Aalborg University Copenhagen
Medialogy MED4 2007

# Sensors Technology

## Installation of Teleo Intro board, and software utilization examples

Smilen Dimitrov

## Contents

# 1 Introduction

This document is continuation of the Sensors Technology study sheet "Software utilization of a sensor signal". It is a practical introduction to our the data acquisition board used in this course, Teleo Intro board by Making Things.

This is a board that can be characterized with:

- 4 analog inputs
- analog input voltage range supperted: 0V - 5V
- sampling rate of max 1 KHz (sampling period min 1 ms)
- sampling resolution of max 10 bits (max 1024 quantization levels).

as well as a driver and API that interfaces well with (among others) Adobe (Macromedia) Flash, or Cycling '74 Max/MSP software development environments.

# 2 Installation of Teleo driver

Although the Teleo as a product line is discontinued, its manufacturer Making Things still makess all the documentation and downloads available on the Internet. Downloads for the Teleo Intro board can be found on the Downloads page of the Making Things website:

http://www.makingthings.com/teleo/products/downloads/index.htm

From here, we can separately download packages for driver files for Max/MSP, Flash, as well as an SDK (which we will not use here).

For both platforms, the driver files consist of two parts – a USB driver, and interface objects / classes.

***The download package for Max/MSP*** is situated here:

http://www.makingthings.com/teleo/products/downloads/max_externals.htm

There are separate versions for Windows and Mac OSX. The Windows package contains five files/directories to place in the appropriate locations:

Teleo_externals
Teleo_help
TeleoMaxWXPLib.dll
teleo_objectlist.txt
FTDI_USB_Driver

Detailed installation instructions are available online, on the same page, as ReadMe's (for instance, for Windows it is http://www.makingthings.com/teleo/downloads/TeleoMaxV2/Win32/Teleo_Max_Win32/README.txt )

If this package is installed first, files should be copied, then Teleo should be connected to PC and powered – Windows will ask for a driver, and by pointing to FTDI_USB_Driver folder, it will eventually be installed. Then Max patches can be started.

You can use the FTDIUNIN.EXE program in the FTDI_USB_Driver to uninstall the USB driver manually.

***The download package for Macromedia Flash*** is situated here:

http://www.makingthings.com/teleo/products/downloads/flash_class_library.htm

There are several packages to download:

- Teleo Flash Class Library
- Teleo XML Server & USB Driver (R9052151.zip and TeleoXMLServerWin32.exe for Windows)
- Sample Applications & Simulations

This part of the installation has a bit more confusing help, but the important thing here is that first the USB driver should be installed, and tested with the XML Server program – hence, read first the corresponding ReadMe file (http://www.makingthings.com/teleo/downloads/teleo_flash_class_library/teleo_flash_server_win32_v_1_1/readme_xml_server_win32.txt ) , and install the USB driver.

Then, the Flash Class library can be installed (ReadMe http://www.makingthings.com/teleo/downloads/teleo_flash_class_library/readme_teleo_flash_v_1_0_1.txt ), after which we can try testing some Flash movies.

Note that we must run the XML Server program (for Windows, TeleoXMLServerWin32.exe) together with the Flash movie, so we obtain sensor values in it.

Note also that for Windows, the contents of R9052151.zip from Flash package and FTDI_USB_Driver from Max/MSP package are almost the same – that is, they contain installation files for the FTDI USB driver. They are however of different versions:

- R9052151.zip - February 4, 2004
- FTDI_USB_Driver - June 16, 2003

Keep in mind that once the USB driver is installed, say for Max, we do not need to install it again for Flash (Windows will not autodetect it anyways ☺) - only the installation of the Flash class library files will be necessary.

# 3 Preparing the Teleo

For now, we can simply follow the instruction in the Teleo Intro user guide:

http://www.makingthings.com/teleo/products/documentation/teleo_intro_user_guide/

to connect the sensors that are already provided in the package.

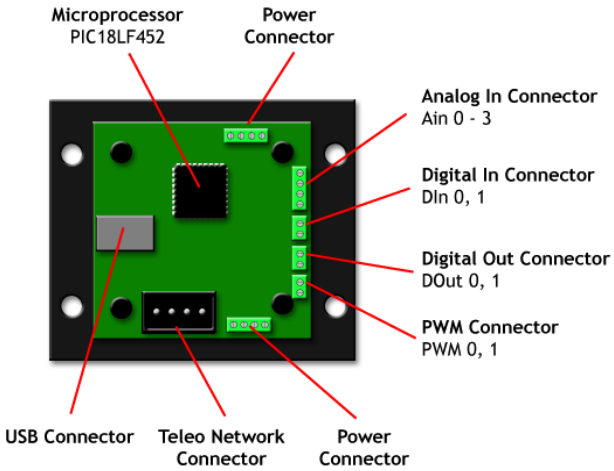



*Figure 1.Overview of the Teleo Intro data acquisition board ( Ref. [1])*

First we make sure the Teleo is connected to computer via USB:

