# WSDL WxShield

## Warnings and Cautions

## !!! VERY IMPORTANT !!!

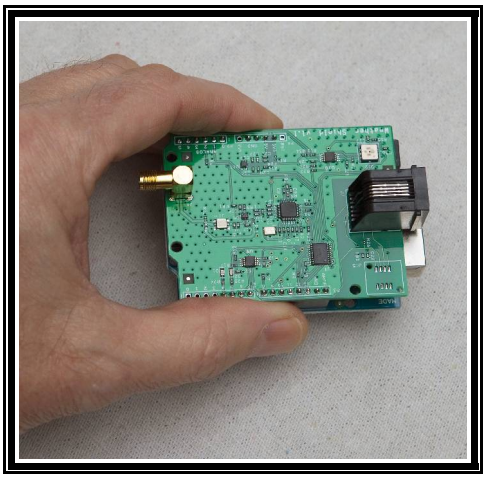
## Read carefully before unpacking the WxShield

By following the cautions and warnings below, your WxShield will enjoy a long, accurate and productive life!

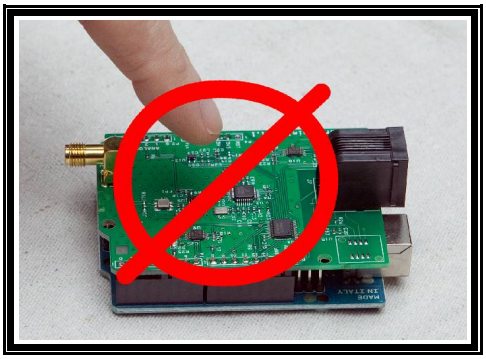
# Static Discharge Warning

This is a prototype item, not a finished product and as such is not fully protected against damage caused by static discharge. Please read and follow these directions to avoid static damage to your WxShield.

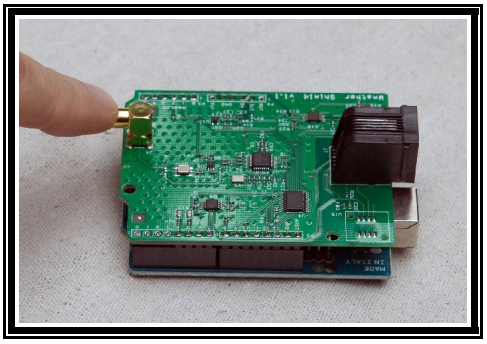
We recommend that you avoid physically handling the WxShield any more than necessary. The Arduino/WxShield package should be handled by the edges; avoid touching any of the circuitry on top of the WxShield. Before picking up the WxShield, it is a good idea to first make contact with the metallic housing of the antenna or USB connectors. This will safely drain the static charge you are always carrying. If the unit's USB port is plugged into a computer, touch the metal case of the computer (if it has one) before touching the WxShield; otherwise just touch the RF or USB connectors first.

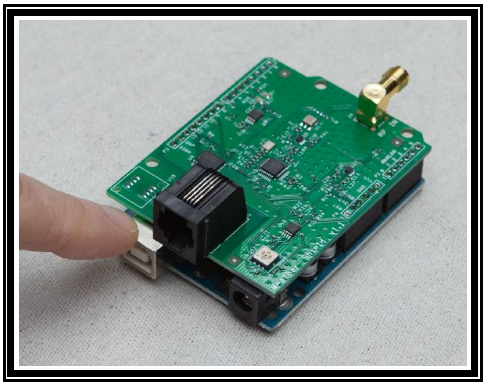


**Handle WxShield by the Edges**

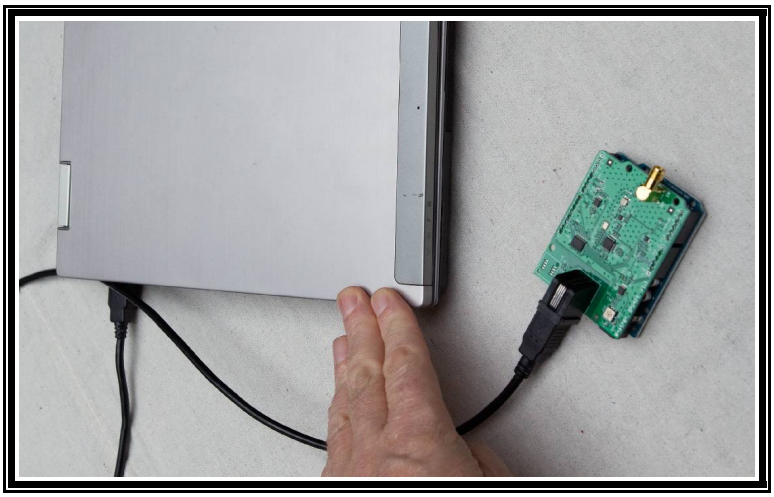


**Don't Touch the top of the WxShield**



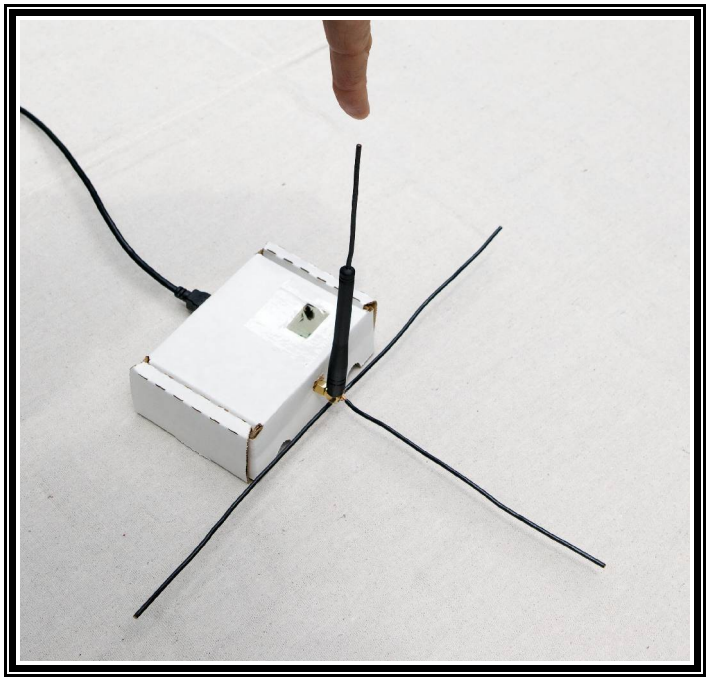


**Make first contact with metal body of either RF or USB connectors**



**When USB is connected, touch computer metal case first**

The antenna connection IS protected against static damage. You do not need to worry about accidental static discharges directly to the antenna, but intentionally zapping the antenna with static is still not a good idea.



**OK to touch antenna, but please avoid when possible**

*"No WxShields were harmed or mistreated in any way during the taking of these photographs"*

Most folks are aware of static discharge on dry days when they get “zapped” touching a door knob or other metallic object. The human body often carries static charges much smaller than this which cannot be felt but are still capable of damaging the WxShield’s sensitive electronic circuitry.

The symptoms resulting from static damage can range anywhere from subtle to catastrophic. For example, discharging small amounts of static into some of the more sensitive parts of the receiver circuitry (by touching the top of the WxShield) might cause only a slight degradation of receiver sensitivity. Each small discharge could make this a bit worse until the receiver eventually has trouble receiving strong, nearby sensors. Static damage is cumulative.

At the other extreme, a very strong static discharge to an unprotected part of the circuitry may completely disable the receiver, barometer or temperature/humidity sensor in one fail swoop.

## Watch Your Eyes!

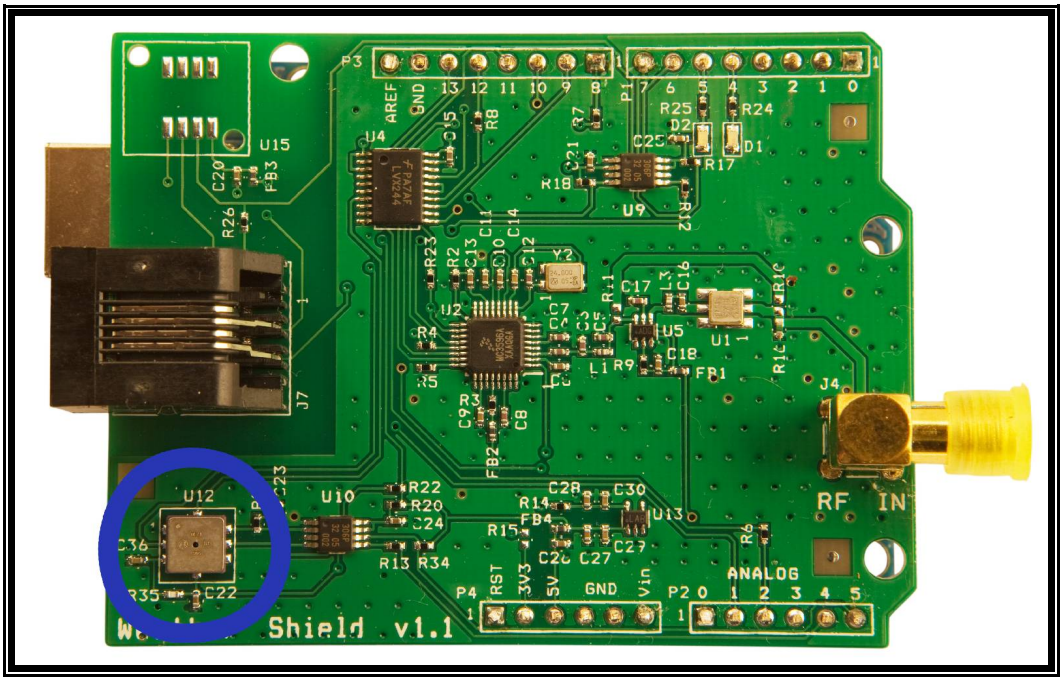
The four antenna wires are kind of sharp on the ends and the way the antenna is mounted next to the LED viewing window, someone might poke themselves in the eye while trying to look at the LEDs. In the interest of safety, some foam ear plugs have been glued onto the ends of the wires for protection.

