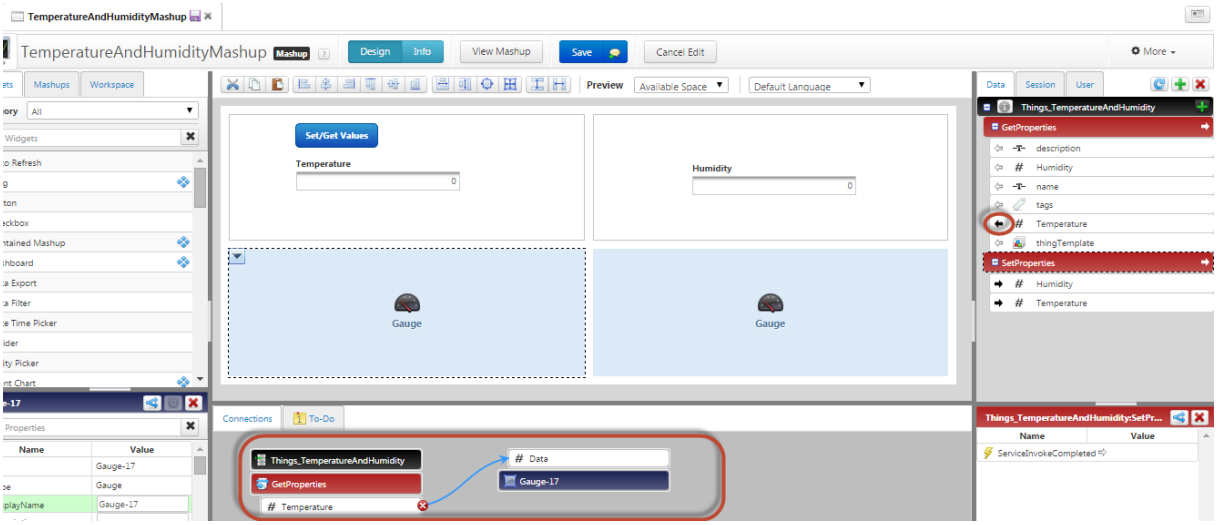


d) Set the Temperature and Humidity values of the gauges straight from the mashup – *bind the two numeric entry widgets (Temperature and Humidity), the two gauges and the button to the GetProperties service;*

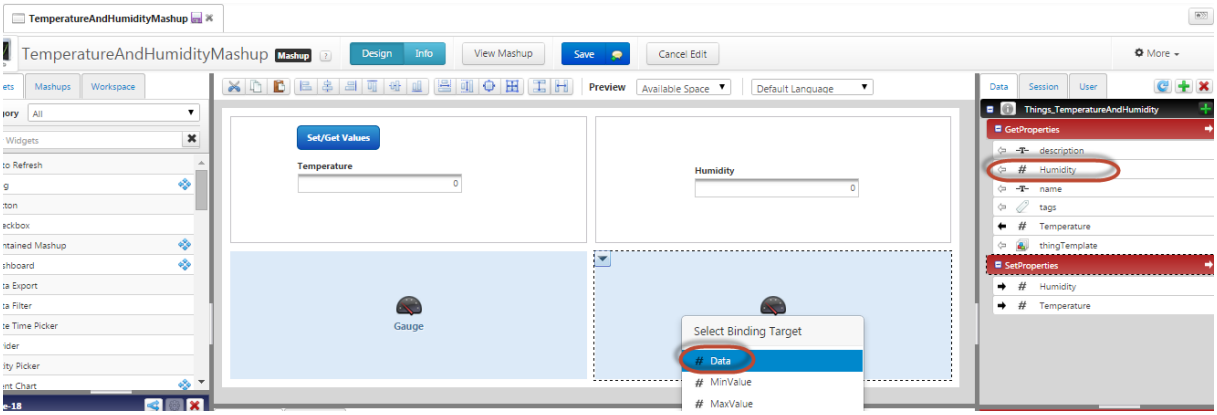
e) Binding the first gauge widget to the Temperature from the GetProperties service *- drag and drop **Temperature** from the GetProperties service into the first gauge widget and choose **Data** as shown below:*



