Arduino Web Server Gauge Displaying Analog Value

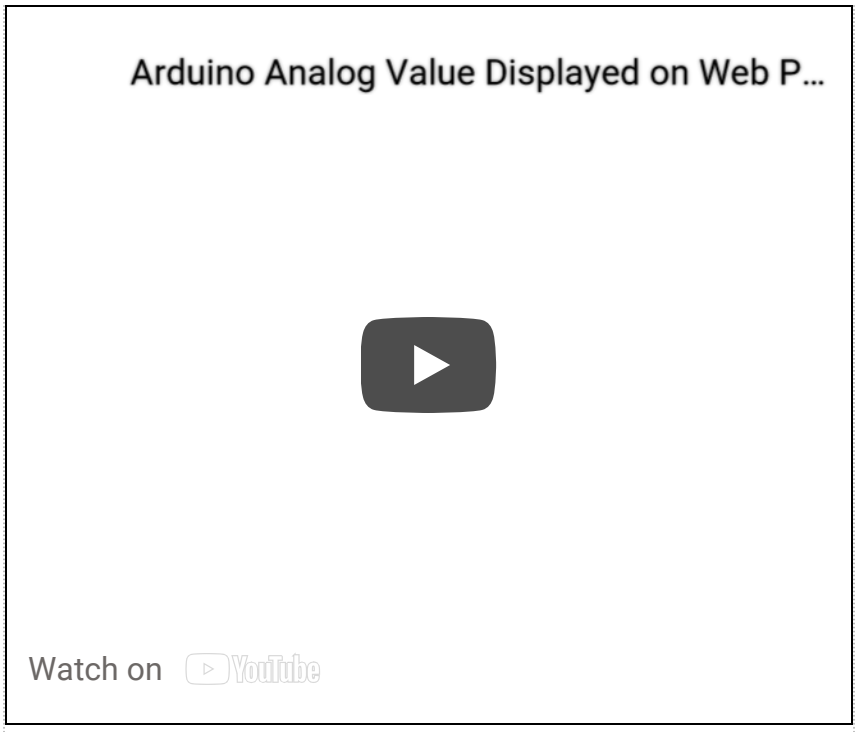
Created on: 28 March 2013

Part 15 of the Arduino Ethernet Shield Web Server Tutorial

A gauge component is used to display the analog value from one of the Arduino's analog pins on a web page. The dial gauge is updated using Ajax.

The gauge is written in JavaScript and uses the HTML5 canvas. The gauge is used as a component (unmodified) and is simply set up to display the analog value of one of the Arduino analog pins. The value is updated every 200ms.

This video shows the gauge on the web page that is hosted by the Arduino web server.



Can't see the video? View on YouTube →

Getting the Gauge Component

The gauge component is written by Mykhailo Stadnyk and can be downloaded from Mikhus at github. Also see a blog article on the gauge.

The JavaScript code from the file gauge.min.js (downloaded from the above github link) was used on the web page in this part of the tutorial.

Arduino Sketch and Web Page

Arduino Sketch

The Arduino sketch for this part of the tutorial is shown here.

```
/*-----------------------------------------------
  Program:      eth_websrv_SD_Ajax_gauge

  Description:  Arduino web server web page displays
```

```
The web page is stored on the SD ca
Ajax is used to update the analog v
web page.
```

This sketch is a slightly modified version of the sketch from the previous part of this tutorial (part 14).

The sketch sends a single analog value from Arduino pin A2 to the web browser which is then fed to the dial on the web page. The analog value is updated on the web page using Ajax and the value is sent as part of an XML file from the Arduino.

The web page hosted by the Arduino web server is stored on a micro SD card on the Arduino Ethernet shield.

Web Page

The web page for this example is shown here.

```
<!DOCTYPE html>
<html>
    <head>
        <title>Arduino Ajax Dial</title>
        <script>
        var data_val = 0;
        eval(function(p,a,c,k,e,r){e=function(c){return(c<a?'':e(parseInt(c/a)))+((c=c%a)>35?String.fromCharCode(c+29
        function GetArduinoInputs()
        {
            nocache = "&nocache=" + Math.random() * 1000000;
            var request = new XMLHttpRequest();
            request.onreadystatechange = function()
            {
                if (this.readyState == 4) {
                    if (this.status == 200) {
                        if (this.responseXML != null) {
                            document.getElementById("input3").innerHTML =
                                this.responseXML.getElementsByTagName('analog')[0].childNodes[0].nodeValue;
                                data_val = this.responseXML.getElementsByTagName('analog')[0].childNodes[0].nodeValue;
                        }
                    }
                }
            }
            request.open("GET", "ajax_inputs" + nocache, true);
            request.send(null);
            setTimeout('GetArduinoInputs()', 200);

        }
    </script>
    </head>
    <body onload="GetArduinoInputs()">
        <h1>Arduino Ajax Dial</h1>
        <p>Analog (A2): <span id="input3">...</span></p>
        <canvas id="an_gauge_1" data-major-ticks="0 100 200 300 400 500 600 700 800 900 1000 1023"
                data-type="canv-gauge" data-min-value="0" data-max-value="1023"
                data-onready="setInterval( function() { Gauge.Collection.get('an_gauge_1').setValue(data_val);}, 200);">
        </canvas>
    </body>
</html>
```

The Web Page for Displaying the Gauge with Gauge Code Cut Off – click for a bigger image

The web page is a modified version of the web page used in the previous part of this tutorial.

The Gauge

The gauge is made available to the web page by inserting the gauge code between the script tags in the head of the web page HTML code.

The gauge is displayed by using the HTML5 canvas tag in the body of the HTML.

Updating the Gauge

The analog value from the Arduino is received in the same way as it was in the previous part of this tutorial and displayed at the top of the web page in the same way as the previous part of this tutorial. The analog value is also saved to the JavaScript variable **data_val** so that it can be used by the gauge.

The value in **data_val** is fed to the gauge by using the line of JavaScript in the data-onready property of the canvas tag:

```
setInterval( function() { Gauge.Collection.get('an_gau
```

The 200 in the above line tells the gauge to update every 200ms (200 milliseconds).

Timing

The gauge updates every 200ms, and the Ajax request for an analog value is also set to 200ms.

The timing for the Ajax refresh of the analog data is done in this line of JavaScript code from the web page:

```
setTimeout('GetArduinoInputs()', 200);
```

This refresh rate may cause a problem on a busy or slow network. If there are any problems, try changing this value to 1000 to make the analog value refresh every second.

Running the Gauge Sketch

Wire up the potentiometer as shown in the circuit diagram of part 9 of this tutorial – leave the push buttons shown in the diagram off, they will not be used.

Copy the web page (index.htm) to a micro SD card and insert it into the card slot of the Arduino Ethernet shield.

Load the above sketch to the Arduino web server. The web page is available for download below.

Web Page Source Code

Download the web page source code (index.htm) for this part of the tutorial here:

Arduino_web_gauge.zip (6.3 kB)

The license for the gauge is included in the download as a text file.

← Go back to Part 14 Go to Part 16 →

Comments

Login

There are no comments posted yet. Be the first one!

Post a new comment

Enter text right here!

Comment as a Guest, or login:

Name	Email	Website (optional)
Displayed next to your comments.	*Not displayed publicly.*	*If you have a website, link to it here.*

Subscribe to None

Submit Comment