There's a photo of this below. Yeah, they look goofy but if you remove them, please don't poke yourself in the eye with the wire and please don't blame us if you do – you were warned!

Goofy-looking eye protection!

# The Barometer is Light Sensitive

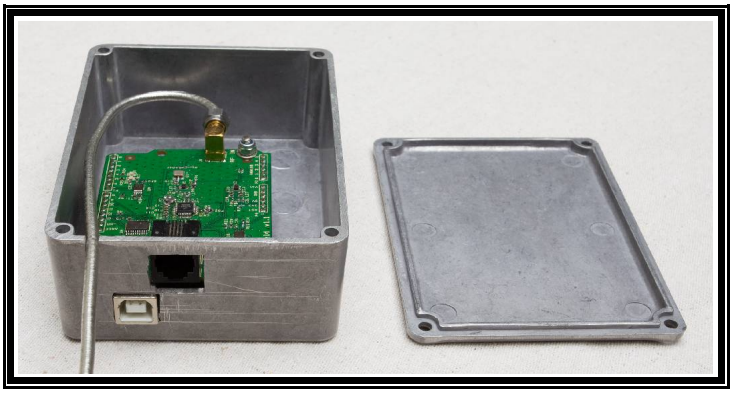
The BMP085 barometer has a visible hole in the top for sensing pressure. The barometer reading can be corrupted by light entering through this small hole so it is a good idea to keep the WxShield away from strong light sources.




The above photograph shows the barometer circled in blue. The small pressure sensing hole is visible in the middle of the barometer's metal lid.

# Mounting Box Suggested

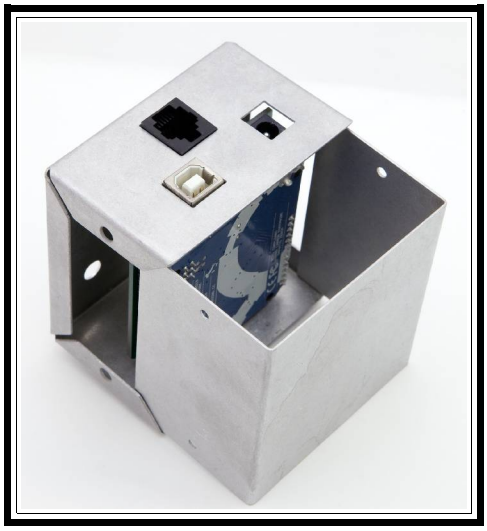
We recommend that you house the WxShield in a covered box to reduce the possibility of static damage and to keep stray light out of the barometer pressure port.



The cardboard shipping box can be used for this purpose and there are markings for holes on the inside of the box for this purpose. You can connect the antenna, USB and telephone cables through these holes. This will reduce the likelihood of static damage and provide a dark environment for the barometer.

One example of a custom mounting box is shown above.

Some WxShield units are now provided with an inexpensive custom aluminum enclosure, the “WxBox” pictured below. More information is presented on this enclosure later in this document.



## The “telephone” jacks are NOT telephone connections!

We have used standard “RJ14” telephone jacks as an inexpensive method to connect the remote SHT15 sensor board to the main unit. If you plug either one of these units into a standard telephone outlet it will probably destroy the unit.

***Do not plug the either the WxShield or remote board into a standard telephone outlet!*** This will void your warranty and you will be responsible for any damage caused.

## Acceptable Telephone Cords

The WxShield comes with a properly wired telephone cord roughly 36-inches in length. You can replace this with your own cord as long as 50 feet if desired. There are three requirements you must meet in doing so, however.

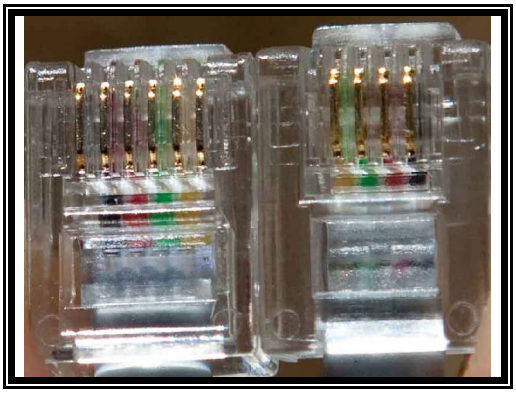
1. The telephone cord should have four conductors (RJ14). A six-conductor cord (RJ25) will work too, although only four of the six conductors will be used. Connecting a two conductor cord (RJ11) may damage the remote board and/or WxShield.

2. A cord wired for “voice” – NOT a “patch” cord must be used. Using a “patch” cord may destroy the SHT15 on the remote board.

3. The cord should be of the flat variety where the conductors maintain a constant relationship to each other throughout the length of the cord. Round cords where the wires are all wrapped into pairs may or may not work, especially with longer cords.

Below is a picture of the two ends of a cord wired for voice. Look at the order of colors from left to right on each plug, visible through the plastic body:

Left plug: Black, Red, Green, Yellow
Right plug: Yellow, Green, Red, Black

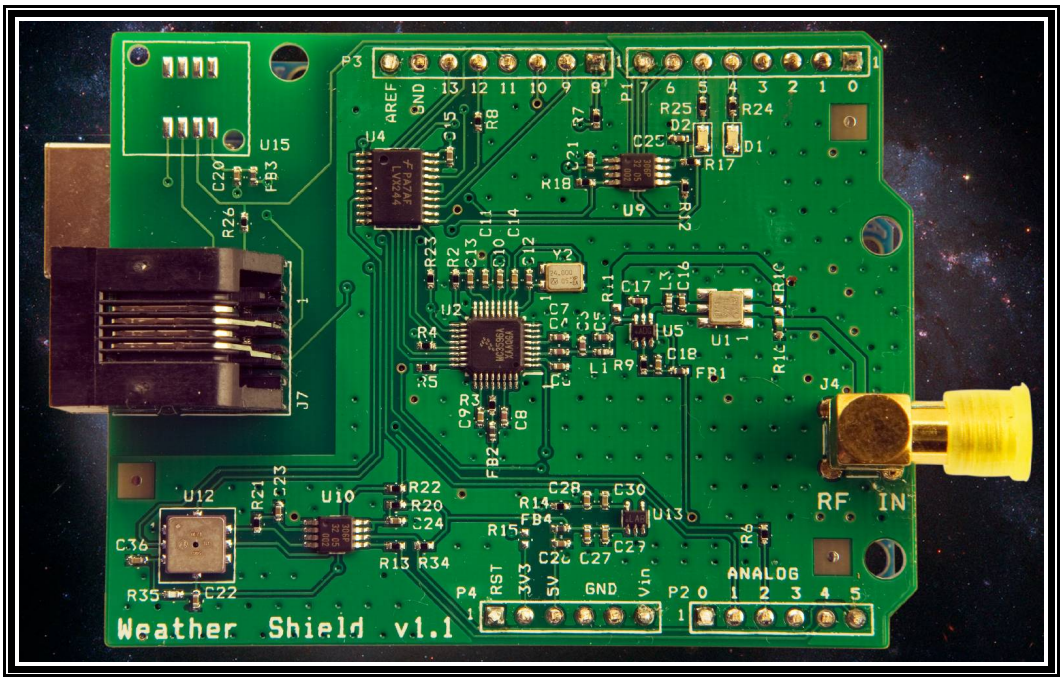


The wires are installed in reverse order on one of the plugs. This is called “voice” wiring. If the wires were in the same order on both plugs, this would be a “patch” cord instead. The majority of telephone cords sold in the stores are wired for voice, but you should confirm this before using one. You can do this visually if colored wires are visible as above, or with an ohm-meter or continuity tester otherwise. If you’re not sure, we recommend you stick with the included 36-inch cord.