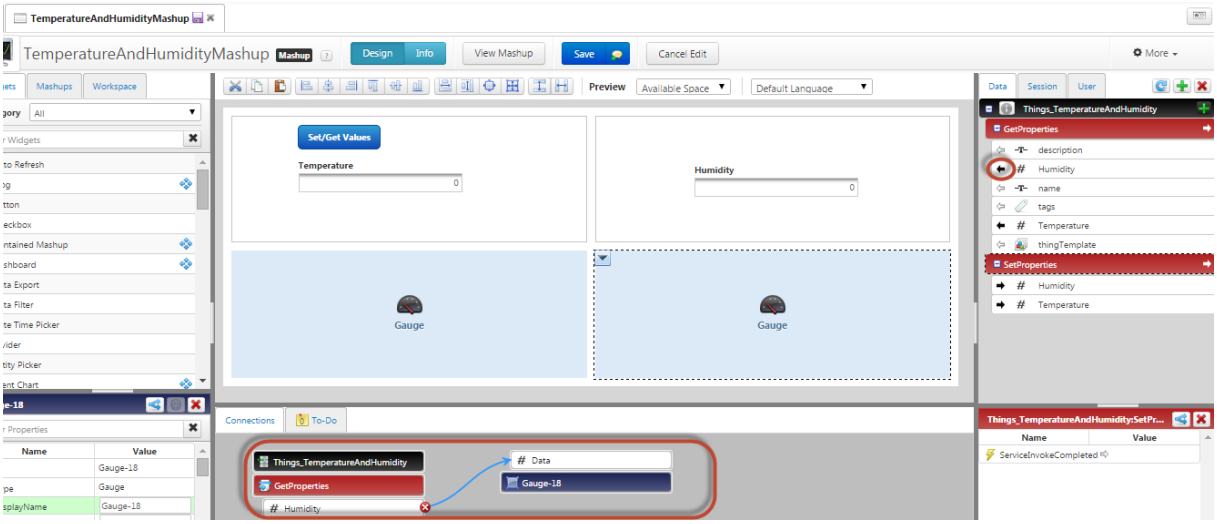
- *If the binding is correct, the arrow from Temperature is black and you can see in the* ***Connections*** *tab from the bottom of the mashup the binding as shown below:*



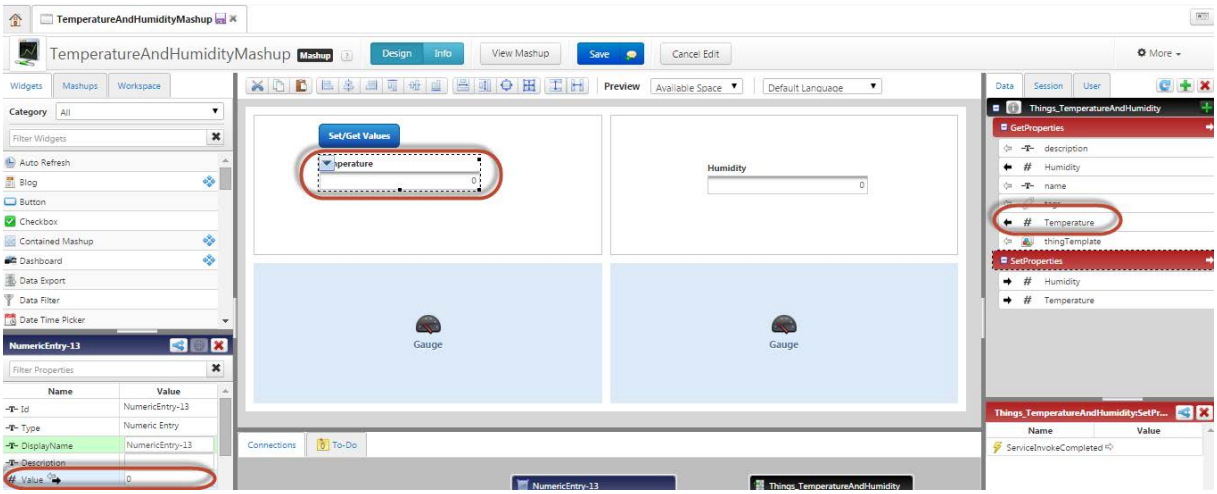
f) ***Binding the second gauge widget to the Humidity from the GetProperties service*** *- drag and drop* ***Humidity*** *from the GetProperties service into the second dauge widget and choose* ***Data*** *as shown below:*



