INTERNET OF THINGS WITH ARDUINO AND DELPHI

BY BOIAN MITOV

In the previous articles, you learned how to program Arduino using Visuino, and how to communicate with it using USB simulated serial port from your Delphi code. This opens a lot of interesting possibilities for collecting and processing live data, but the direct USB connection imposes some limitations. What if you want to collect data from many sensors spread over large area? Or what if you want to communicate with remote sensors over Internet? The basic Arduino UNO does not have built in network adapter, but there is Ethernet shield available for it. In addition many of the more advanced Arduino boards and their clones come with WiFi or wired Ethernet built in.

There are also cheap and simple ESP8266 WiFi modules that can be connected to the Arduino, so networking Arduinos is routinely done.

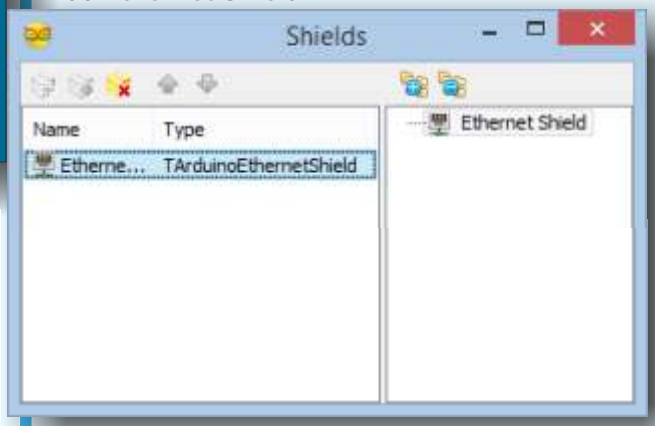
In this article you will learn how to setup Arduino to use Ethernet Shield, how to program it with Visuino, and how to connect to it from a Delphi application over the local network or Internet.

Before you start, you will need to install Ethernet Shield on the Arduino. This is fairly easy. Just snap it on top of the board as shown in the picture.

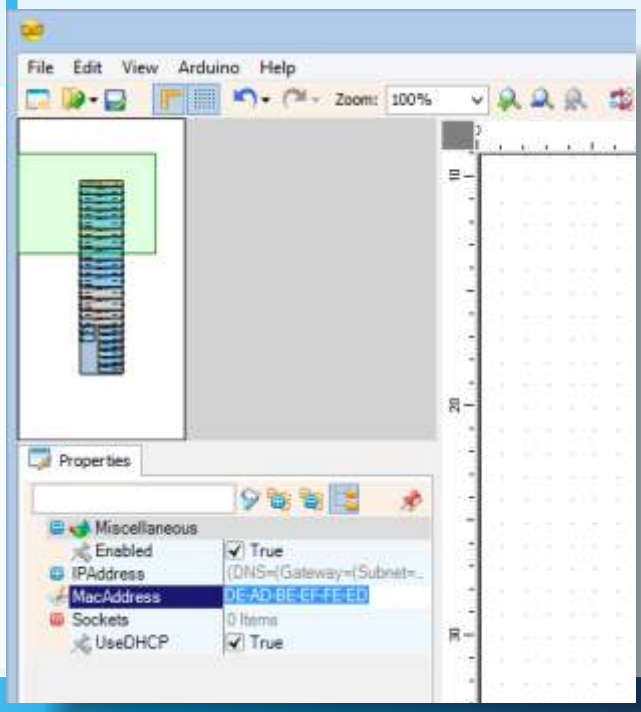
You also will need to install CommunicationLab, PlotLab and InstrumentLab from Mitov Software. CommunicationLab is not officially released yet, but prerelease builds are available on request. You can also easily modify the examples in this article not to use PlotLab or InstrumentLab.

First you will create a simple Arduino server. Start Visuino. Click on the Down arrow button in the top right corner of the Arduino component, and from the menu select "Add Shields..." :

Add Ethernet Shield:

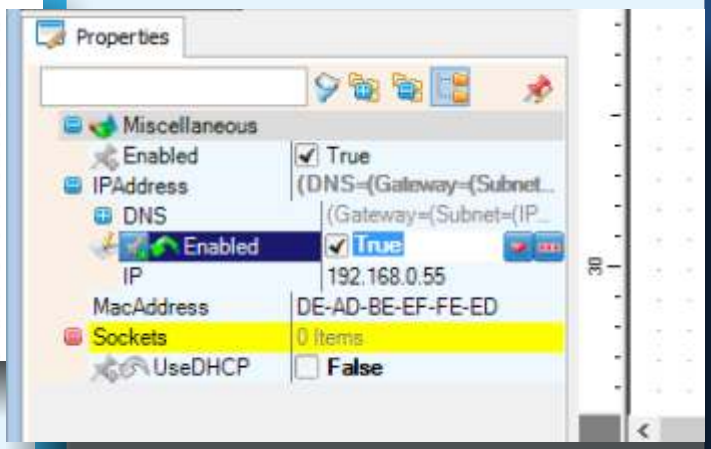


Next you need to specify the MAC address for the shield. You can use a MAC address generator, or one of the MAC addresses from the Arduino tutorials. Here I use DE-AD-BE-EF-FE-ED:

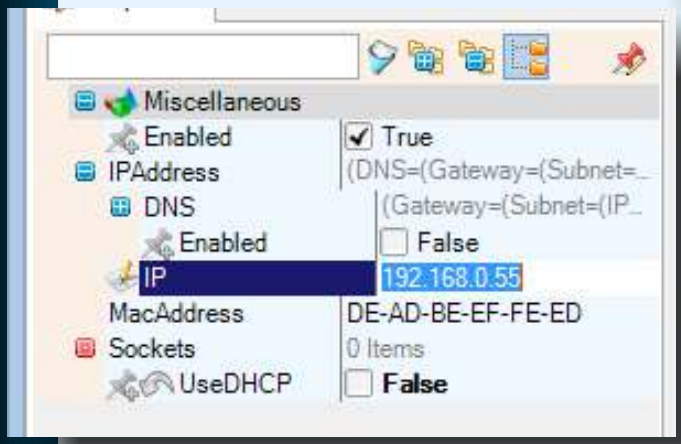


Set the "UseDHCP" to false, so Arduino will work with a fixed IP address:

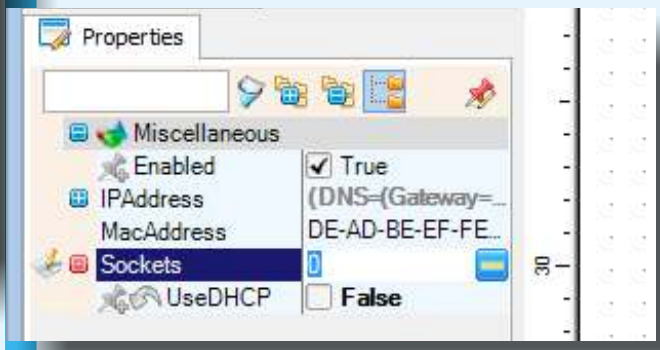
And set the Enabled property of the IPAddress to True so the IP address will be used when Arduino starts:



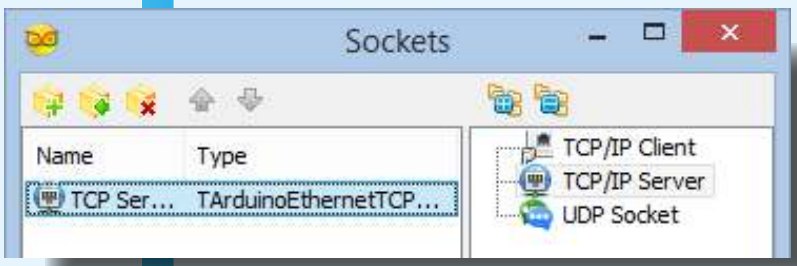
Set the IP address for the Arduino as example 192.168.0.55:



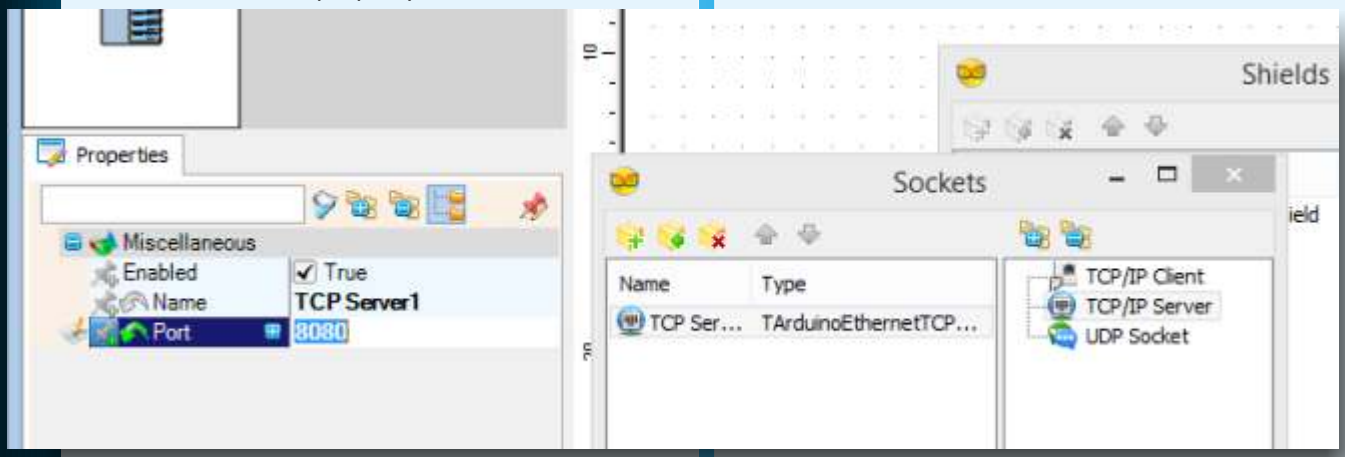
Once the Ethernet Shield is configured, you can add one or more TCP/IP Client, TCP/IP Server, or UDP sockets to it. Click on the "..." button after the Sockets to add a socket:



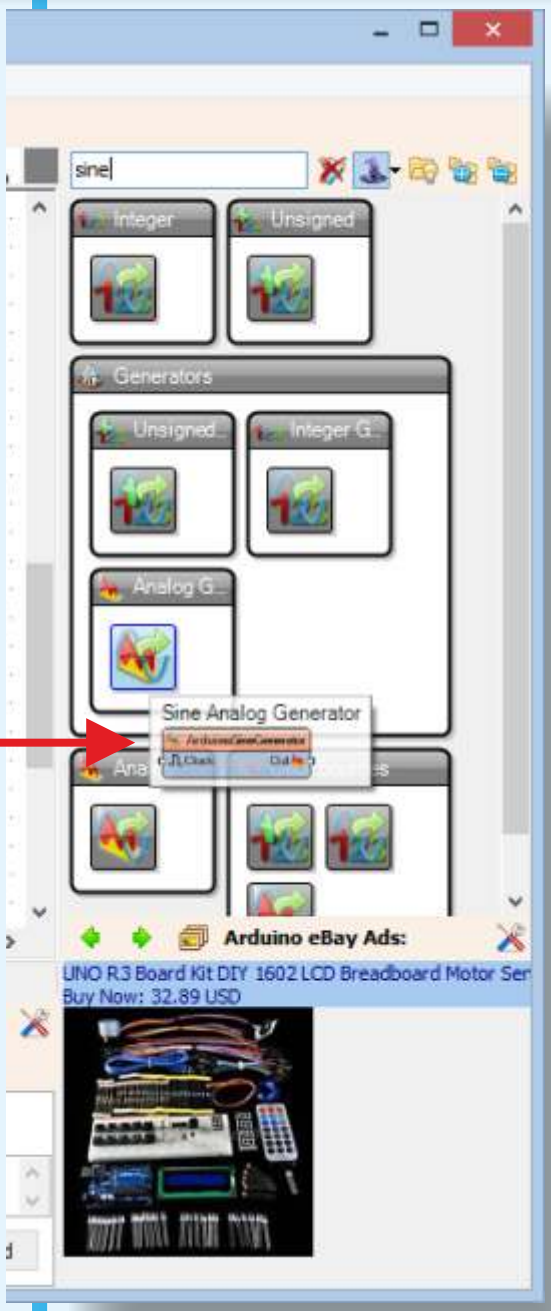
Add TCP/IP Server socket:



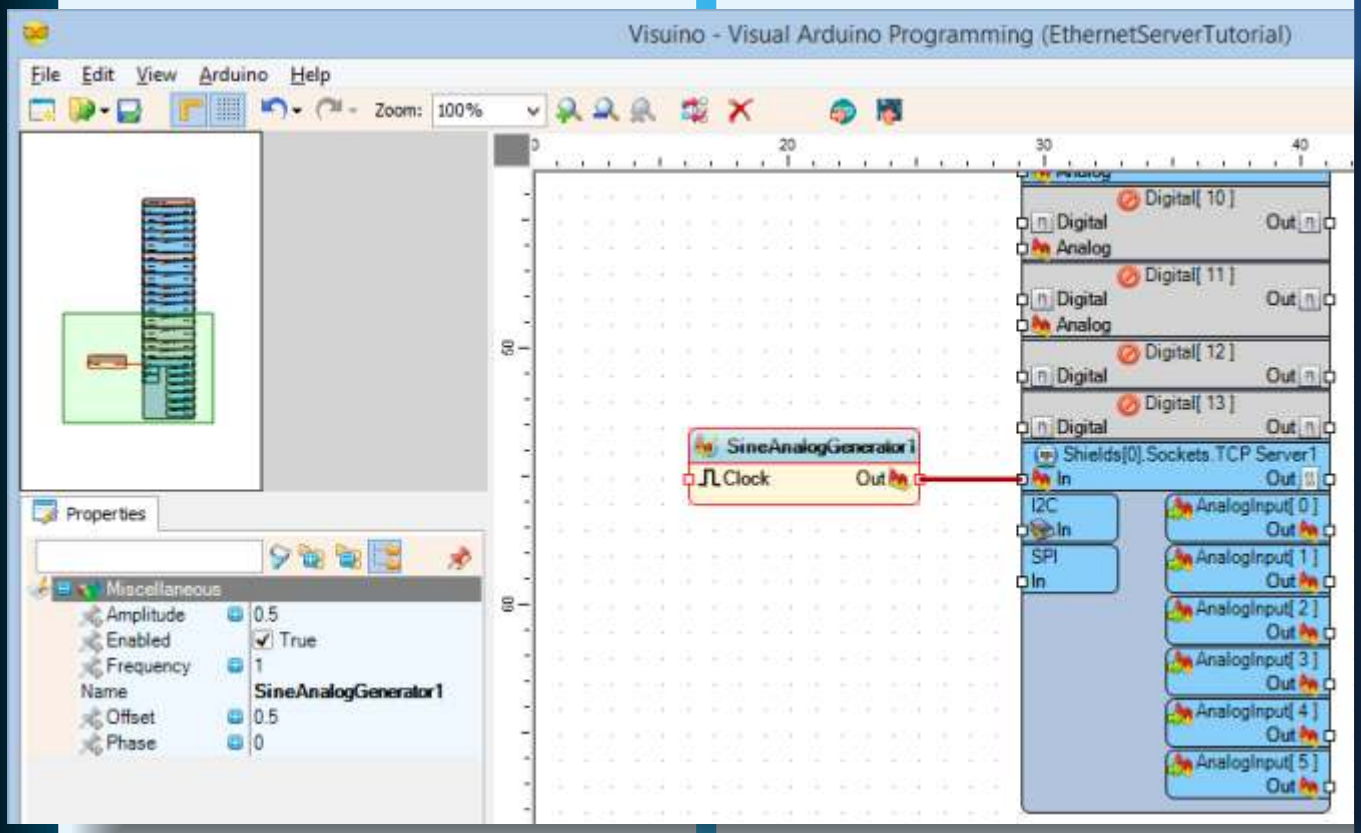
Set the Socket Port property to 8080:



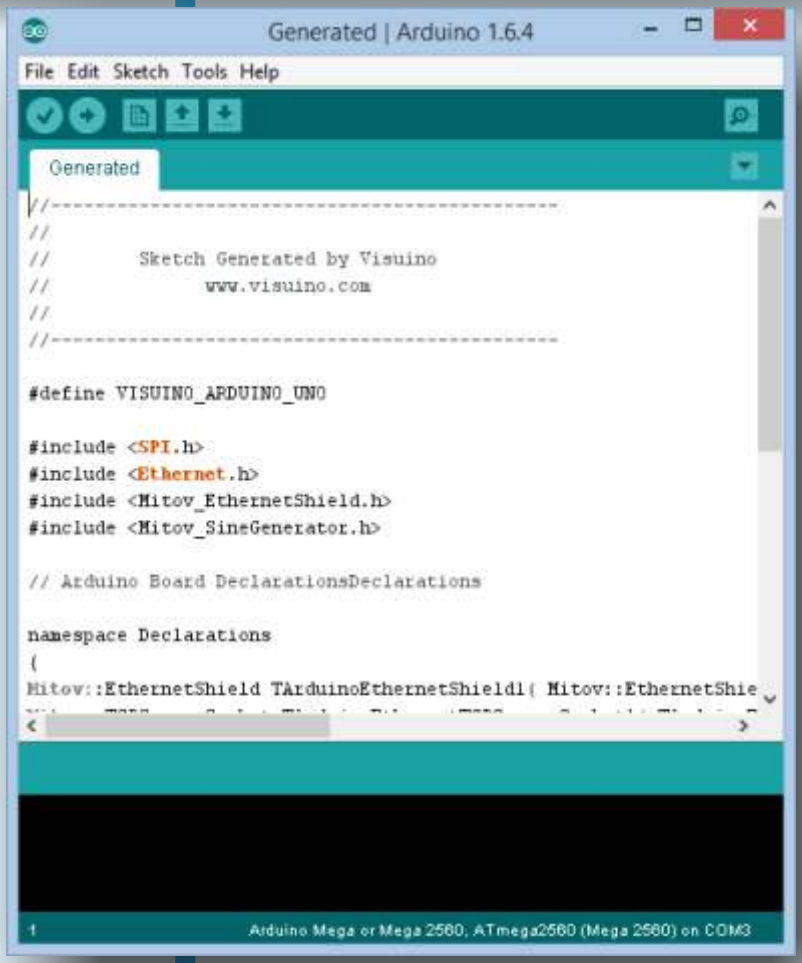
To generate some test data from Arduino, you can use a Sine Generator as shown here, or you can use any other source of Analog data, or one of the Analog channels:



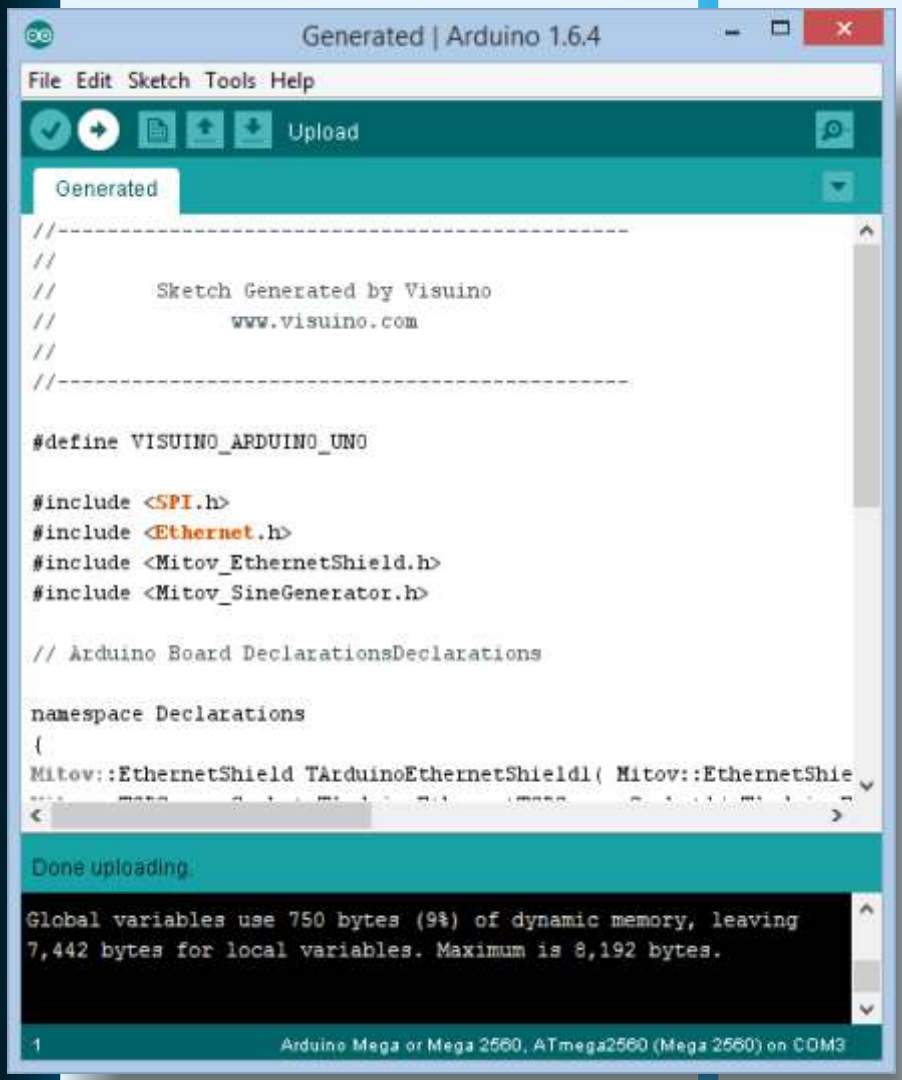
Connect the data source to the Input Pin of the Server Socket:



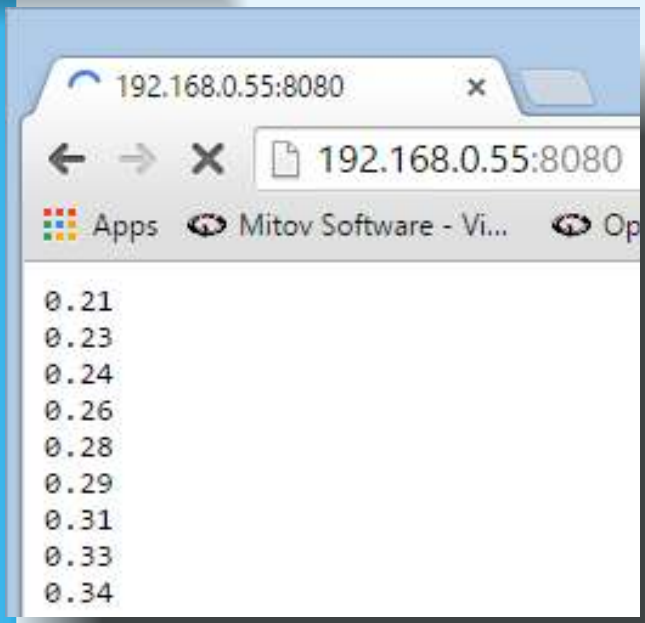
Press F9 to generate the Arduino Sketch and open the Arduino IDE:



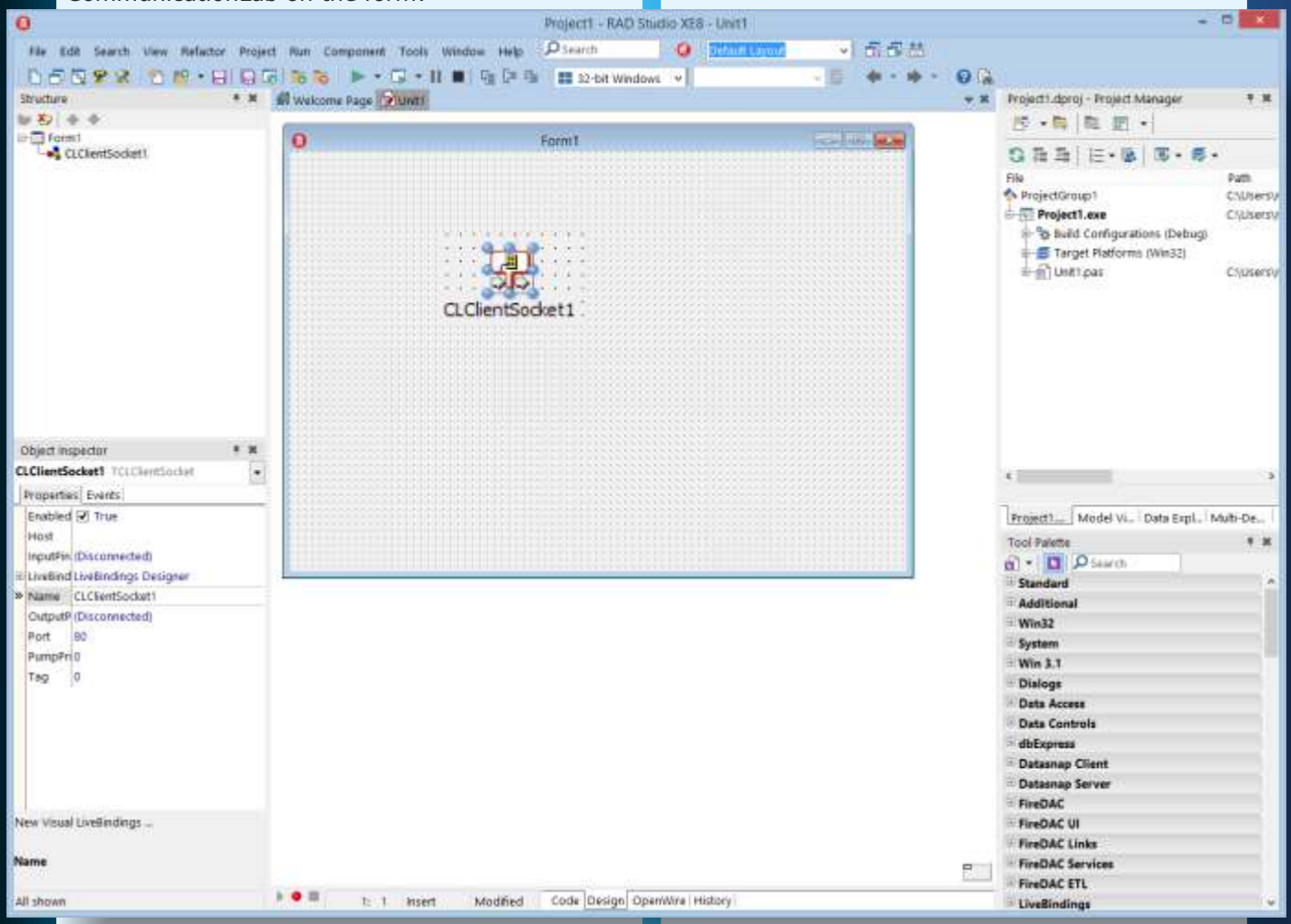
Click on the Upload button to compile and upload the sketch:



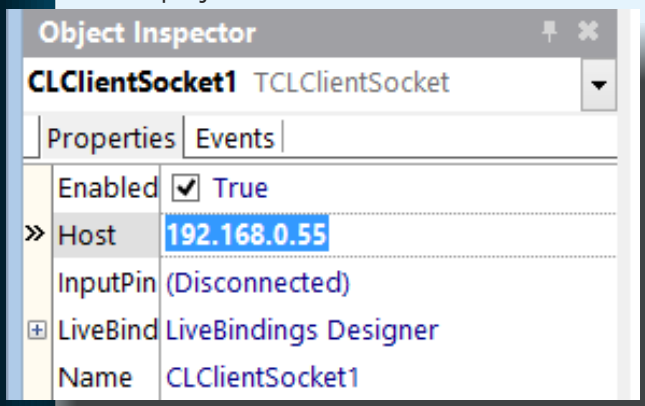
The simplest way to see if the Arduino project is working is to open a web browser and enter the Arduino IP address, and socket number – 192.168.0.55:8080 . You should see the data appearing in the browser, in this case Chrome:



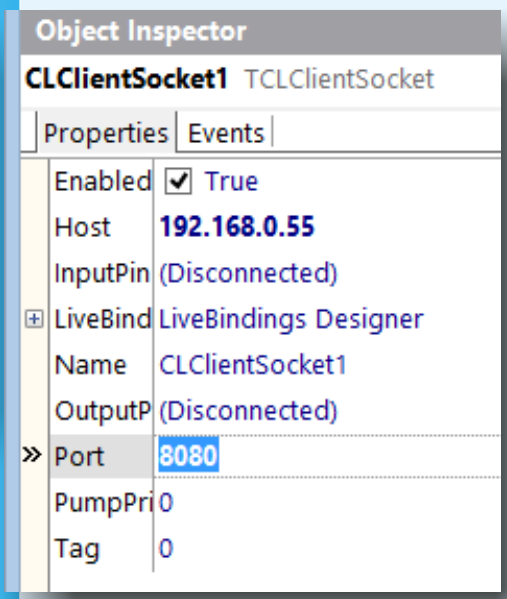
Next its time to receive the data in Delphi. Start RAD Studio, create a VCL Form project, and drop a TCLClientSocket from CommunicationLab on the form:



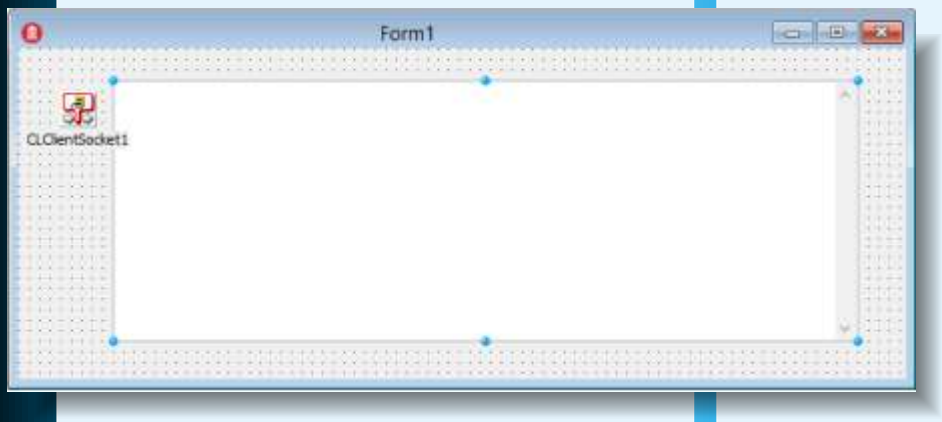
Set the IP Address to the same used in the Visuino project 192.168.0.55:



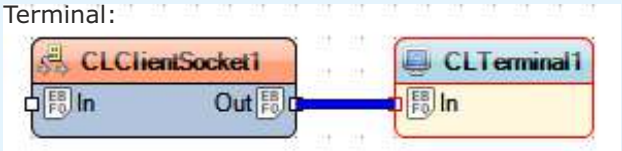
Set the Port to 8080:



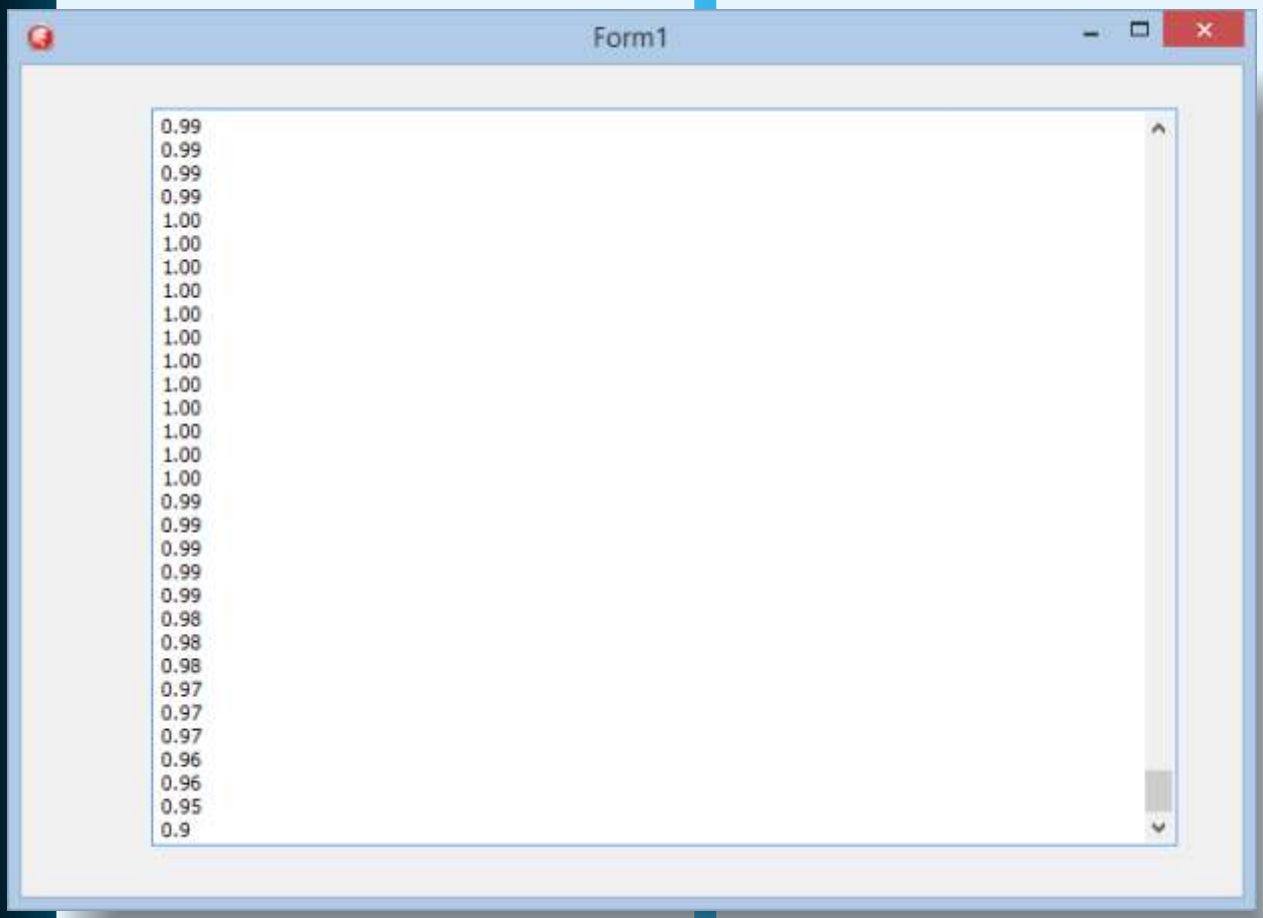
Drop TCLTerminal on the form:



Switch to the "**OpenWire**" tab, and connect the Output Pin of the Socket to the Input Pin of the Terminal:



Compile and run the application. You will see the data arriving in the terminal:



If you need to access the data in your code, the Socket component has OnReceive event:

Project1 - RAD Studio XE8 - Unit1

File Edit Search View Refactor Project Run Component Tools Window Help

Structure

- Classes
  - TForm1(TForm)
    - Published
      - CLClientSocket1: TCLClientSocket
      - CLTerminal1: TCLTerminal
      - CLClientSocket1Receive(ASender
- Variables/Constants
- Uses

Object Inspector

CLClientSocket1 TCLClientSocket

Properties | Events

OnConnect	
OnDisconnect	
OnReceive	LClientSocket1Receive
OnSend	

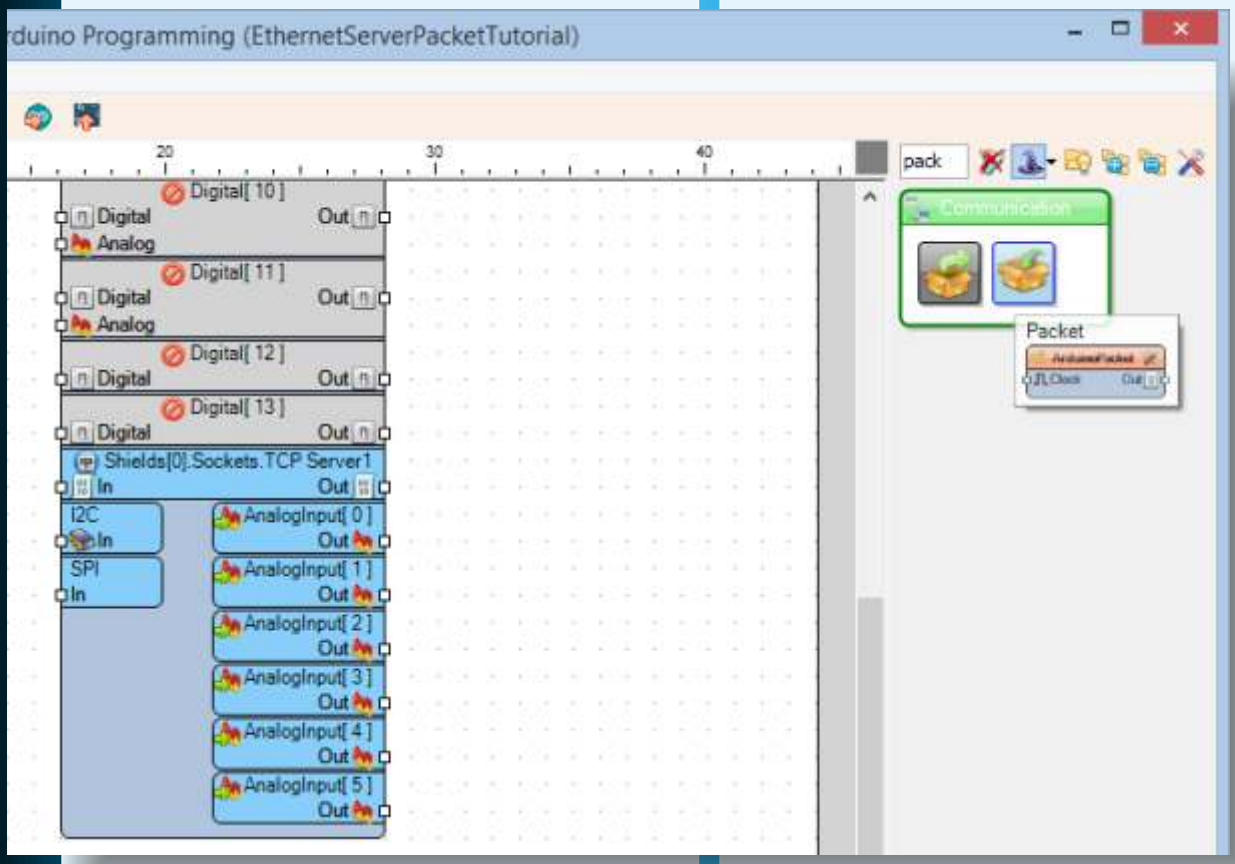
New Visual LiveBindings ...