In this photograph there are six metal contacts on the left-side connector but only the central four are actually being used. This is okay and acceptable if you are making up your own cables. Just be sure that you get the four wires connected to the middle four pins.

## Do NOT Connect/Remove Remote Board while Power is ON

The telephone cord must not be inserted or removed while power is applied to the WxShield (through the USB cable and/or DC power jack on the UNO board). Doing so may damage or destroy the SHT15 sensor on your remote board. Doing this may also damage or destroy interface circuitry on the WxShield main board.



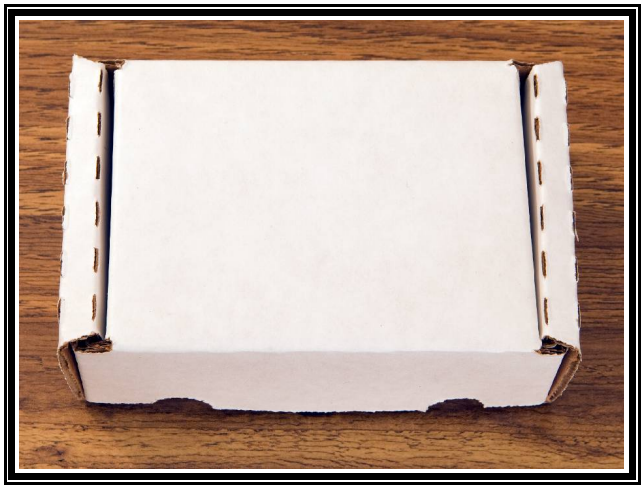

# WSDL Weather Shield User Manual

Version 2.3

January, 2013

# Unpacking and Setup

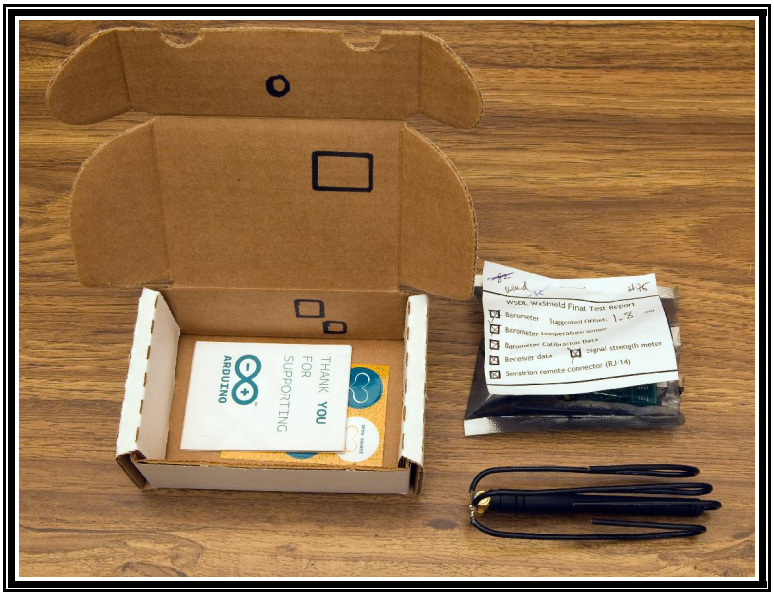
The WxShield may be delivered in a small shipping box as pictured below.



The box contains the WxShield, SHT15 remote sensor, SF1 filter cap, 3-foot RJ14 cord and custom modified antenna.

The dark antistatic bag contains the WxShield/Arduino board set, the remote SHT15 board and the SF1 filter cap for the SHT15 sensor. Do not remove the contents from this bag just yet.

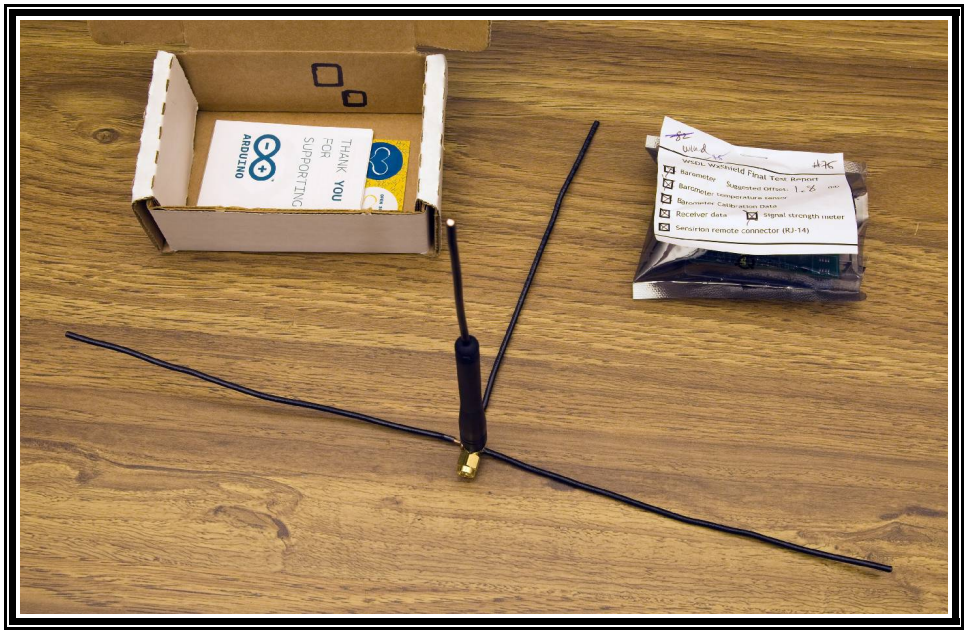
You will also notice several markings on the inside of the box to aid in making cutouts. This allows the WxShield to be mounted in the shipping box.




The antenna has been folded up to fit inside the shipping box.

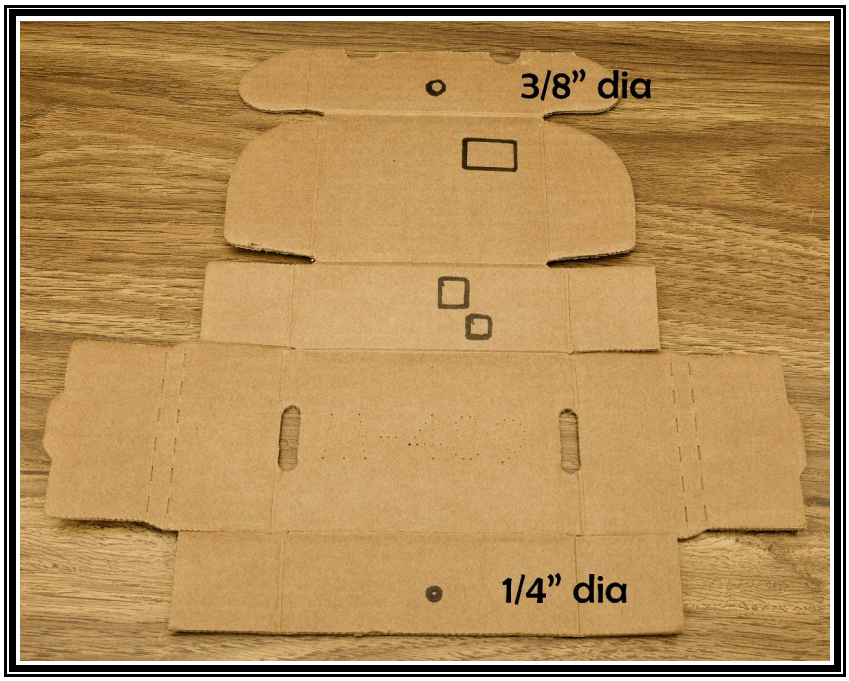
As shown below, the antenna wires have some goofy-looking foam ear plugs glued onto the ends. At the last minute it was realized that the ends of the wires are kind of sharp and someone might poke themselves in the eye accidentally. In the interest of product safety, these were added as a safety measure.

The next step is to carefully unfold the antenna's wires as shown in the two photos below. Some shipping boxes may not have the antenna wires folded up as much as this and there will be less unfolding to do.