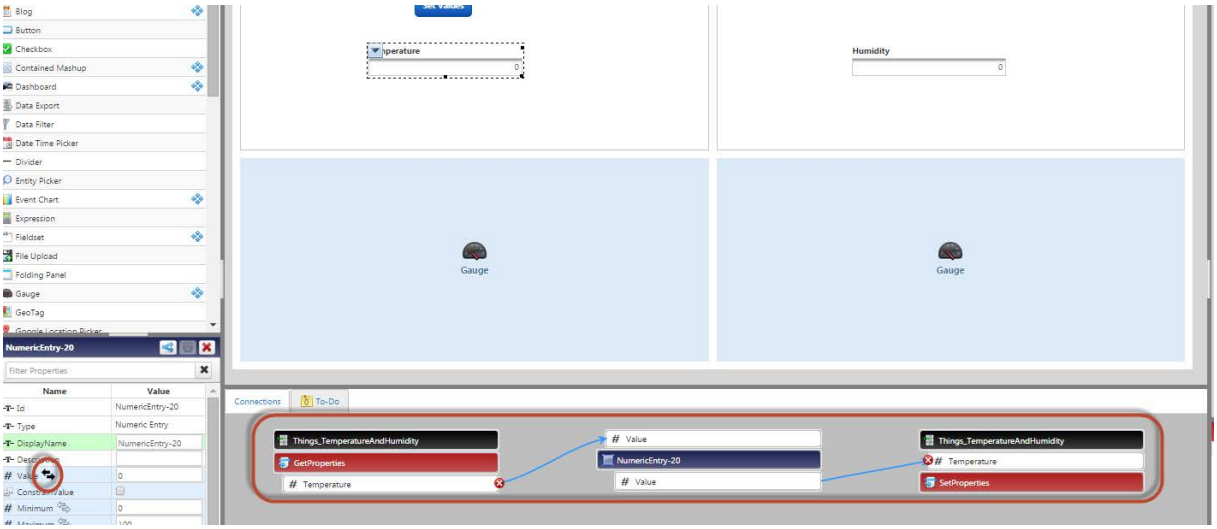
- *If the binding is correct, the arrow from Humidity is black and you can see in the **Connections** tab from the bottom of the mashup the binding as shown below:*



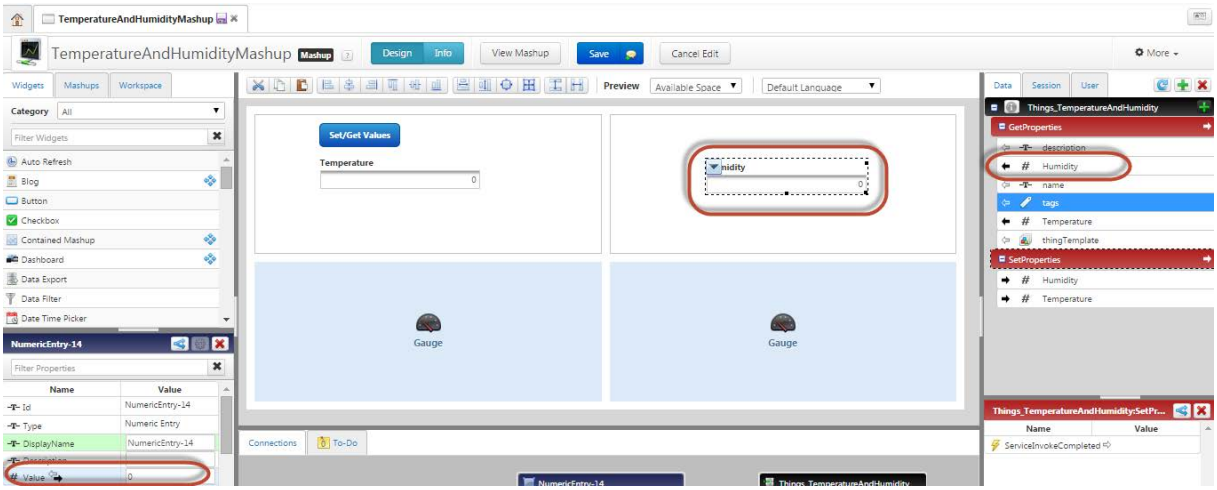
g) Binding the Temperature from the GetProperties service to the first numeric entry (Temperature) *– Select the Temperature numeric entry. Then, drag and drop the Temperature from the GetProperties service to the **Value** of the Temperature numeric entry as shown below:*



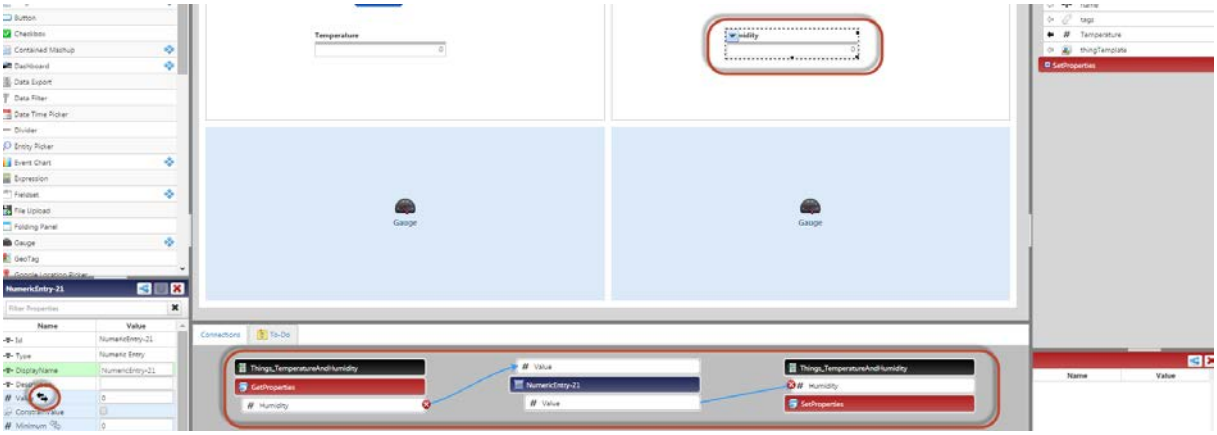
- *If the binding is correct, the arrows from the **Value** properties of the Temperature numeric entry are both black and you can see in the **Connections** tab from the bottom of the mashup the binding as shown below:*