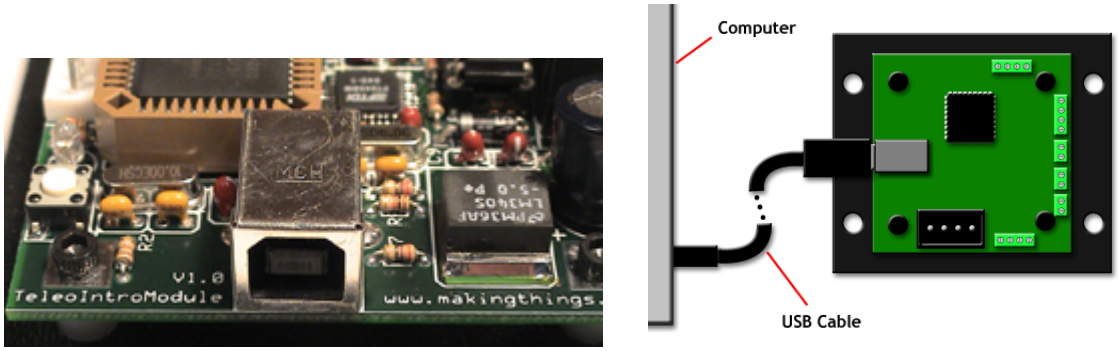



*Figure 2. USB connector of the Teleo, and connection diagram ( Ref. [1])*

Then we make sure power is connected to Teleo:

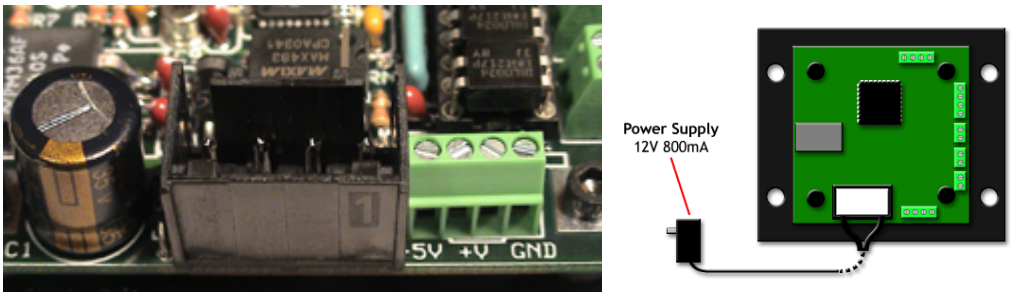



*Figure 3. Power connector of the Teleo, and connection diagram ( Ref. [1])*

By now we should have installed the driver, so powering the device will not raise the Found New Hardware wizard.

We should be able to identify the analog input connectors – and the ground (GND) and +5 V connectors next to them:

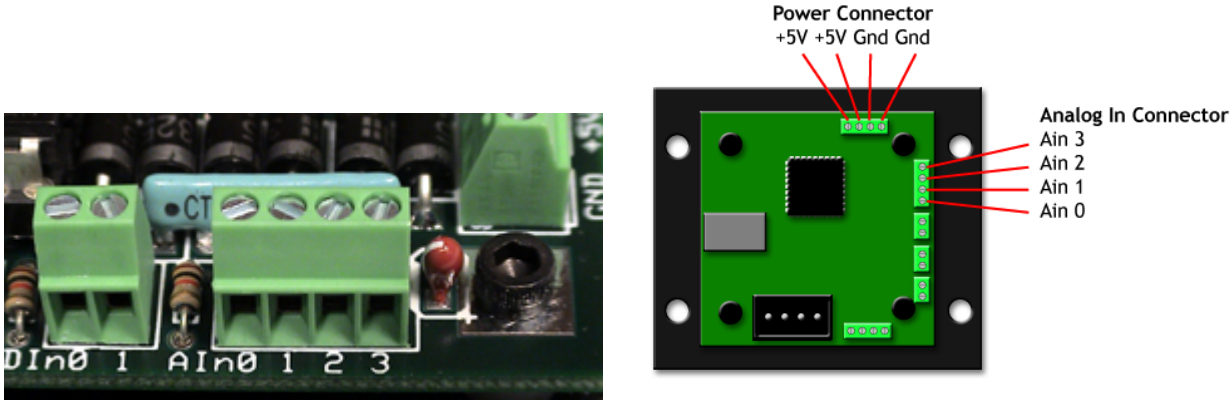



*Figure 4. Analog input connectors of the Teleo, and ouline diagram ( Ref. [1])*

We will use the Analog Input (AIN) and ground (Gnd) connectors to connect sensor circuit outputs to the Teleo; in addition, we can use the +5V pin of the teleo as a power supply.