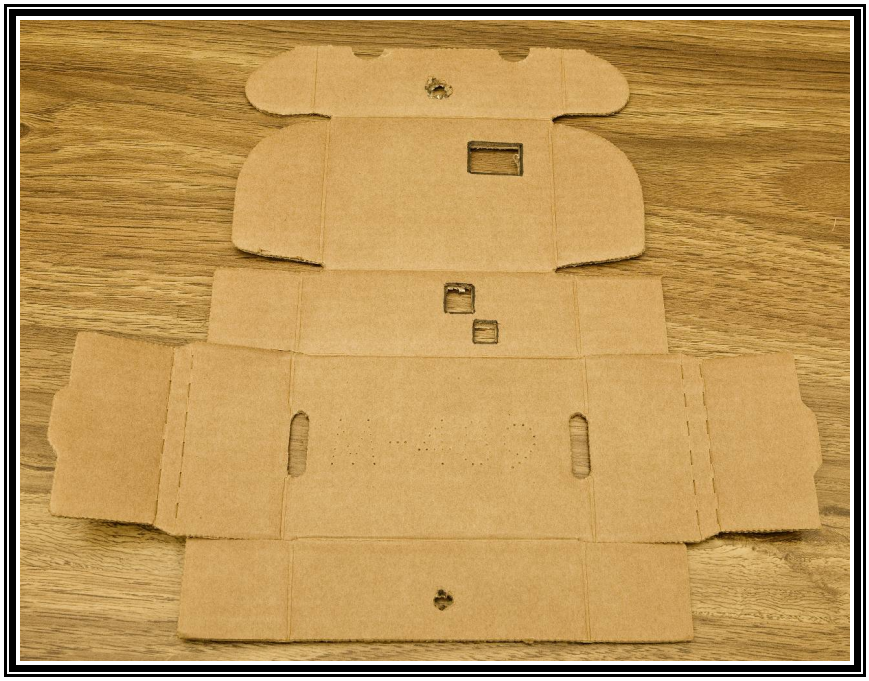




If you do not plan to mount the WxShield in the shipping box, this next part can be skipped.

Start by un-folding the shipping box and placing it on a flat surface such as a desktop as shown below.



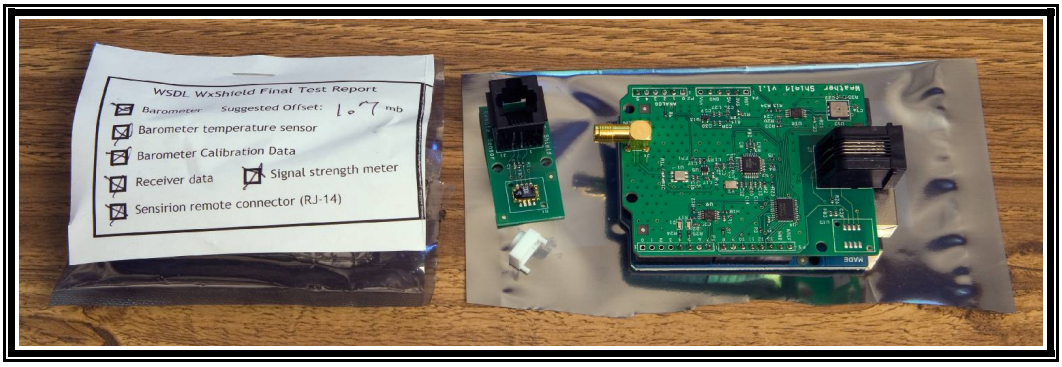

Place some newspaper or a magazine under the box to protect the desktop. Cut out three square and two round holes. It may be easier to use a drill to make the round holes. The hole diameters should be approximately 0.250 (1/4) and 0.375 (3/8) inches. Here's the box after cutting out the holes:



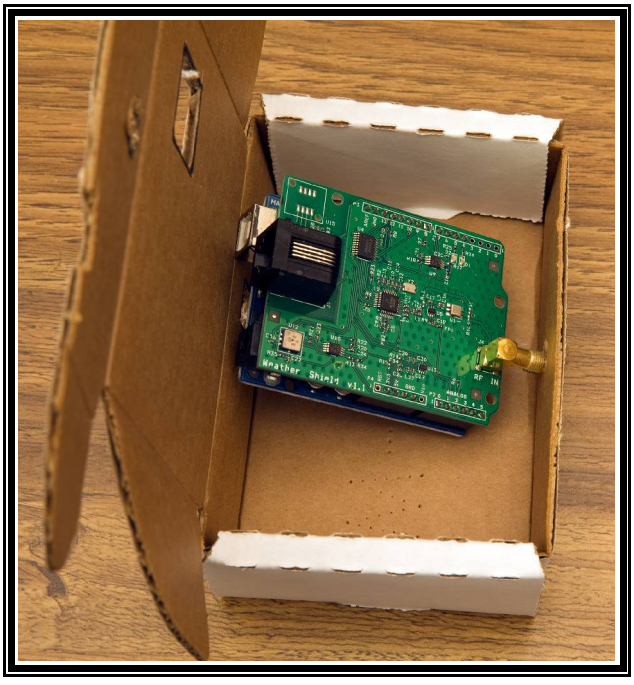
Carefully remove the WxShield, remote SHT15 board and SF1 filter cap from the antistatic bag. The white SF1 filter cap is very small so be careful not to lose track of it. The photo below shows these items removed from the bag. The

remote board and filter cap can be stored in the original anti-static bag until needed later.

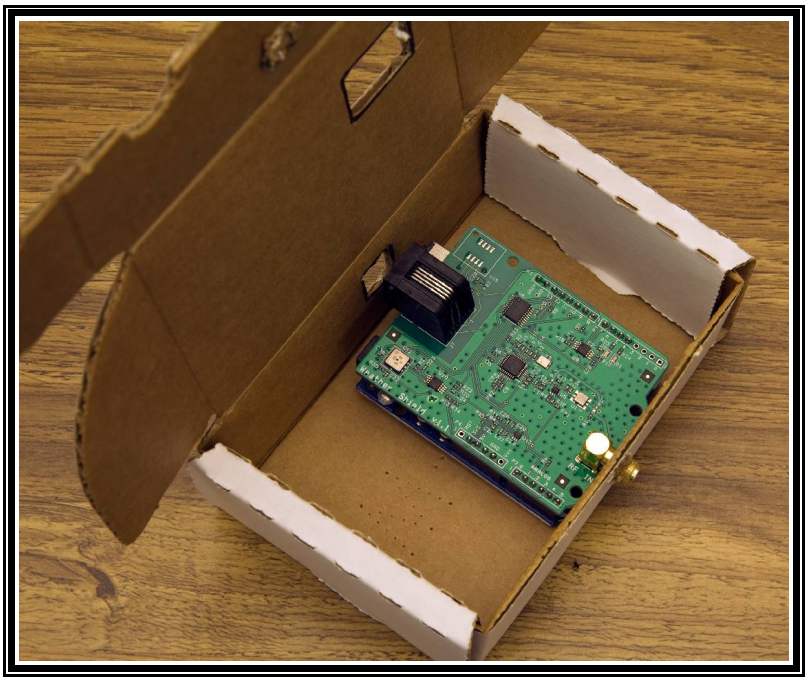
Remember, these items are static sensitive so handle them as little as possible and hold them by the edges to avoid touching circuitry on the top of the WxShield and remote boards.



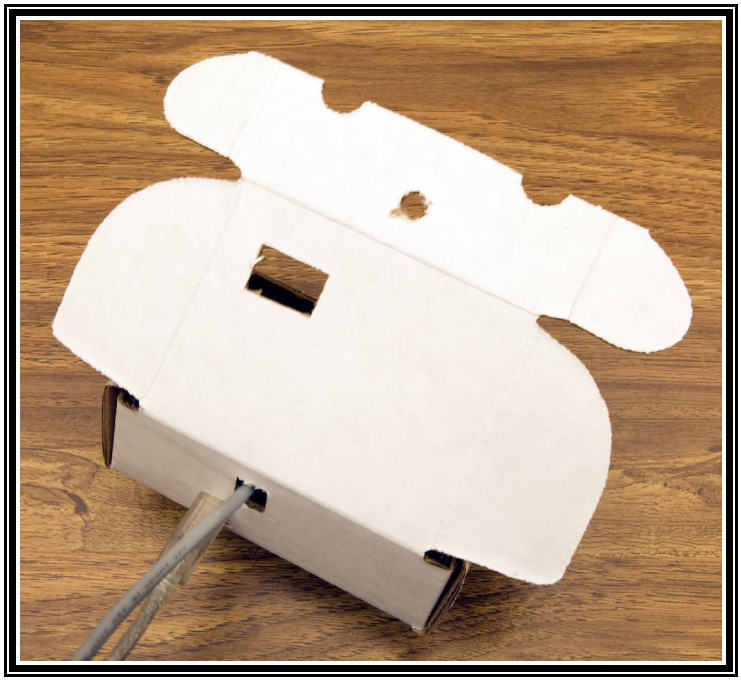

Now, fold the cardboard back up into a box. Insert the WxShield's antenna connector through the smaller hole in the side of the box. It may be necessary to tip the WxShield up a bit as shown below.