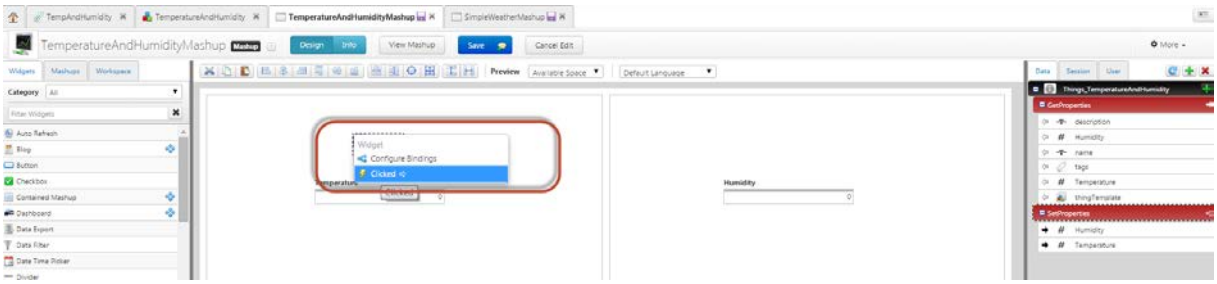
h) *Binding the Humidity from the GetProperties service to the second numeric entry (Humidity)* – *Select the Humidity numeric entry. Then, drag and drop the Humidity from the GetProperties service to the **Value** of the Humidity numeric entry as shown below:*



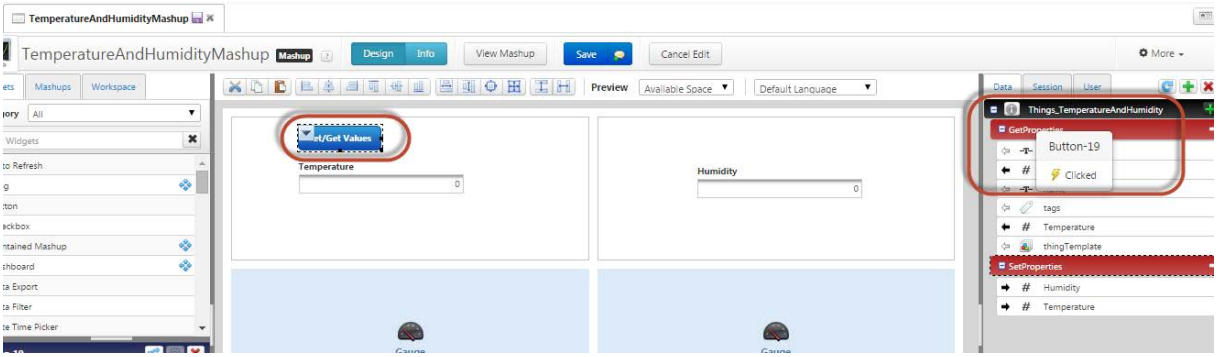
- *If the binding is correct, the arrows from the **Value** properties of the Humidity numeric entry are both black and you see in the **Connections** tab from the bottom of the mashup the binding as shown below:*



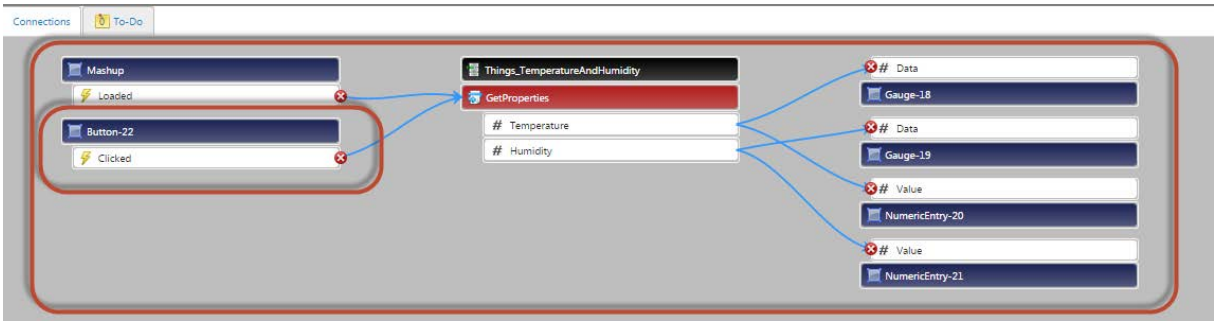
i) ***Binding the Button (Set/Get Values) with the GetProperties service*** – *click the* ***Set/Get Values*** *button, hover over the left arrow, this opens a menu from which you select* ***Clicked*** *as shown below:*