The package comes with a photoresistive sensor circuit, and the user guide contains the following construction diagram:

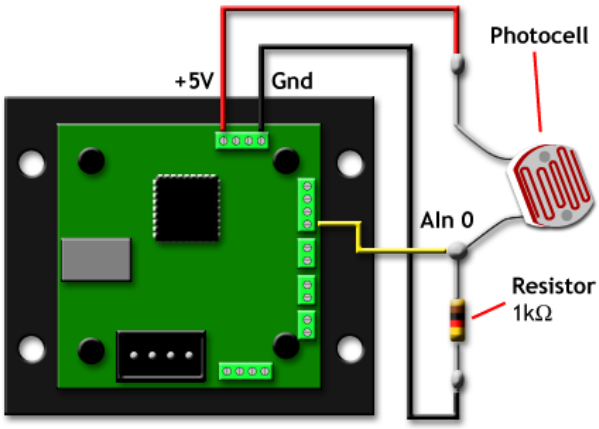



*Figure 5. Connecting a photocell sensor circuit to a Teleo ( Ref. [1])*

The wire coloring on the provided photocell should be as on the image – black goes to ground, red to +5 V connector, and yellow to any analog input (say AIN0). There is a screwdriver in the package as well – so sensor wires can be easily connected to the board. And that is all there is to it in this case – we now have a light sensor that we can use in our application!

There is also a push button available in the package, and the user guide contains the following construction diagram:

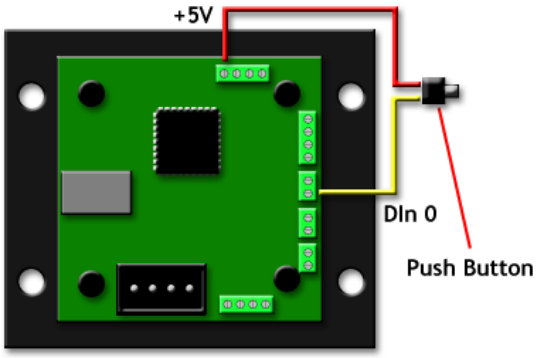



*Figure 6. Connecting a photocell sensor circuit to a Teleo ( Ref. [1])*

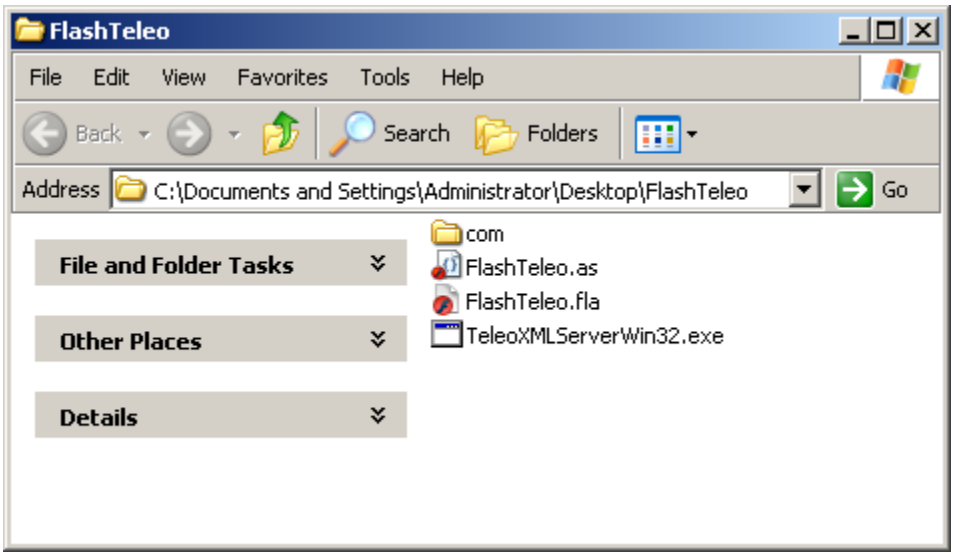
The wire coloring on the provided photocell should be as on the image – red goes to to +5 V connector, and yellow to any analog input (say AIN0). Obviously, the push button is digital in nature, so it will only provide as with two variable values once we start reading it – max for pressed and 0 for not pressed.

We can now look at the software utilization examples, assuming we have a Teleo with the sensors ready.