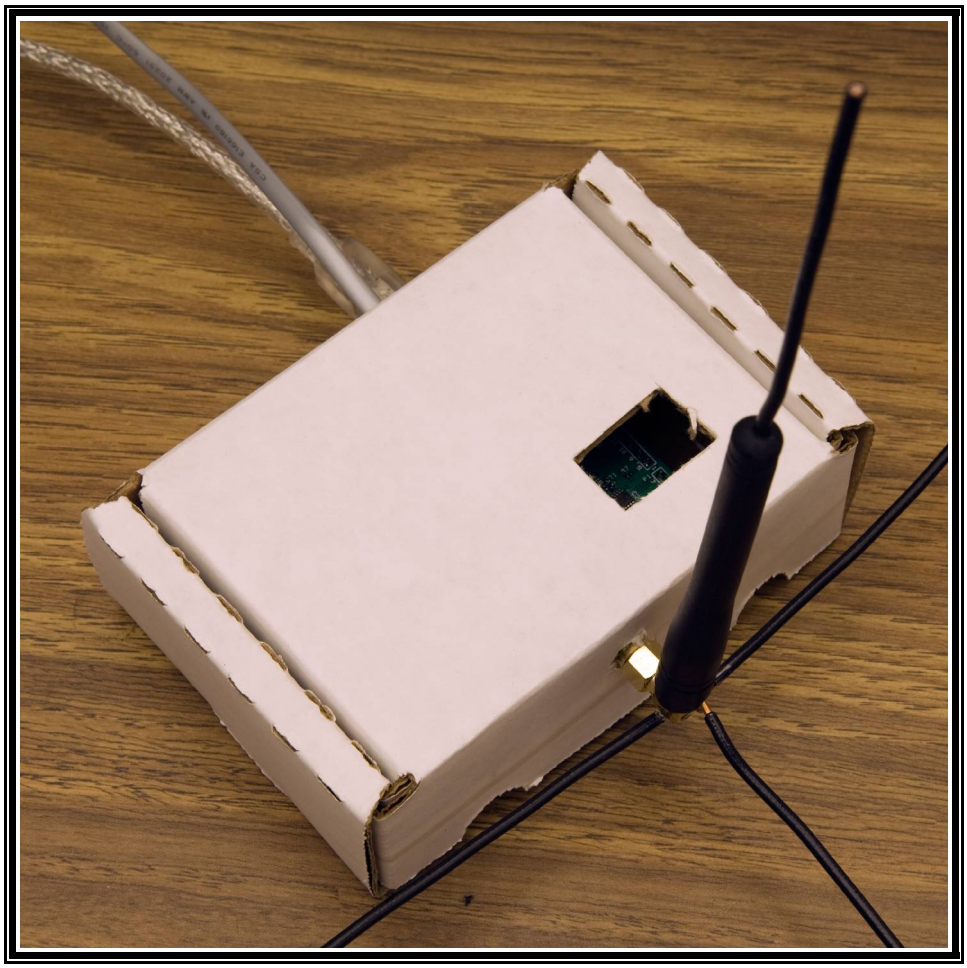

After pushing the connector through the hole, the board can be laid flat in the bottom of the box.



Holding the WxShield by the edges in one hand, plug in the USB cable and telephone cord through the square holes in the rear panel of the box.



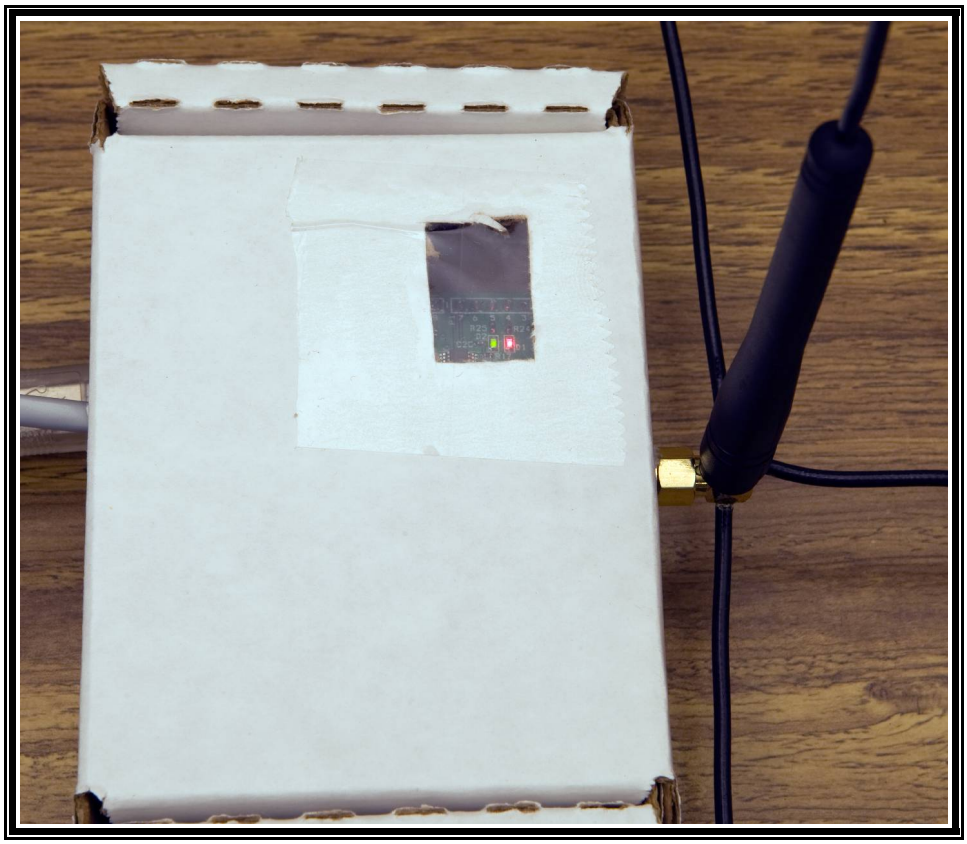
The box can now be closed; be sure to tuck the flaps in as shown. Holding USB connector if necessary, thread the antenna onto the round connector protruding from the other side of the box. It may be necessary to enlarge the hole in the box's front flap if it does not line up exactly.



The square hole cut into the lid of the box gives a view of the red and green LED lights on the WxShield. It is a good idea to cover this hole with a piece of clear tape to keep dirt and debris out of the box.

The barometer has a hole in the middle of its top through which atmospheric pressure is sensed. The barometer is sensitive to light entering through this hole and strong light may cause inaccurate readings or a malfunction of the barometer. The hole in the lid does not let enough light into the box to cause large errors, although it is a good idea to keep the opening away from direct sunlight while the WxShield is powered up.

The LEDs on the WxShield do not emit enough light to cause significant errors in the barometer reading.



Here is the WxShield installed in the shipping container with a piece of clear tape covering the observation hole in the lid. The red and green LEDs are both lit in this photo, and visible through the hole.

# Installing the Antenna Remotely

It is also possible to insert a piece of coaxial cable between the WxShield and the antenna. This allows the antenna to be located for better reception or for aesthetic purposes (out of sight). It is best to use 50-ohm cable, not the 75-ohm variety that is often used for television signals.

The WxShield uses “SMA” connectors on the antenna. Until recently, there was only one kind of SMA connector. With the advent of wireless internet routers however, a new kind of SMA connector has been created – the “Reverse Polarity” or “RP” SMA connector.

The WxShield uses normal polarity SMA connectors so be careful not to purchase the RP variety as they will not work. Below are two photos showing the ends of two coaxial extension cables. The one on the left is normal polarity (CORRECT) and the one on the right is reverse polarity (WRONG).

**Normal Polarity**

**Reverse Polarity**

The normal SMA connector features a protruding center pin on the end with the nut. In reverse connectors, the center pin is on the end with the male threads.

There is one other thing to consider in selecting an extension cable. All coaxial cable has loss – adding a cable between the antenna and WxShield will always result in some loss of signal. Just how much loss depends on the type of coaxial cable used and its length.

Each type of coaxial cable has a different amount of loss per unit length. This is usually specified in units of decibels (or “dB”) per 100 feet (or meters) of length. For example, the extension cables shown above are made with “RG174” coaxial cable. An internet search for “RG174 loss db” reveals that this cable has a loss of 60dB per 100 meters. A 10-meter length of this cable would have a loss equal to one tenth of the loss for 100 meters, or 6dB.

The “RG58” variety of cable has a loss of roughly 28dB per 100 meters, so a 10-meter extension cable made from RG58 would have a loss of about 2.8dB.

Loss also depends on the frequency of the RF signals (434MHz in our case). Cable loss tables will show the loss at several different frequencies; pick the entry closest to 434MHz. There will usually be entries for 400, 450 or 500MHz and any of these values is close enough. This data is sometimes also shown as an X-Y graph so you can read the loss at 434MHz directly from the graph.

So then, how much loss is too much? Its difficult to give an exact answer, but here are some rough guidelines to help in selecting an extension cable.

- Anything less than 3dB of loss will probably not be noticeable.

- Loss up to 6dB is probably fine, although you may notice some reduction in signal strength depending on the antenna's new location. On the other hand, relocating the antenna may increase signal strength by more than 6dB so there is often a net gain.

- Loss values above 10dB will be noticeable in many cases and exceeding 20dB of loss will often result in loss of signals. With this much loss, a low-noise preamplifier might need to be installed at the far end of the coax (at the antenna); unfortunately that task is beyond the scope of this manual.

