

# TN.h — Arduino Library for Tangible Networks TN-04

Espen Knoop

This is an Arduino library for use with Tangible Network TN-04 nodes. It handles inputs, outputs, reading switches etc. For serial communication, use the standard Arduino functions.

The library defines a class TN, so using it requires creating a TN object and calling its methods. Here is a minimal example of use:

```
#include<TN.h> // Requires TN.h, TN.cpp, Keywords.txt in folder <Arduino>/Libraries/TN/
TN Tn = TN(-1.0,1.0); // Create TN object with range [-1.0, 1.0]
void setup () {} // Don't need anything in here - inputs/outputs set up in constructor
void loop () {
  Tn.colour(255,255,255); // Set LED to white
  delay(500);
  Tn.colour(0,0,0); // Set LED to off
  delay(500);
}
```

Nodes have 3 inputs and 3 outputs. Inputs/outputs are numbered 1-3 so as to be in keeping with labels on the PCB/units. Nodes also have a pot (potentiometer, knob), a pushbutton switch and 3 DIP configuration switches that can be switched with a small screwdriver or similar.

Most models will probably use `analogRead()` and `analogWrite()`, but `digitalRead()` and `digitalWrite()` are also provided for models requiring only binary information (on/off) to be sent between the nodes.

The library defines the following methods:

<code>TN(double minVal=0.0, double maxVal=1.0)</code>	Constructor for TN object. Input arguments specify range of <code>analogRead()</code> and <code>analogWrite()</code> : values outside range will be clipped. If arguments are not specified, range is set to [0.0,1.0].
<code>void colour(int r, int g, int b)</code>	Set LED colour. Integer arguments $\in [0, 255]$ .
<code>void colour(double r, double g, double b)</code>	Set LED colour. double arguments $\in [0.0, 1.0]$
<code>boolean isConnected(int input)</code>	Returns <code>true</code> if input is connected, <code>false</code> otherwise.
<code>double analogRead(int input)</code>	Read the value of an input. Returns <code>minVal</code> if input is not connected.
<code>void analogWrite(int output, double value)</code>	Write a value to an output. Value is clipped if outside [minVal,maxVal] range.
<code>int digitalRead(int input)</code>	Read the value of an input as <code>true</code> or <code>false</code> . Only use in conjunction with <code>digitalWrite()</code> .
<code>void digitalWrite(int output, int value)</code>	Write an output to <code>true</code> (maxVal) or <code>false</code> (minVal).
<code>boolean dip1()</code>	Get state of DIP switch 1 ( <code>true</code> is on).
<code>boolean dip2()</code>	Get state of DIP switch 2 ( <code>true</code> is on).
<code>boolean dip3()</code>	Get state of DIP switch 3 ( <code>true</code> is on).
<code>boolean sw()</code>	Get state of pushbutton switch ( <code>true</code> is pressed).
<code>double pot()</code>	Get position of pot. Returns <code>double</code> between 0.0 (fully CCW) and 1.0 (fully CW).
<code>boolean masterConnected()</code>	Returns <code>true</code> if master controller is connected.
<code>double masterRead()</code>	Get value of master controller. Returns <code>double</code> between 0.0 (fully CCW) and 1.0 (fully CW). Returns 0.0 if master controller is not connected.
<code>void printState()</code>	For debugging. Prints out the current state (ins, outs, switches etc) to serial. Requires <code>Serial.begin(115200)</code> in <code>setup()</code> . Runs in approx. 5 ms.

The library also includes fast functions for max and min, `MAX(x,y)` and `MIN(x,y)`, as well as `MINMAX(x,l,u)` which returns `x` if `l < x < u`, `l` if `x < l` and `u` if `u < x`.