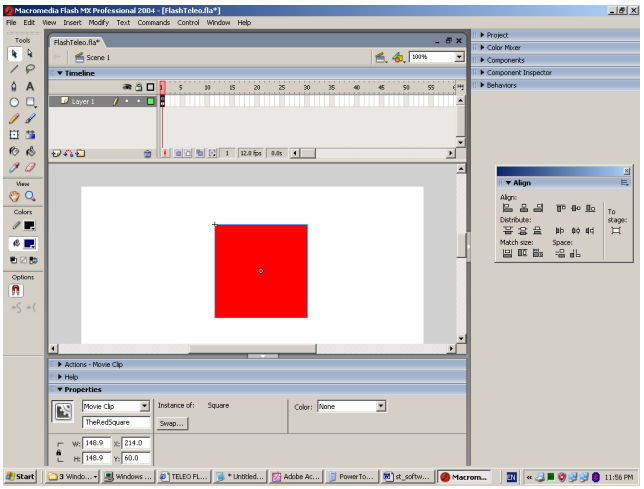
# 4 Example program Flash

Before starting to develop with Flash, make sure you have downloaded the ZIP Archive teleo_v_1_0.zip from the Teleo Flash Class Library. This Zip file contains a folder called com, which contains all needed action script class files.

Then, make a new folder for the Flash file somewhere (say FlashTeleo). Create a new Flash file (say FlashTeleo.fla), and make it #include an external actionscript file, which will also be saved in the same folder (say FlashTeleo.as). Finally, copy the com folder from teleo_v_1_0.zip into the chose folder. Although it is not necessary, you may want to copy the XML server there as well, for easier start up of the Flash file. The situation should look like this:




The example file provided relies on a red square defined as a movie clip in the Flash IDE, called TheRedSquare:



We start coding for Flash by importing the classes which will interface to the Teleo, and which are situated in the com folder that we previously copied:

```
import com.makingthings.*;
```

then we declare the variable which will contain our sensor value

```
var ain : TIntroAin = new TIntroAin(1);
```

In the declaration, we specify which analog input we reference through the argument of `new TintroAin`. Sampling rate and resolution can also be specified – consult the docs.

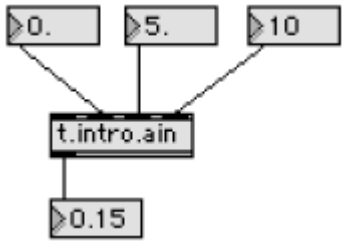
We do any initialization necessary the standard way in Flash – in the _root.onLoad function. Our 'endless' loop is in the _root.onEnterFrame Flash function – which repeats each time before a frame is rendered – at the Flash movie defined frame rate. Then, in Flash, the sensor utilization is easy – in our 'endless' loop, we simply read the sensor value through calling the getValue method of the ain object variable and storing the result in a numeric variable (here `NumberFromAnalogInput_Zero`) – we expect numeric integer values, determined by the sampling resolution (i.e. 10 bits resolution – integer values from 0 to 1023). Then, we simply apply it to a graphic property of TheRedSquare movie clip object – here for instance, we map it to the ._x property, which defines the x position – and thereby each frame the square is rendered at a new position – so we experience displacement animation according to the sensor signal. The code of this 'endless' loop is simply made of two commands:

```
_root.onEnterFrame = function()
{
      var NumberFromAnalogInput_Zero = ain.getValue();

      _root.TheRedSquare._x = TheRedSquare_SavedX +
NumberFromAnalogInput_Zero;

}
```

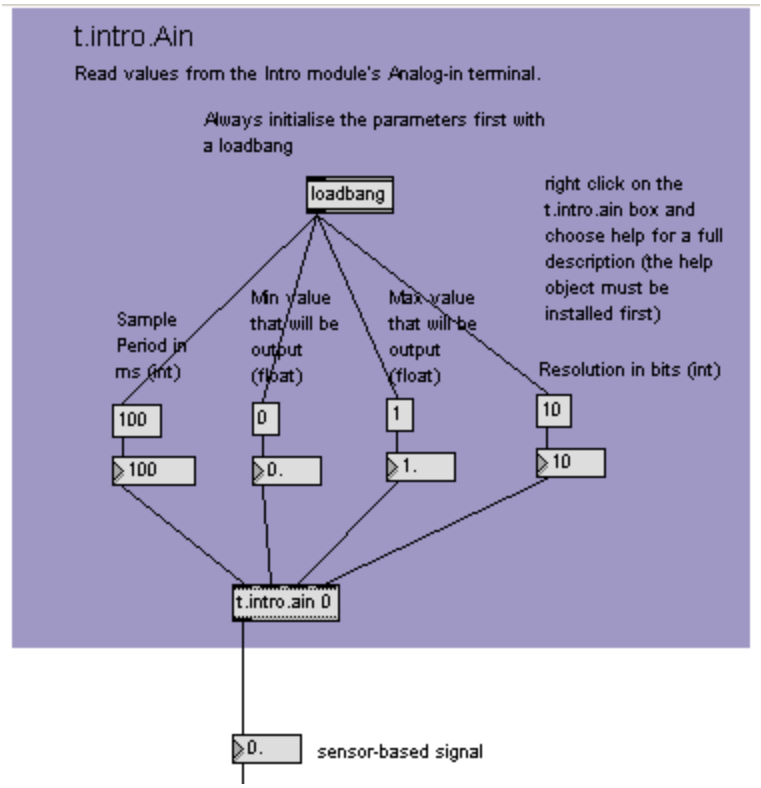
Again, the XML Server must run together with the Flash file, to make the sensor values available.