Remember – these are just guidelines and your results ***will*** vary.

Here is a quick overview of three types of coaxial cable commonly available. There are many, many other types which will also work if the loss is acceptable.

The coaxial cable you choose may or may not be readily available with SMA connectors on each end. As a result, it may be necessary to purchase adapters to connect these cables to the WxShield's SMA fittings.

- RG174 (60dB/100m) can usually be purchased with a pair of male/female SMA connectors. RG174 cables setup like this between 10 and 30 feet in length are available on E-bay for as little as $15US. RG174 cables longer than about 20-30 feet may have too much loss.

- RG58 (28dB/100m) may not be available with SMA connectors. The least expensive viable alternative is probably to get male BNC connectors on each end. This will require two adapters (photo below); one female BNC to male SMA, and one female BNC to female SMA. 50-foot lengths are often available on E-bay for less than $15US and the two adapters can usually be found for less than $10US. Runs of RG58 longer than about 75-100 feet may have too much loss.

- For those really long runs, RG8 (9dB/100m) may be required and either BNC, TNC or Type-N male connectors are often used here. It is best to

avoid PL259 connectors – they are not designed to work at these frequencies. Long lengths of cable like this are often custom ordered cut to length with your choice of connector. This can start to get a bit pricey. Again, appropriate adapters are required for connecting the antenna and WxShield at each end.

There are two photos below showing the adapters that would be required to use a cable that came supplied with male BNC connectors on each end (only one adapter of each type is required).

**BNC Female to SMA Female**

**BNC Female to SMA Male**

One last note: even within each cable type there can be variation in the amount of loss due to different materials and manufacturing techniques, so check the manufacturers data sheet for that exact cable if available (this is not always possible). For example, different grades of RG8 cable have loss that varies between 9dB and 21dB per 100-meters.

# Setting up the Remote Board

In the previous setup steps, the telephone extension cord was plugged into the WxShield. Now plug the other end of this cord into the remote SHT15 sensor board. If desired, you can postpone installing the SF1 filter cap over the SHT15 sensor until proper operation is verified. In fact the SF1 does not technically need to be installed at all but it is a really good way to protect your $30US SHT15 sensor from dust and dirt.

The SHT15 board is very small and lightweight. If necessary, put a piece of tape near the end of the phone cord to keep the remote board from flopping around while initially powering up the WxShield.

Use the included 36-inch telephone cord when first powering up the WxShield. If you later wish to use a longer replacement cord, be sure it is wired for "voice" as explained in the Cautions & Warnings section at the beginning of this document. Failing to do so may result in permanent damage to the SHT15 remote board and/or WxShield.

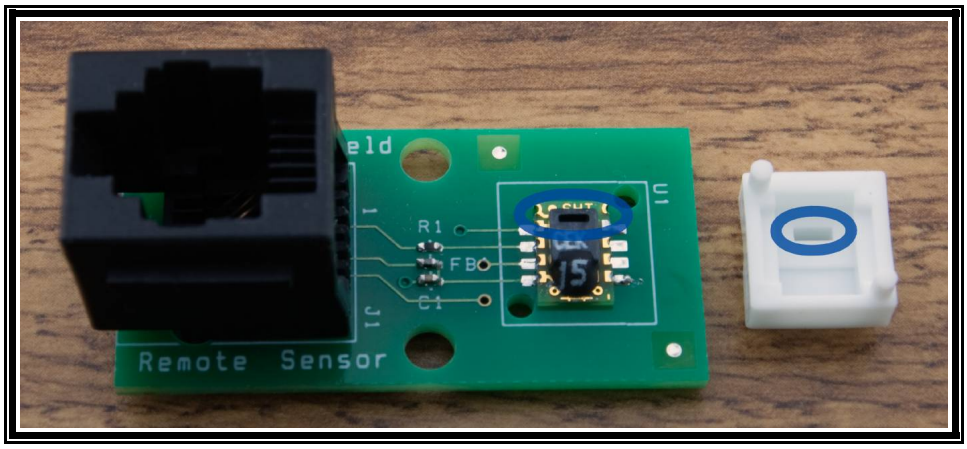
## Installing the SF1 Filter Cap

Installing the filter cap is highly recommended to protect the SHT15 sensor from environmental contamination (dust, dirt, etc.). It will slightly increase the response time to changes in temperature or humidity but for the majority of users this is not an issue.

You can download a datasheet for the filter cap at the URL below which includes more information on the installation process.

www.sensirion.com/en/pdf/product_information/Data_sheet_filter_cap_SF1_E.pdf

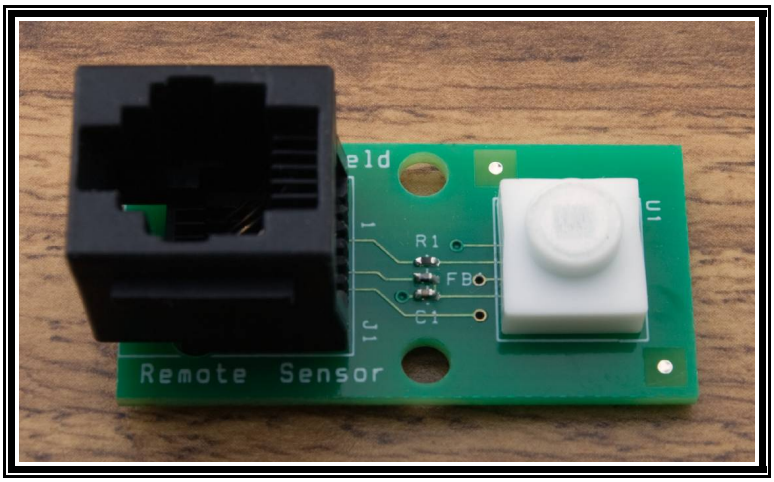
The cap can be inserted into the PCB mounting holes two ways. The photo below shows the correct orientation.



There is a rectangular slot in the top of the sensor (on the left) and a matching slot in the underside of the filter cap. Both slots are circled in blue. The filter cap should be aligned as shown above and then flipped over (left-for-right) and placed over the sensor.

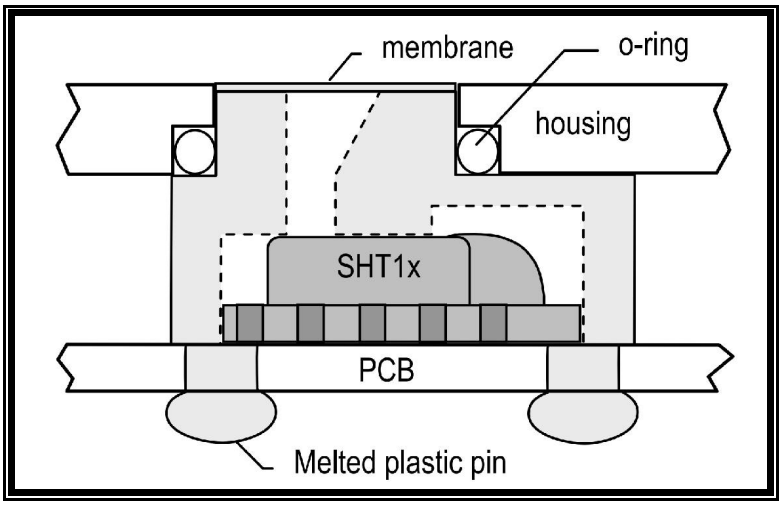
The next photo shows the filter cap properly inserted into the holes in the remote PCB. The next step is to secure the filter cap in place. There are a couple ways to accomplish this as described in the SF1 data sheet.

Unless the remote board will be subjected to relatively harsh environmental conditions, it should be adequate to melt the pins under the PCB to secure the filter cap. Included below is a drawing from the SF1 data sheet which depicts the melted pins.




Unlike the drawing, there will be no “o-ring” or “housing” structure. This drawing simply illustrates how the pins may be melted to secure the filter cap. We have not actually tried doing this before and so cannot offer much guidance.

Sensirion suggests using a hot iron at least 160ºC for this purpose. A soldering iron tip may be too hot and could result in burning the plastic so something a little cooler is probably needed.


membrane
o-ring
housing
SHT1x
PCB
Melted plastic pin

# Powering Up the WxShield

After optionally installing the shield in the shipping box or other housing, you are ready to power it on for the first time. You will need a copy of the driver configuration file named "Arduino UNO.inf" on the PC. There are no other driver files required and this should work for Windows XP, Vista and 7.

You can download this file from the WSDL website at this URL:

http://wmrx00.sourceforge.net/Arduino/ArduinoUNO.inf

Go ahead and plug in the USB cable now. When you first apply power, the red and green LEDs on the shield will flash for about one second. This lets you know that the Arduino "sketch" (a.k.a. software) is running.