OnReceive

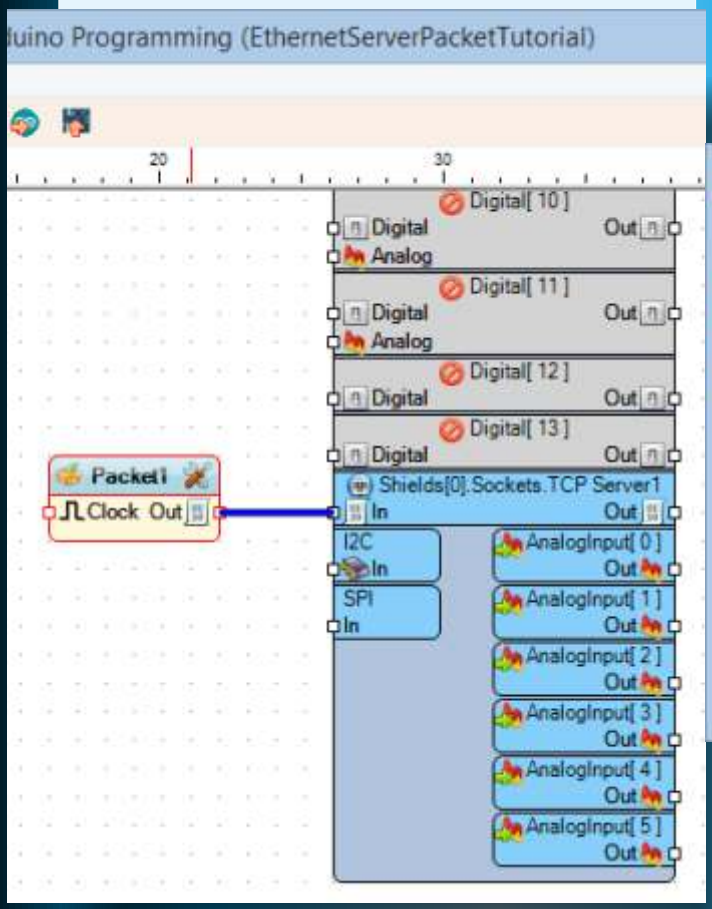
```
  end;

var
  Form1: TForm1;

implementation

{$R *.dfm}

procedure TForm1.CLClientSocket1Receive(ASender: TObject;
  AData: ISLBlockBuffer);
begin

end;

end.
```

Now you can receive data from Arduino over the network, however the data arrives from a single sensor, and in text form. This makes it difficult to work with, and limits the data channels that we can get.
As shown in the previous article, Visuino and CommunicationLab have support for packet data. You can use it with sockets the same way you did with the serial port.
Start a new Visuino project, add and configure the shield and the socket as you did in the previous project.

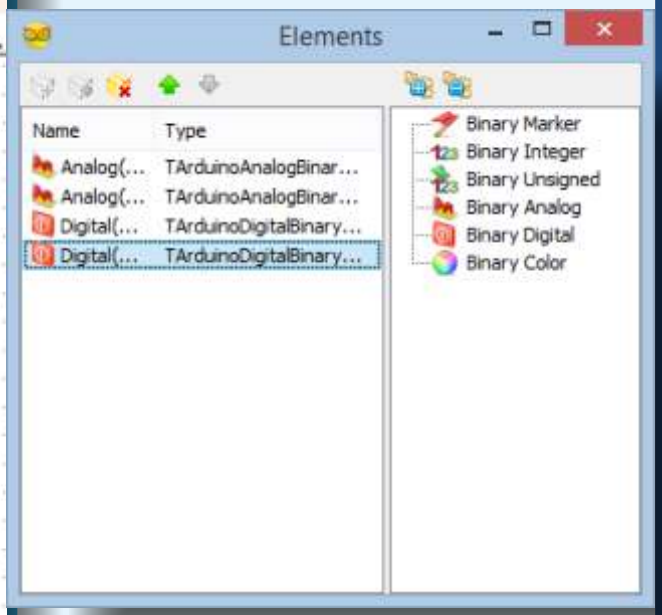
Next, add a Packet component:



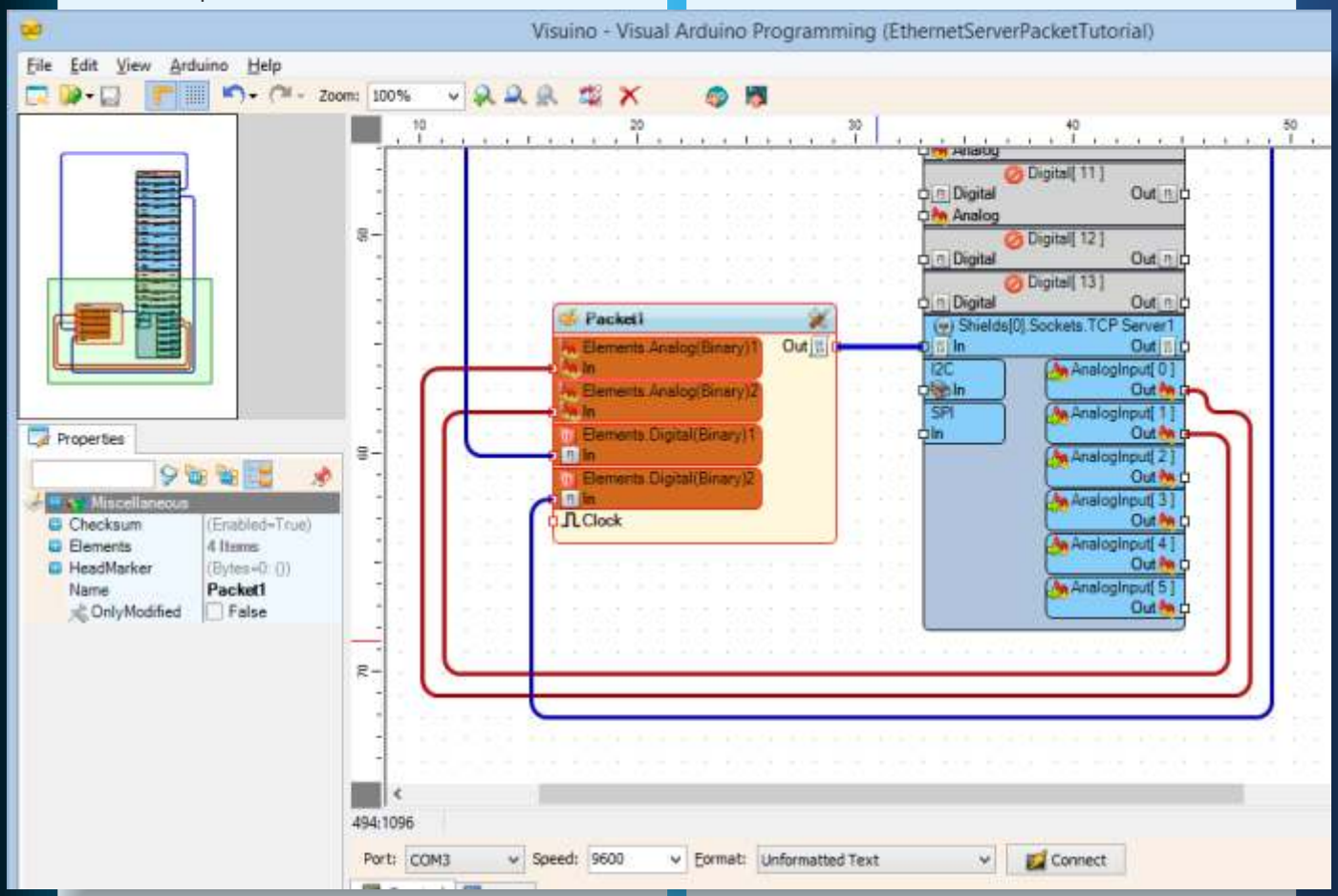
Connect the Output Pin of the Packet component to the Input Pin of the Socket:



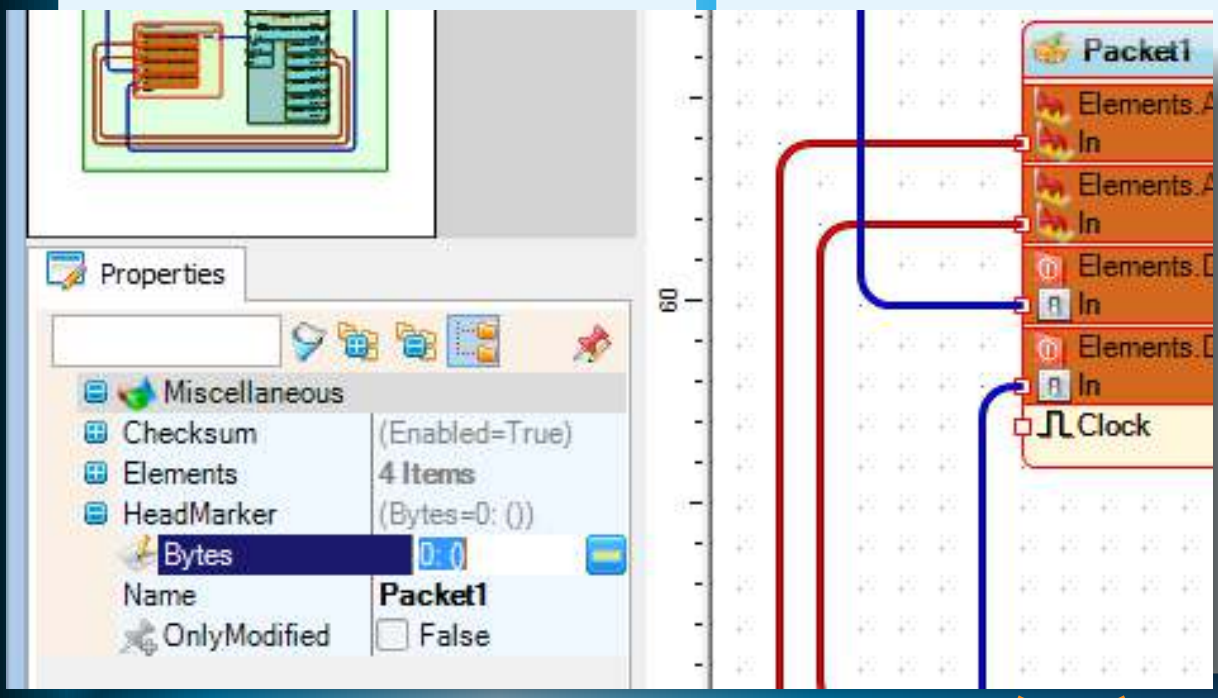
Double click on the packet component to open the elements editor. In the editor add 2 Analog and 2 Digital elements:



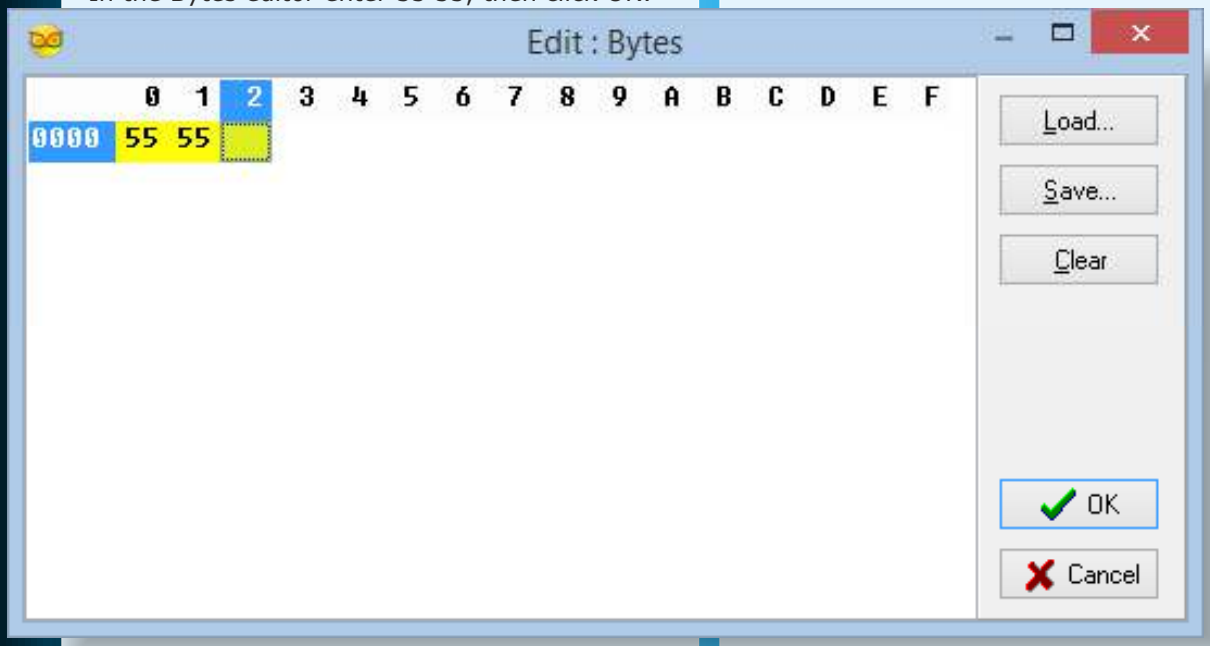
Connect the Input Pins of the Analog Channels to the Output Pins of "Analog Input Channel[0]" and "Analog Input Channel[1]", and the Input Pins of the Digital Channels to the Output Pins of Digital Channel 0 and 1 of the Arduino Board component:



Expand the HeadMarker and for the Bytes click on the "..." button:

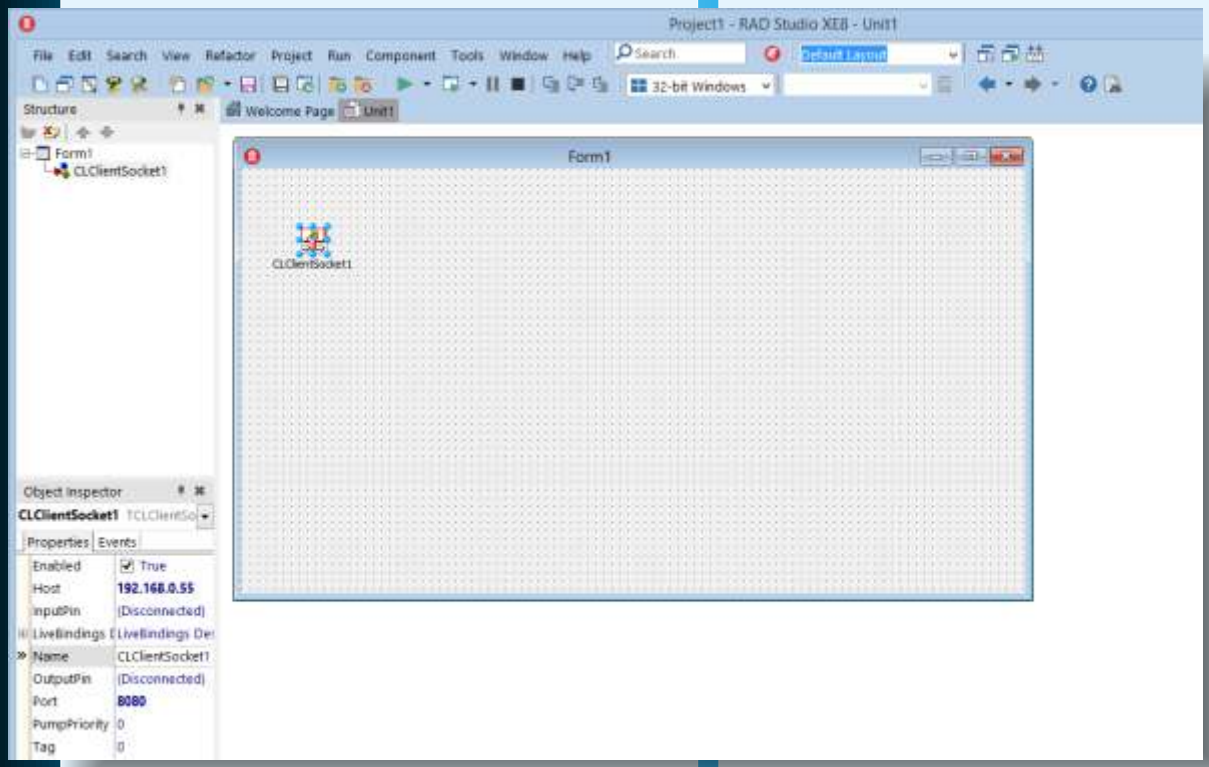


In the Bytes editor enter 55 55, then click OK:

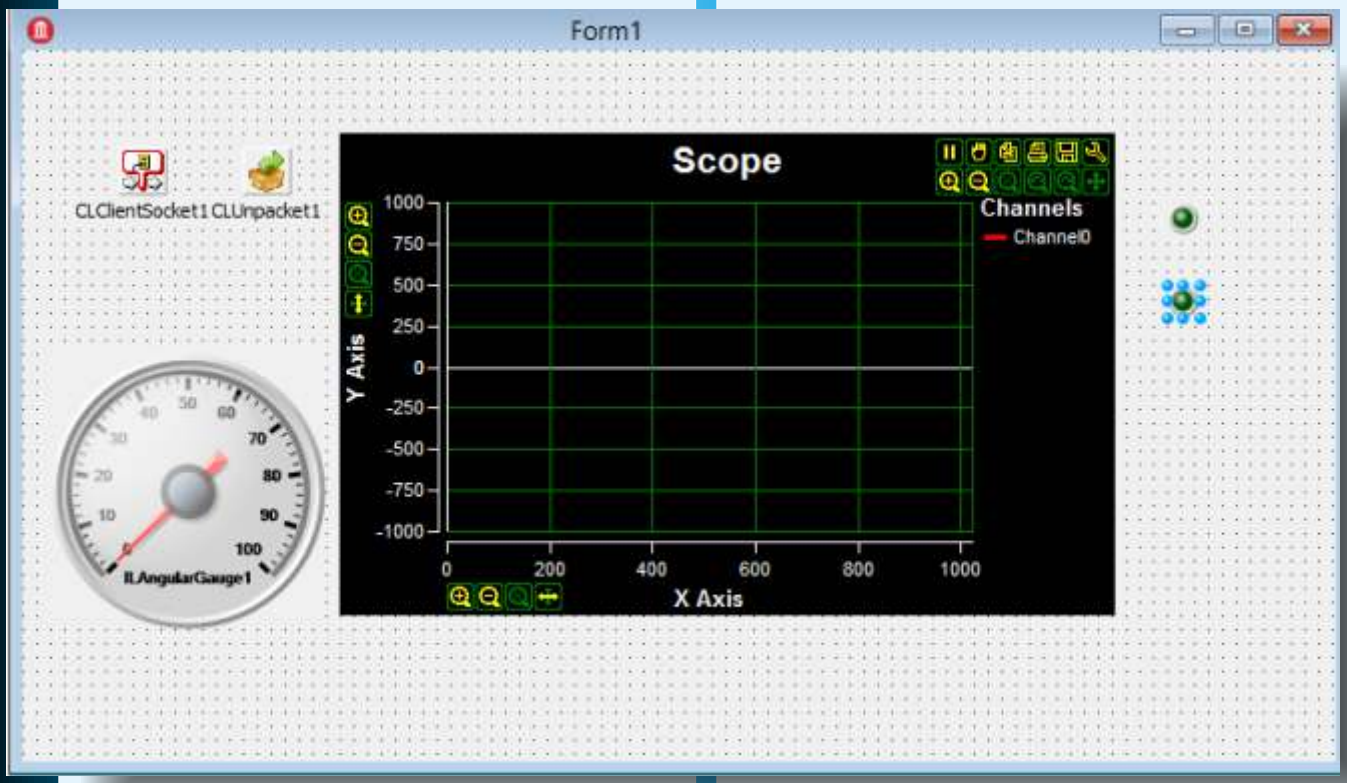


This will enure that the packet has unique header and its starting point can be identified in the data stream. Visuino uses special algorithm to ensure that the header marker will not appear in the packet itself.