- *Next, drag and drop the* ***Clicked*** *to the "GetProperties" service as shown below:*

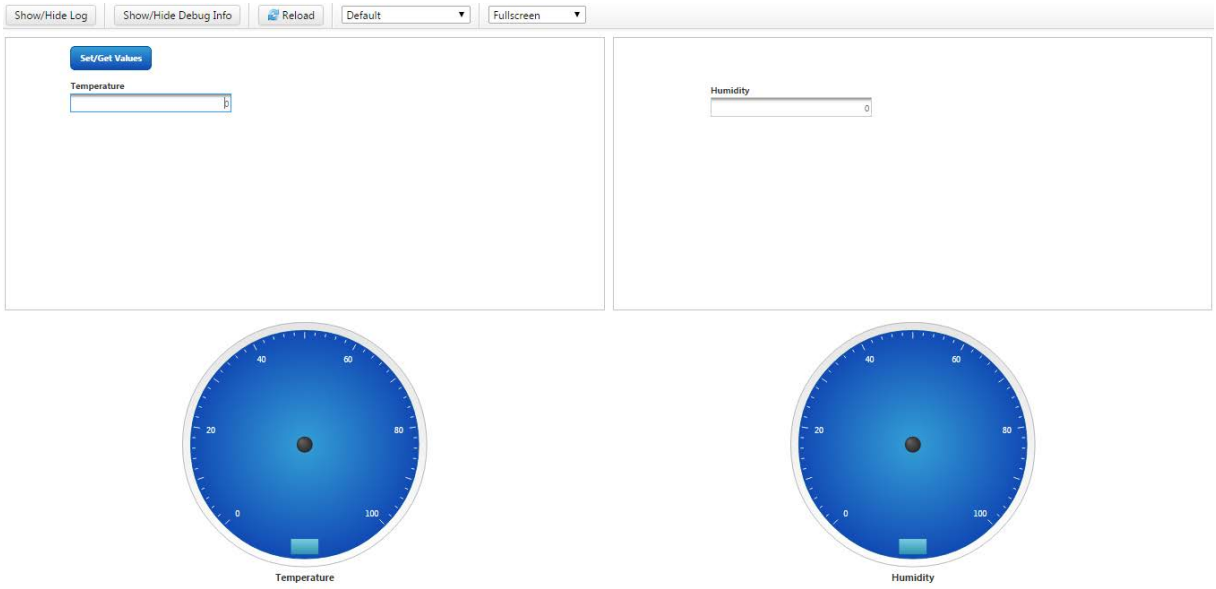


- *If the binding is correct, you can see in the* ***Connections*** *tab from the bottom of the mashup the binding as shown below:*



- *Now, save the mashup by pressing the "Save" button.*

- *Now, you can view the mashup. To do this, press the* ***View Mashup*** *button. If that doesn't work (on some browsers it doesn't work), first press the* ***Cancel Edit*** *button and then press the* ***View Mashup*** *button. The mashup looks like this:*

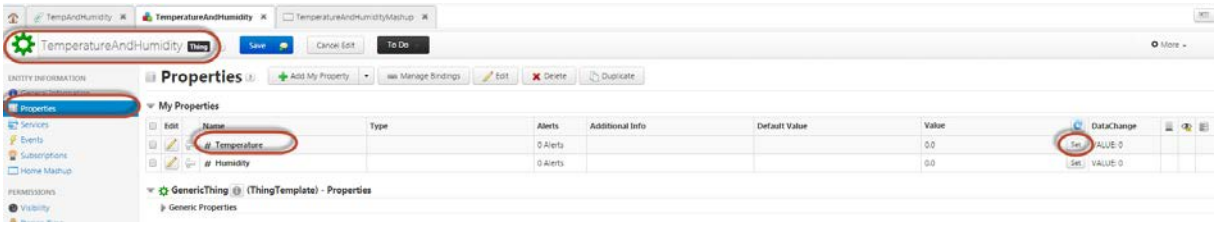


Now, let's check if the mashup works.