## Installing the Windows Driver

One of three things will happen when you first connect the WxShield's USB cable to your computer:

1. If you have previously installed drivers for Arduino UNO, a new COM port number will be assigned to the WxShield and you are ready to go.

2. You will be presented with the "New Hardware Found" dialog.

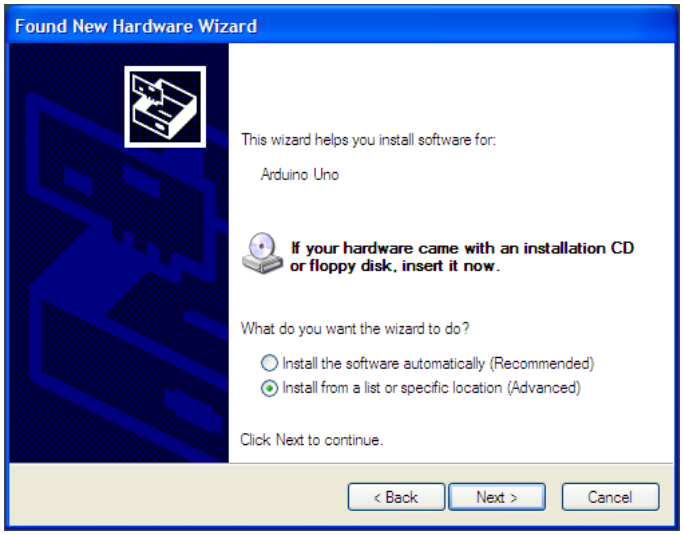
3. Windows will seemingly fail to install the drivers and present you with various error messages. This is more likely to happen with Windows Vista and Windows 7.

Below, we walk you through the process for case 2 above.

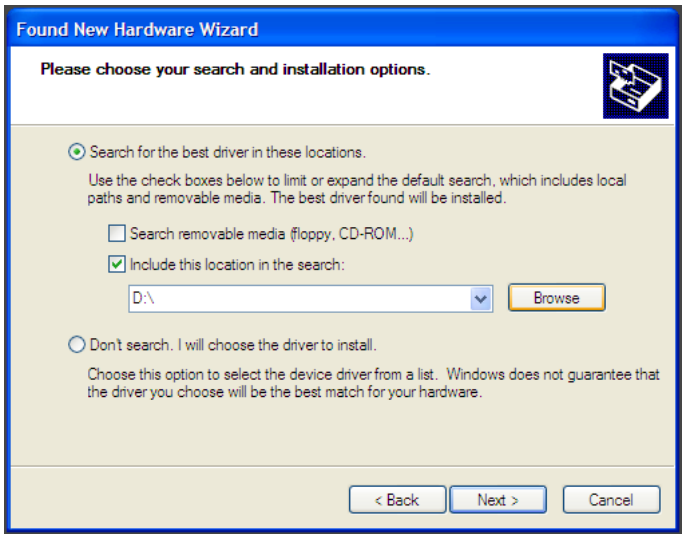
If you get errors at first instead (case 3) then first open the Device Manager. Locate the Arduino UNO device (it may be hiding under the "Ports (COM & LPT)" sub-category). Right-click on the UNO and select "Update Driver…" and proceed with the following steps.

In the first dialog, tell Windows not to look on the internet and click "Next".

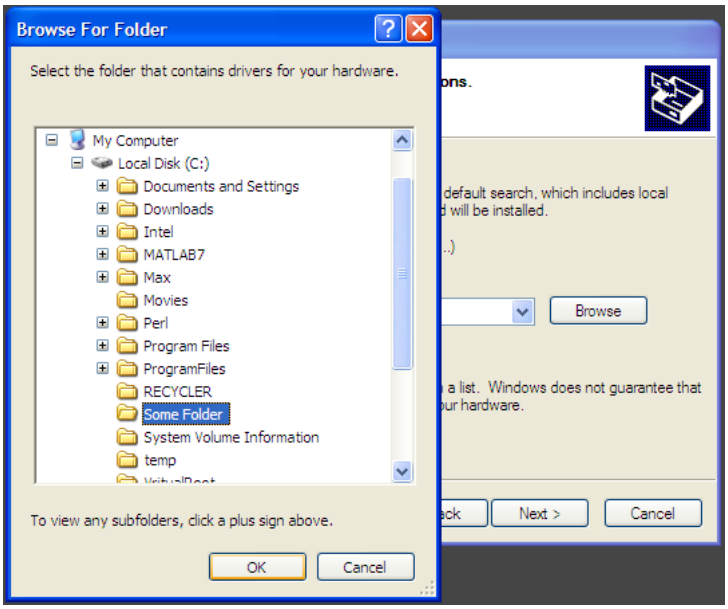

Select the option to install from a specific location and click “Next”.



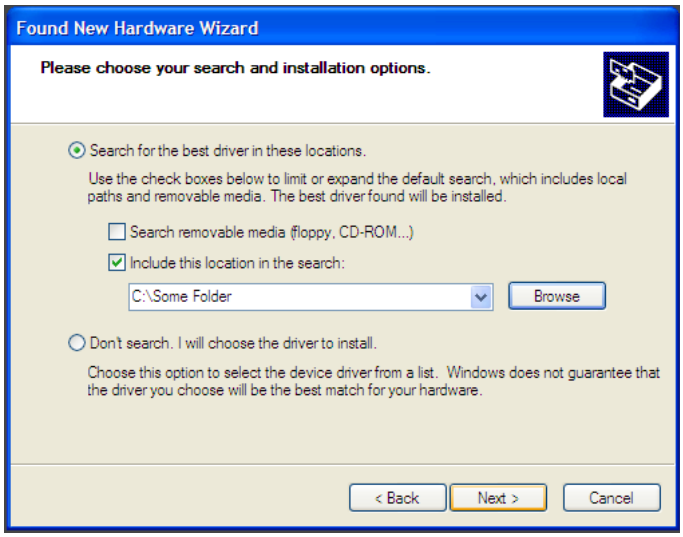

Uncheck the option to search removable media and check the option to include a specific location in the search. Then, click the “Browse” button.




For this example, the “Arduino UNO.inf” file has been placed in the directory “C:\Some Folder”. In the file browser that appears now, navigate to the folder that contains the “Arduino UNO.inf” file. You will not actually see this file displayed but you still need to navigate to that containing folder. Then click the “OK” button.

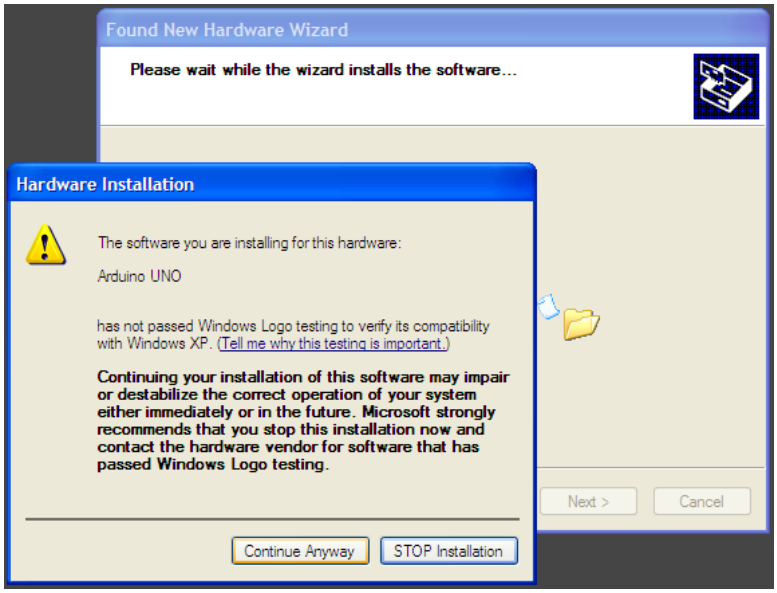



At this point, you will see the folder you selected displayed in the text box next to the “Browse” button. Make sure this is the correct folder and then click “Next”.

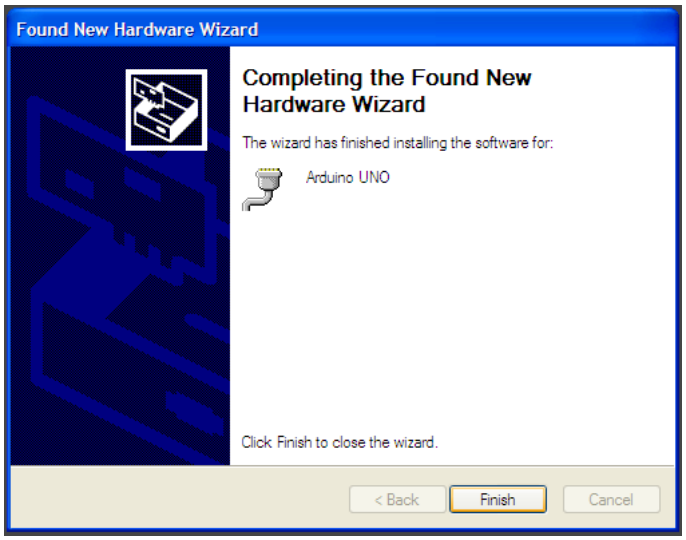



If you have selected the correct folder, Windows will begin the driver installation process now.