# 5 Example program Max/MSP

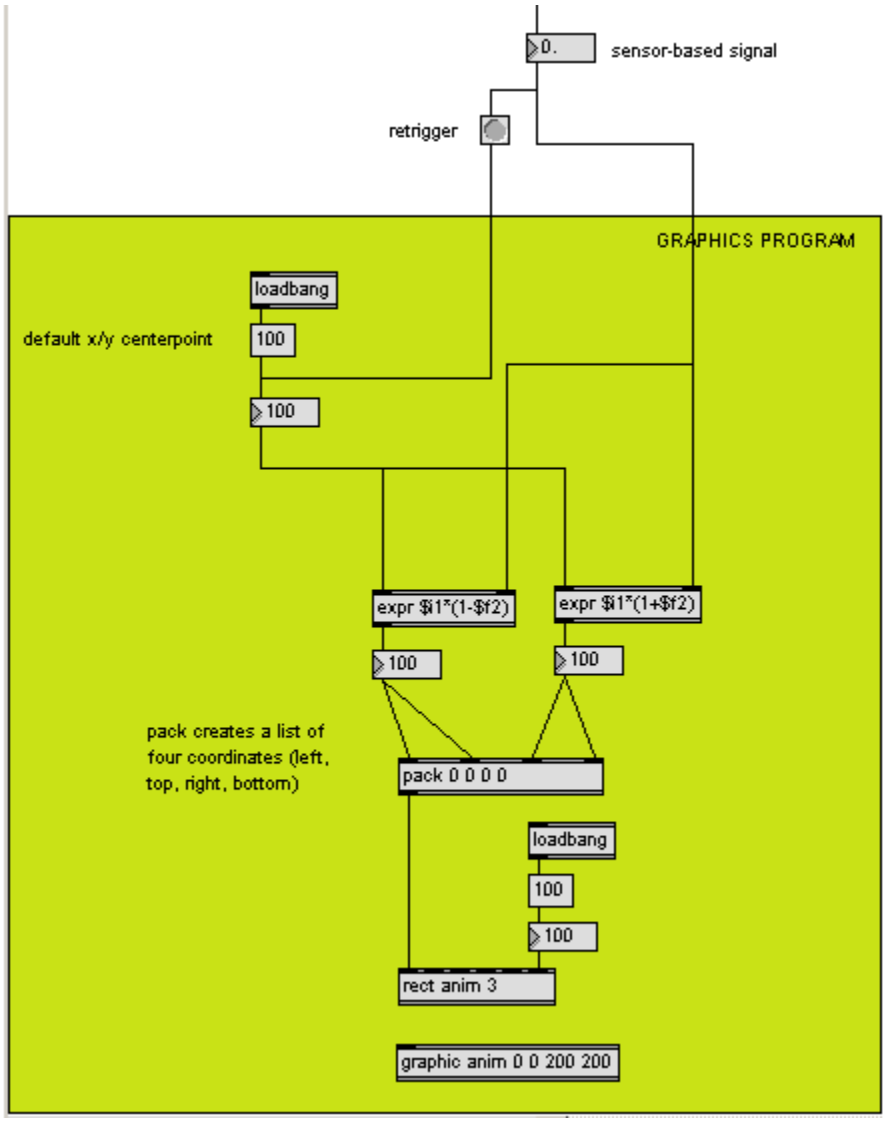
Assuming installation of drivers is completed, we will have access to the t.intro.ain object in Max/MSP, which is our chief interface with the Teleo device – which looks like this.




It allows that both sampling rate and resolution can be set through it – as well as setting of minimum and maximum values that will be output; hence we do not need to bother too much with manual signal scaling (this output will however follow the quantisation steps defined by the sampling resolution). The t.intro.ain object provides the latest sample values as a floating point (or integer) number and that value is refreshed according to the set sample period:



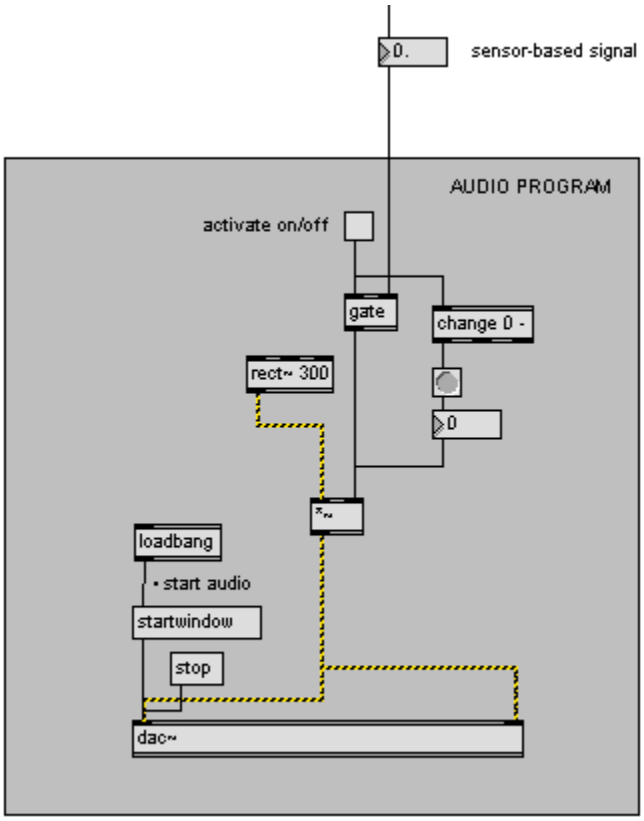
Once we have the sensor variable available, we can apply it – here for instance, as a scaling parameter of a red square:




In this program, simply speaking we define a center point of (x, y) = (100, 100) and we use the sensor parameter, as a floating number from 0 to 1. Using this parameter, we calculate left, top, right and bottom coordinates of the square, and we display it in the graphic anim window.

Here there is no direct 'endless' loop but it is implied, since the graphics program is started each time the sensor values is refreshed – according to the example above it happens each 100 ms, or with 10 Hz refresh rate.

In Max, it is also easy to apply a sensor value to sound:



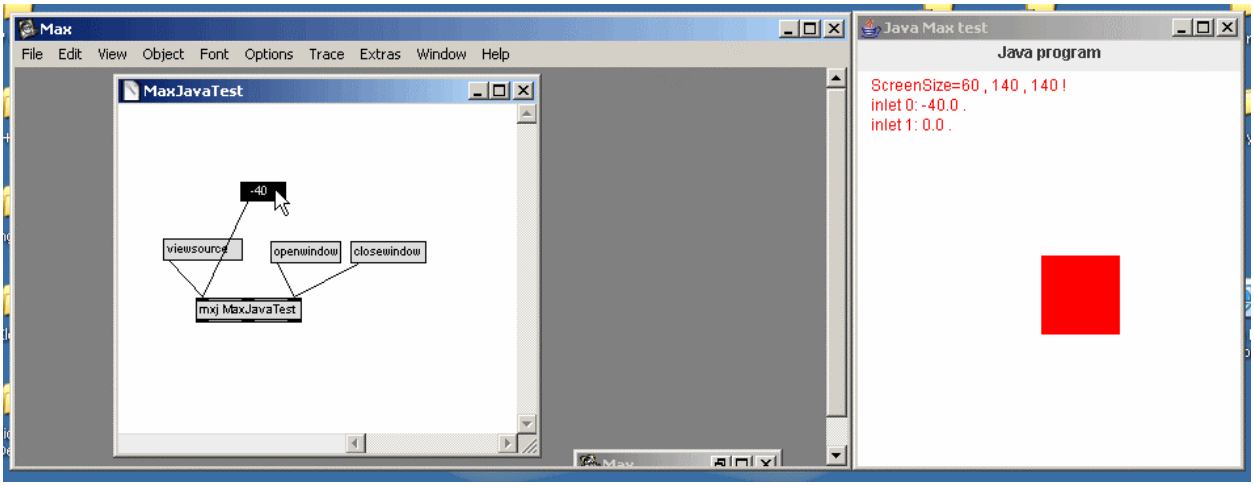

Here for instance, we simply use the sensor value to multiply an MSP audio signal – as it is from 0 to 1, it acts as a volume regulator. Here we realize that rect~ as other MSP signals, imply an internal loop that runs at 44.1 KHz. If the sensor refresh rate is as previously 100 ms, as a slower one, it will win as previously discussed – at such a slow refresh rate, clicks and / or delays when the sound is processed may be noticed.