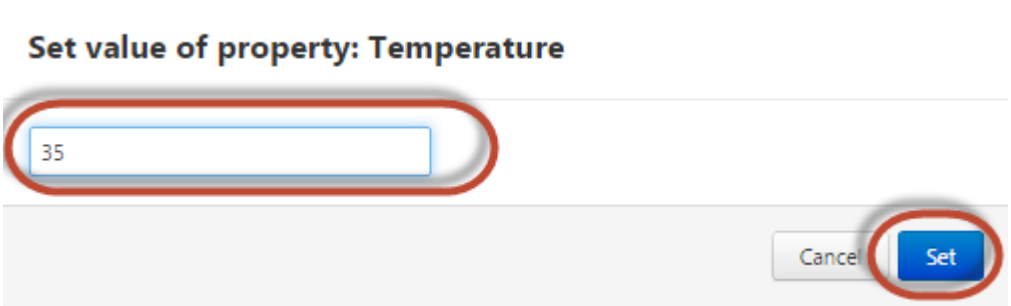
1) Check that you can change the values inside the mashup by setting the values inside the thing's properties:

- *Go to the TemperatureAndHumidity thing and set both Temperature and Humidity properties manually, by selecting* ***Properties*** *under the Entity Information section from the* ***TemperatureAndHumidity*** *thing****.***

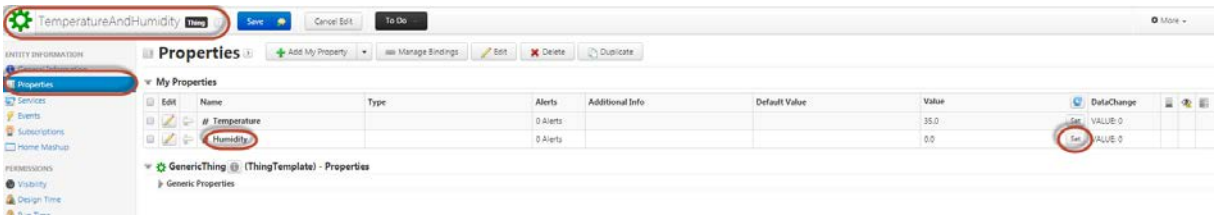
- *Next, press the* ***Set*** *button for the Temperature property as shown below:*



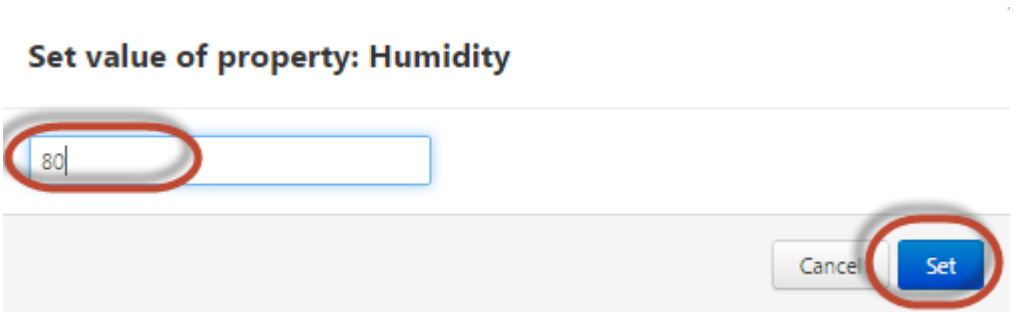
- *Next, enter a value in the* ***Set value of property: Temperature*** *box (for example 35) and press the* ***Set*** *button as shown below:*



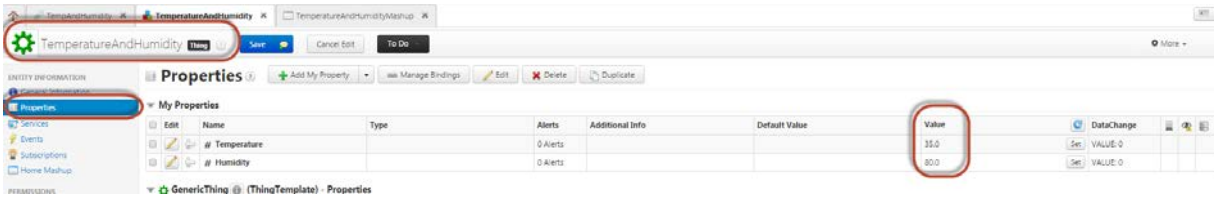
- *Next, press the* ***Set*** *button for the Humidity property as shown below:*



- *Next, enter a value in the **Set value of property: Humidity** box (for example 80) and press the **Set** button as shown below:*