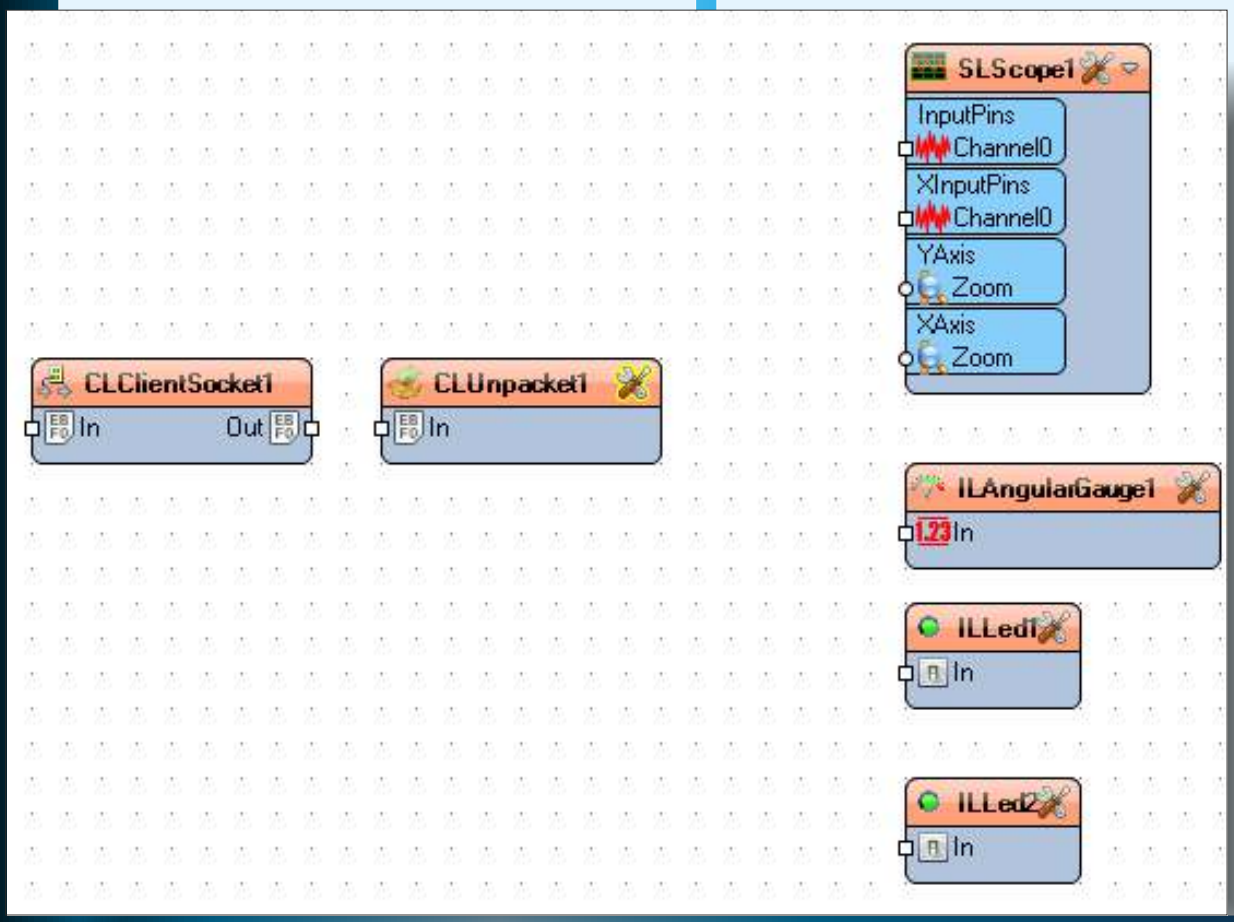
Press F9 to generate, then compile and upload the sketch in the Arduino IDE as you did in the previous project.
Now that the Arduino is ready, lets switch to Delphi.
Start a new VCL Form project, add the TCLClientSocket, and set the IP Address and Port as in the previous project:



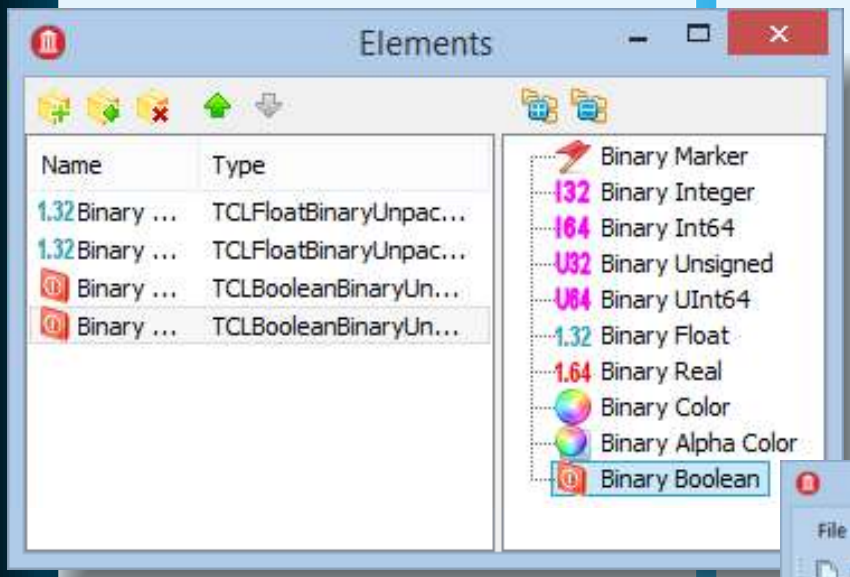
Next add a TCLUnpacket, TSLScope, TILAngularGauge and 2 TILLed components:

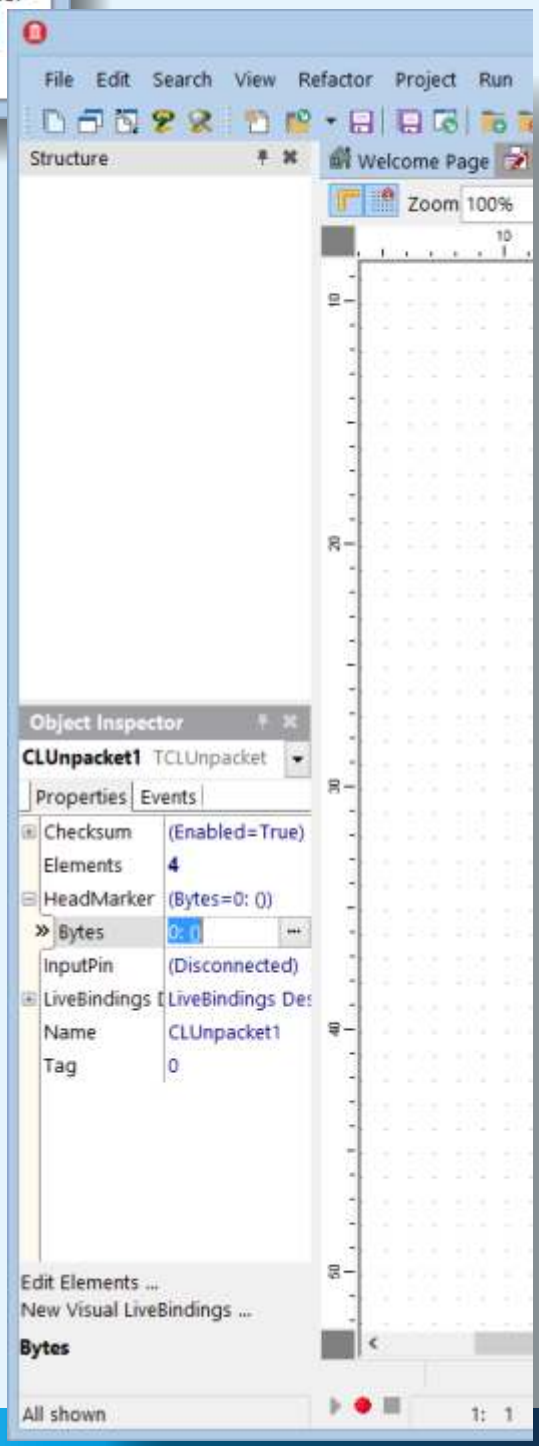


Switch to the OpenWire tab, and double click on the CLUnpacket1:



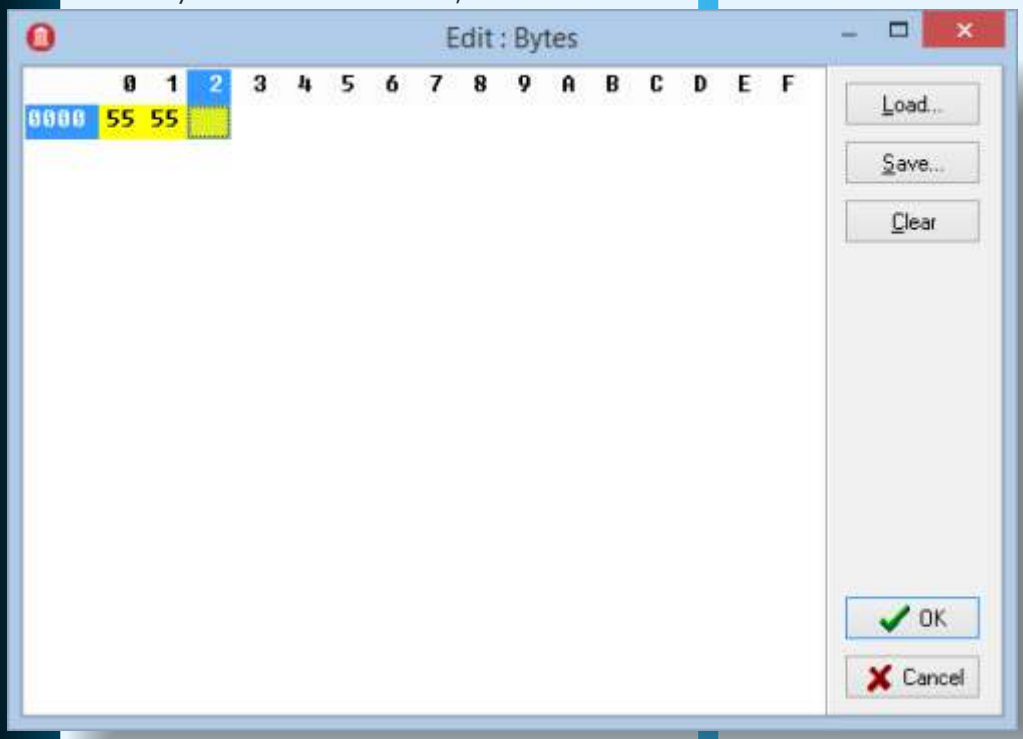
Add 2 Float and 2 Boolean channels:



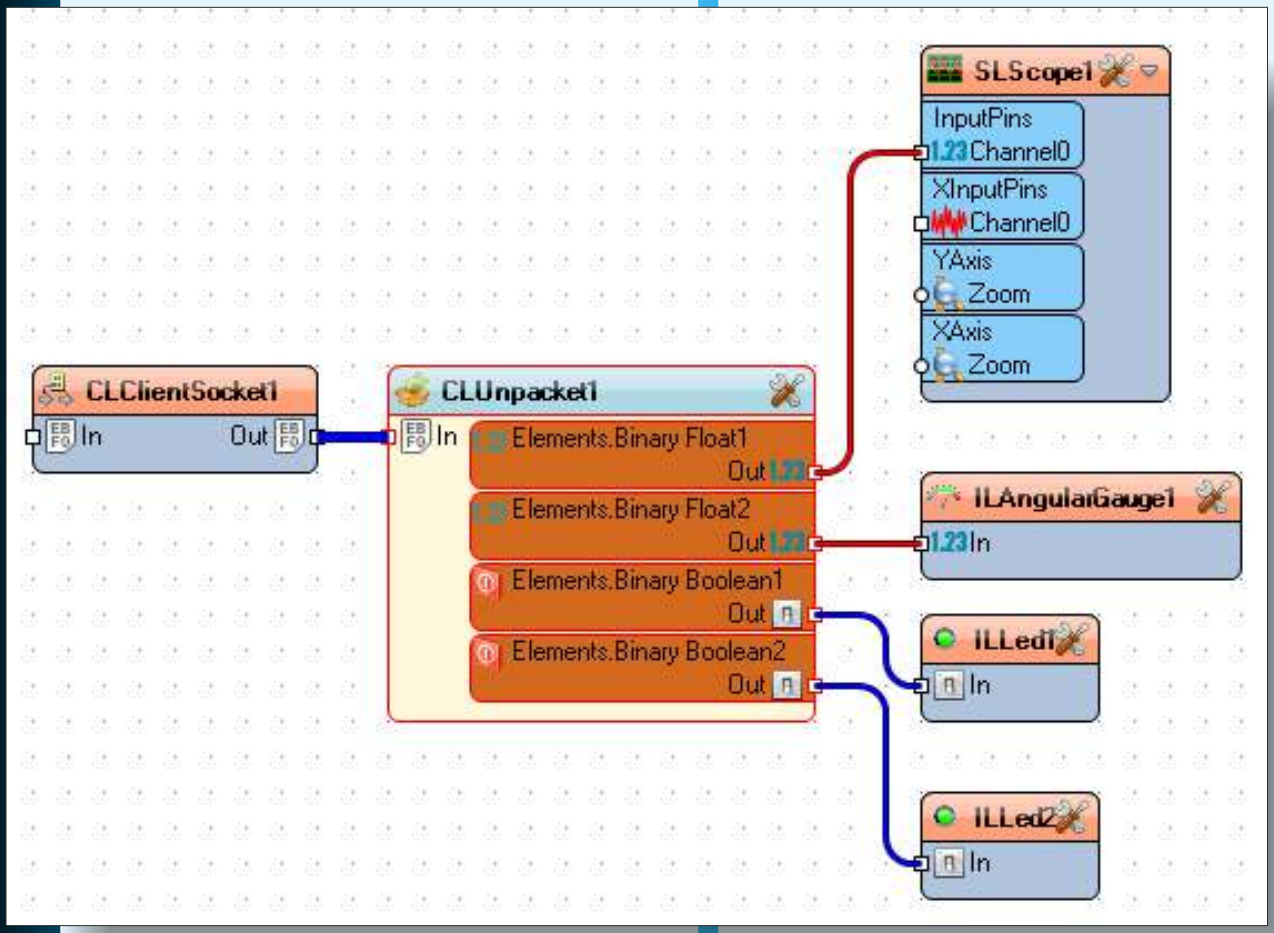


Expand the HeaderMarker and for the bytes click on the "..." (called elipsis)

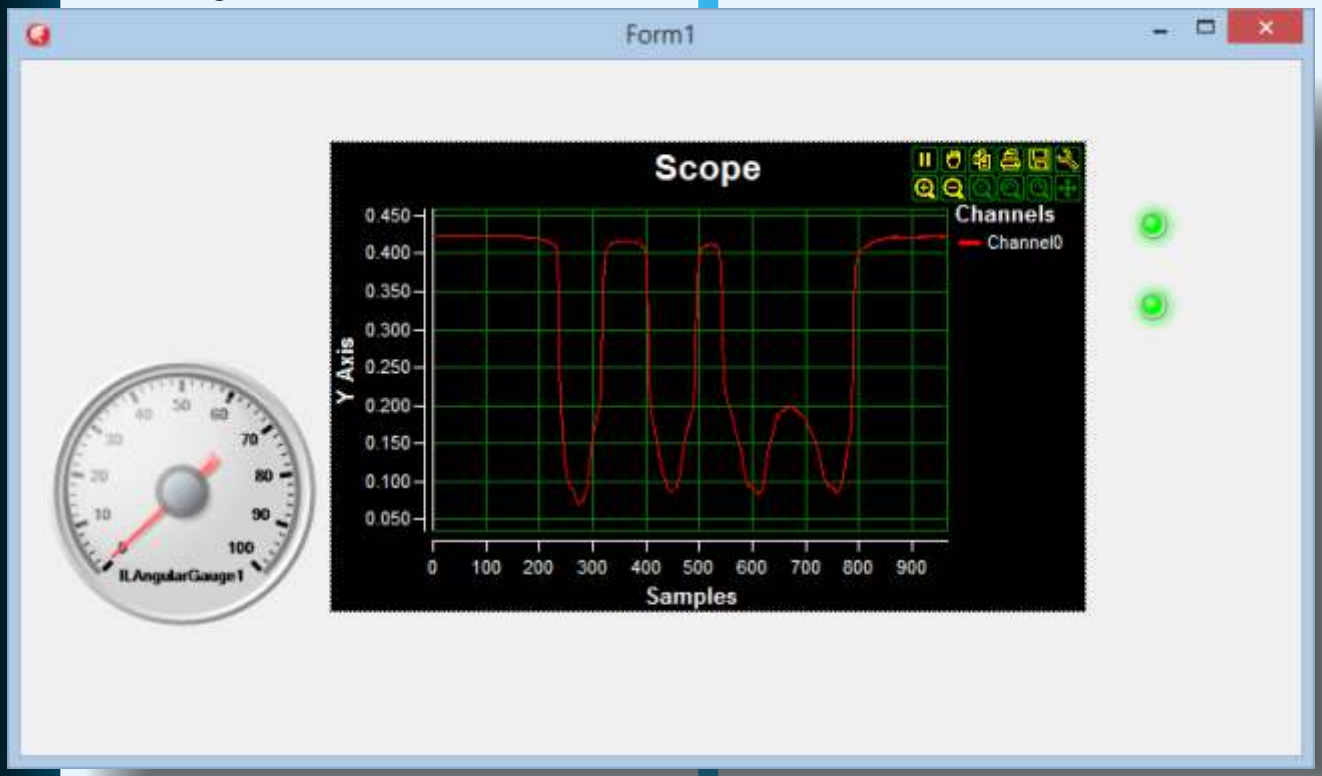
In the Bytes editor enter 55 55, then click OK:



Connect the components as shown in the picture:



Compile and run the application. You will see the data arriving from Arduino over the 4 channels:



When you need to access the data from code, you can use the TSLGenericRealValue as example to receive the Floating point data and process it in the OnProcessData event, as shown in one of the previous articles. There are also similar components for processing the Boolean data included in LogicLab.
The communication to Arduino is equally easy. In order to send the data from Delphi, use TCLPacket component and in Visuino use Unpacket component.

BLAISE PASCAL MAGAZINE subscribers that visit our PASCON - Event will receive a DVD with lots ofprograms, information and as a **VERY SPECIAL INCENTIVE** you will get an **ARDUINO-BOARD FOR FREE INCLUDING THE VISUINO SOFTWARE** from Boian Mitov to be able to compose and create your own software for the board

CONCLUSION
This article has given you enough information to start communicating with one or more Arduino devices over wired network, or Internet from your Delphi code. This is your first introduction to the exciting world of Internet of Things. In the following articles you will learn how to communicate with Arduino over WiFi, and how to make multiple Arduino boards to talk to each other.