# 6 Example program Java (via Max/MSP)

We can interface a Java program with Max/MSP patch through an object in Max/MSP known as mxj. The connection between Max and Java will be discussed in more detail in Software Design – here we provide some short points. Anoother resource you might wanna check is the Max Java Documentation tutorial, located at

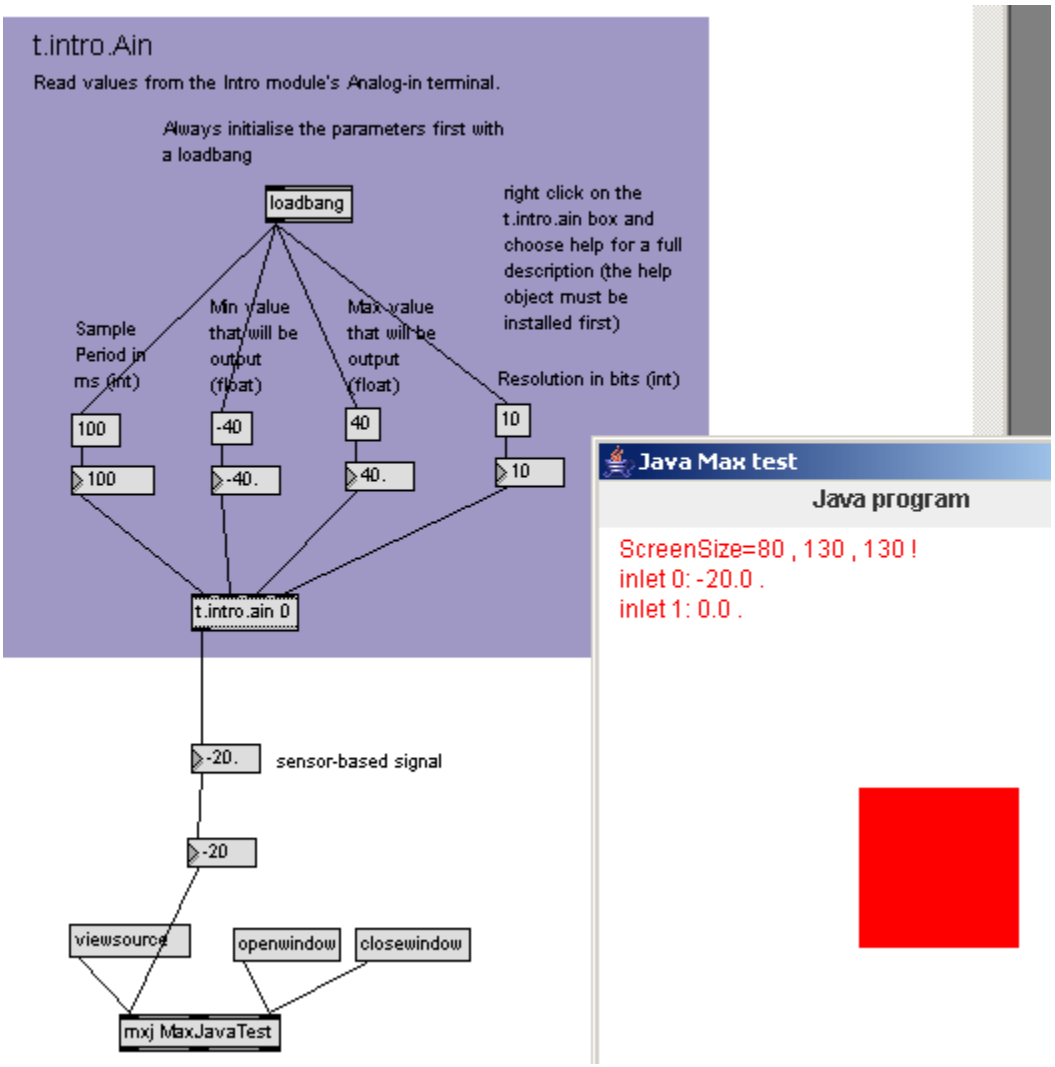
C:\Program Files\Cycling '74\MaxMSP XXX\java-doc\tutorial\html\index.html

The mxj object will act as a graphic container for the Java program, and just as any Max object, it can have inlets and outlets – through the inlets, we can pass a value to the Java program. In the example below, we use a number box, to send a value to the Java program, according to which we can change, for instance the scale of the red square.




Afterwards, interfacing with Teleo is simply to connect the sensor output to the corresponding number box – and the sensor values will thus be automatically sent to Java. In this case, we can work with Max and Java (and Max in general) – even without actually having a circuit connected – since we can change number boxes in Max by hand; so we can develop code *without* actually having a sensor circuit connected.

In this sense, the connection from Teleo to a Max patch would look like this:




Several things should be mentioned about the Max/Java connection:

The Java source code (say MaxJavaTest.java) should be placed in the folder C:\Program Files\Common Files\Cycling '74\java\classes