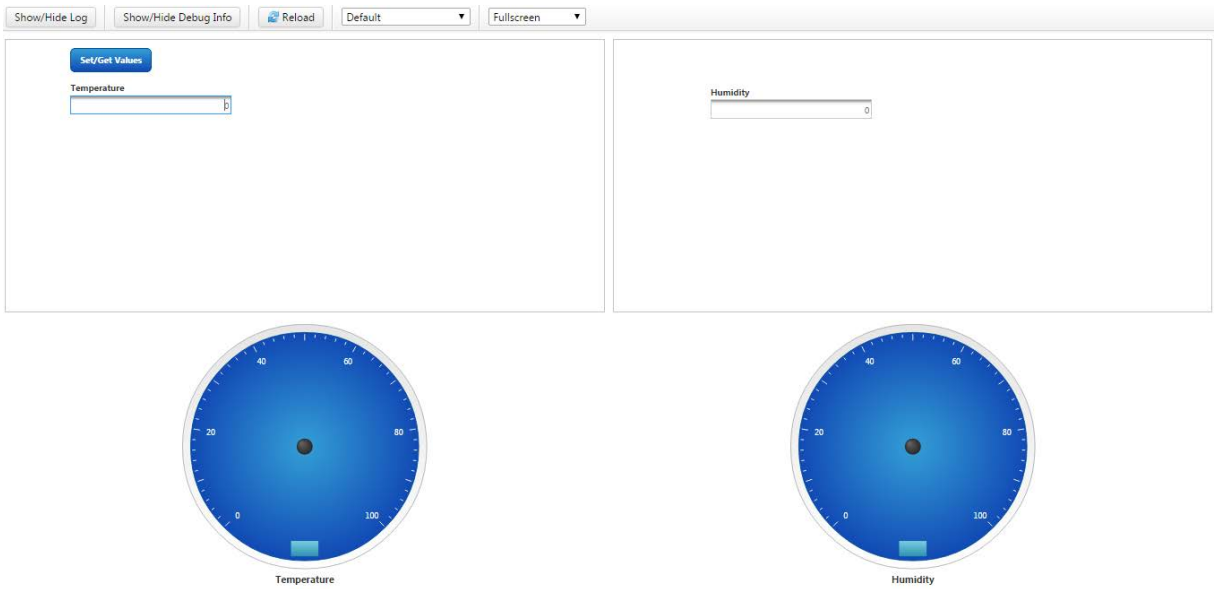


- *If you have set the values correctly, you can see the new values under the **Value** column as shown below:*

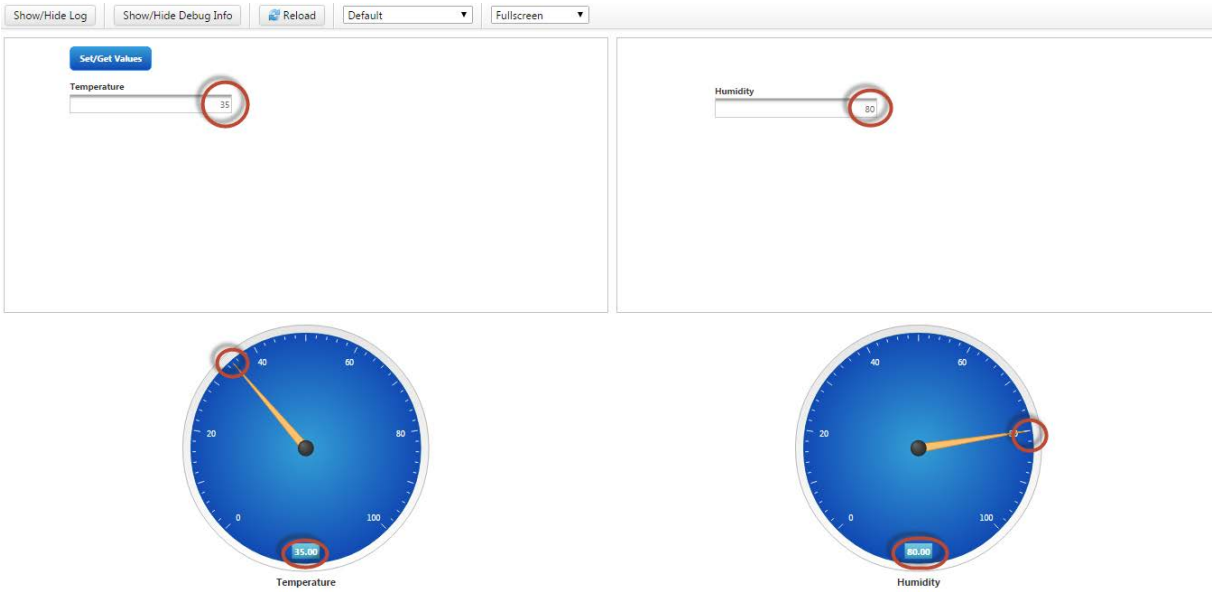


2) Now, you need to check if these new values have been automatically set in the mashup.

- *You can check this by pressing the **Reload** button from the mashup as shown below:*



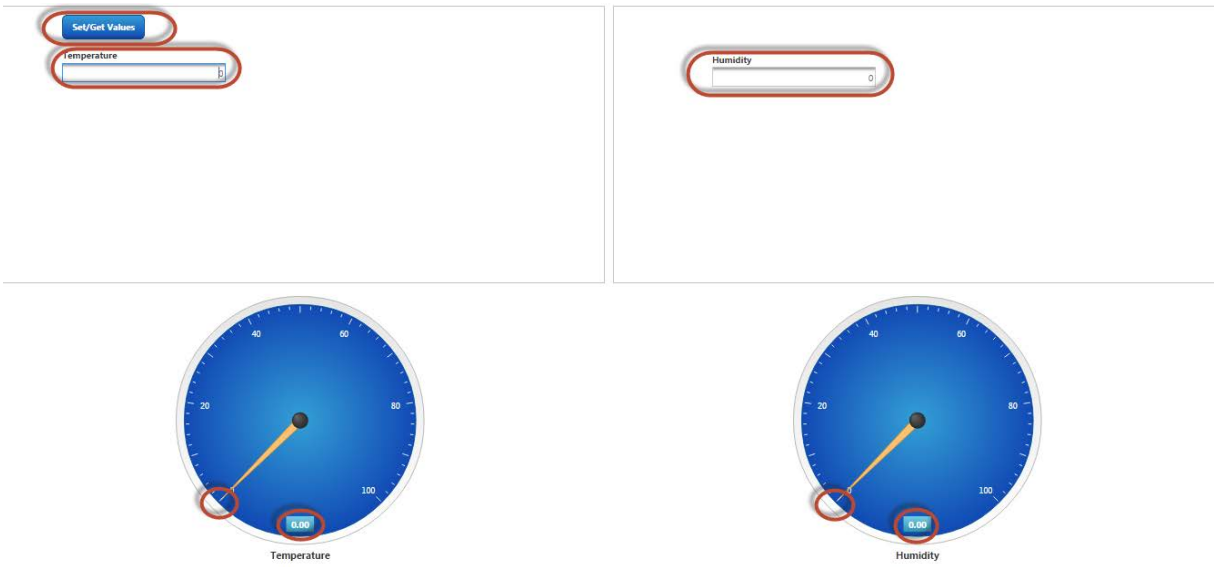
- *After you have pressed the **Reload** button, the mashup looks as shown below:*



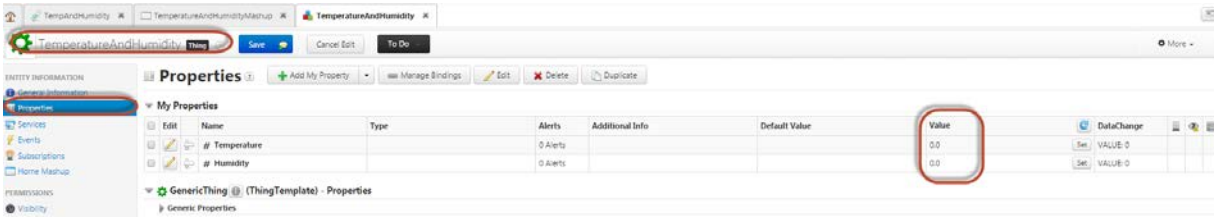
3) **Check that you can change the output of the gauges by entering a value inside the Temperature and Humidity boxes, and that these new values are automatically set in the thing's properties according to what you enter inside the mashup**:

*a) Type a value inside the **Temperature** box(for example 0);*
*b) Type a value inside the **Humidity** box(for example 0);*
*c) Press the **Set/Get Values** button;*

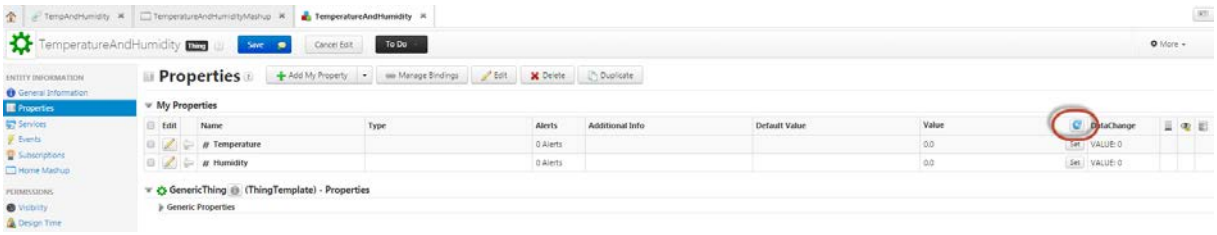
- *The mashup now looks as shown below:*



- *Next, you want to check if the values from the thing's properties were automatically changed to the values set inside the mashup (you have Value = 0 for Temperature and Value = 0 for Humidity). The screen on the properties from the TemperatureAndHumidity thing looks like this:*



- *If, by any chance the values are the same as before (Temperature = 35 and Humidity = 80), press the **Refresh** button from the Value column:*



Summary:

In this documentation you have covered some of the most common features of the ThingWorx platform, such as composer, tags, things, properties, services, mashup's and widgets. You created a tag that you used in order to identify all your mashup's components, than you have created a thing, defined some properties for it and then finally displayed those in a basic form in a mashup.