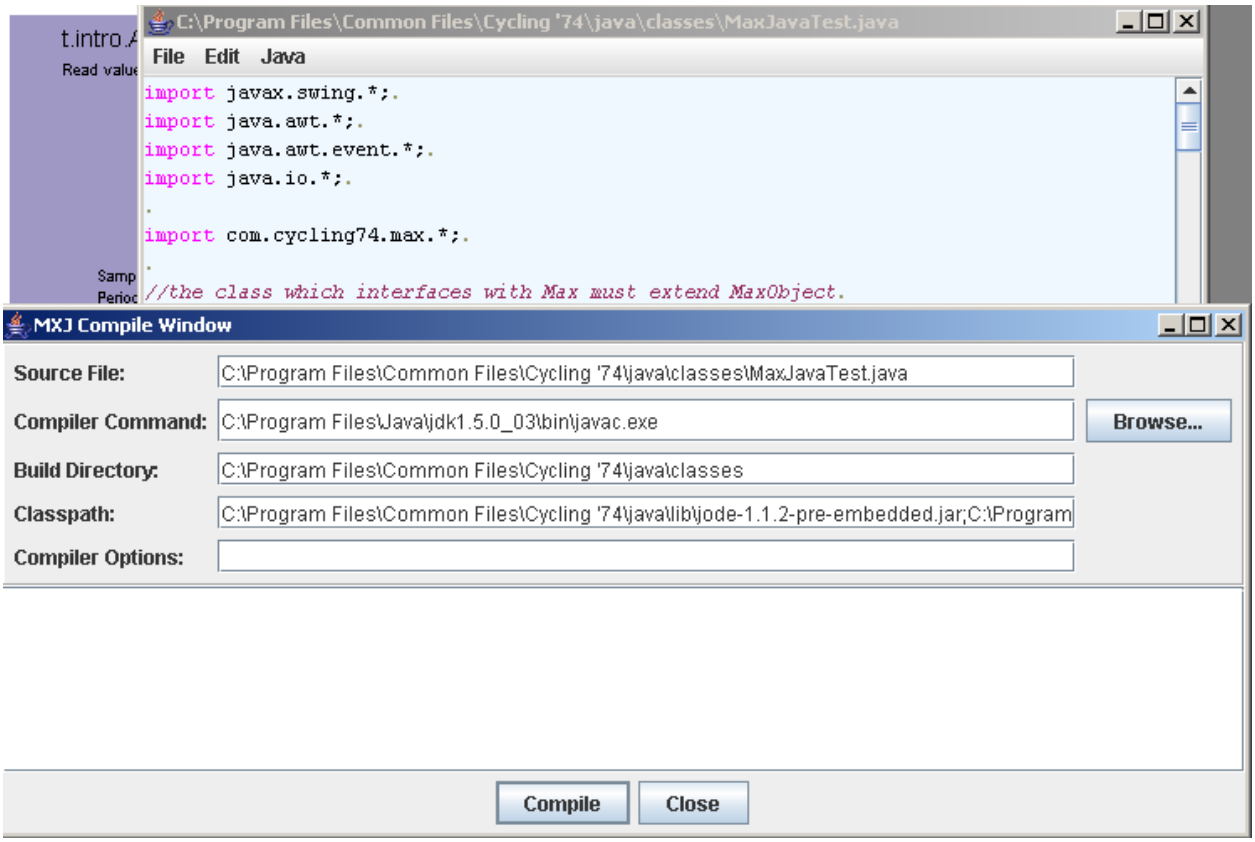
and the Max patch (say MaxJavaTest.mxb) in a folder of your choice.

While the Java code is not compiled (we have no $class files in the C:\Program Files\Common Files\Cycling '74\java\classes directory), Max will not be able to find the corresponding object. Hence we do a trick:

- enter mxj help in the mxj box – this will open the example help.java which is compiled
- by clicking on viewsource we can then open the source window

- through that window we open the MaxJavaTest.java source code
- we compile the source code – which generates the needed class files
- we enter back mxj MaxJavaTest in the mxj box – it should now be found

When the source window is open, by going to Java/Compile we open the Compile window – the Compiler Command should point to a valid java compiler (javac.exe), which comes with Java Development Kit (jdk), otherwise Java classes cannot be compiled.




Finally, a couple of words about the Java class:

1. the master class of the program should extend MaxObject class, (which is defined by Max/MSP)

```
public class MaxJavaTest extends MaxObject implements Runnable
```

2. What would otherwise be a master Java class, becomes a child class of the master one:

```
public class MaxJavaTestFrame extends JFrame
```

3. Max inlets/outlets of the mxj box are declared in master class constructor

4. Max messages can be processed either through ‘anything’ method or by declaring a function in the master class

5. Max number values brought to inlets should be processed through ‘inlet’ method of the master class – in there, check which inlet is activated through getInlet(), and act accordingly (a sensor value should be saved in a local variable)

```
public void inlet(float f)
```

6. Refresh of screen can go either each time a sensor inlet is triggered, or in a run method (which would act as an ‘endless’ loop in this case). Again, the worse refresh rate wins.

7. The local variable copy of the sensor value accepted from Max, should be directly applied in the drawing function – for instance paint() method of a Java canvas.

# Resources and references

[1]. Making Things. “TELEO STARTER KIT USER GUIDE.” http://www.makingthings.com/teleo/products/documentation/teleo_intro_user_guide/