The next thing you will see is this lovely little dialog, warning that you are about to destroy your computer! Not to worry – this is just Microsoft trying to shake down hardware vendors for some protection money. You can safely ignore this warning. Click the “Continue Anyway” button and life will be good.

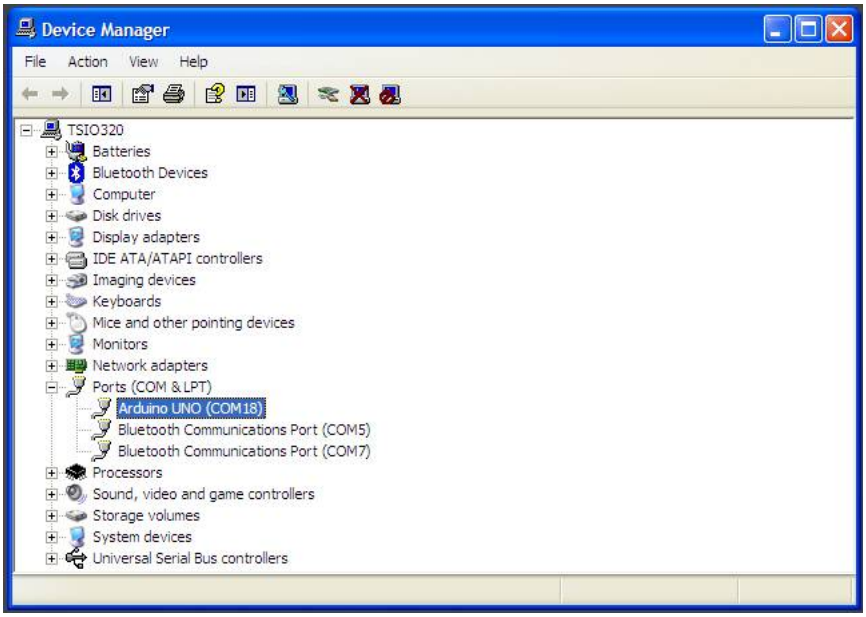



The installation should then finish, presenting you with the final dialog window.



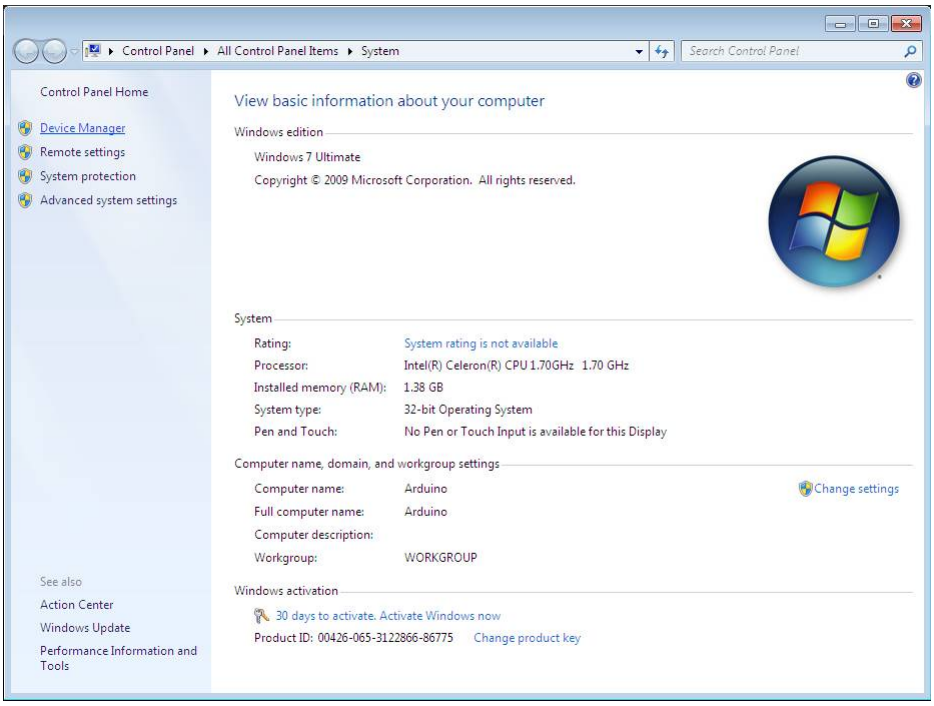

The last thing you need to do is to open the device manager and locate the UNO device as shown here. Make a note of the COM port number (COM18 in this case). As shown below, the UNO will show up in the device manager's category of "Ports (COM & LPT)" – click on the plus sign next to the category to expand it.

You will need this number when setting WSDL options for your Arduino hardware.

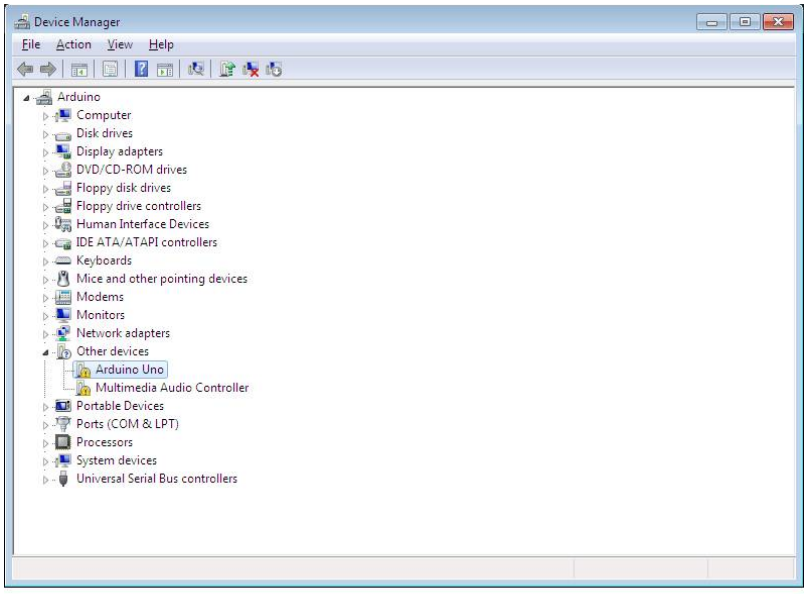


# Windows 7 Driver Installation

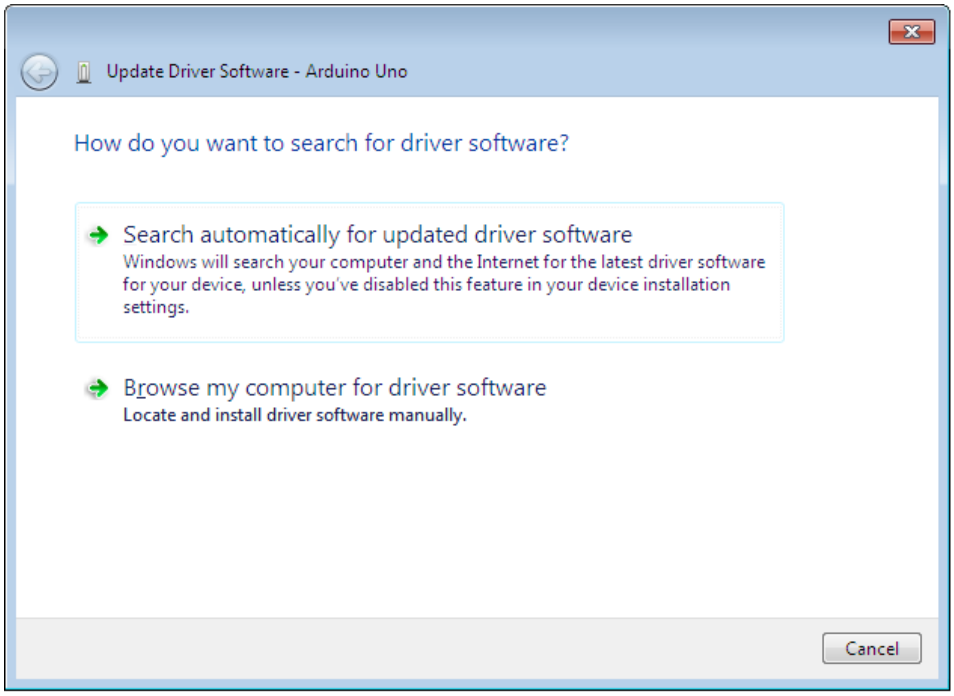
The dialogs for Windows 7 are a little different for driver installation. Below is a terse guide through this process. Start by selecting the “System” item in your Control Panel. Then click on “Device Manager” (as highlighted below).



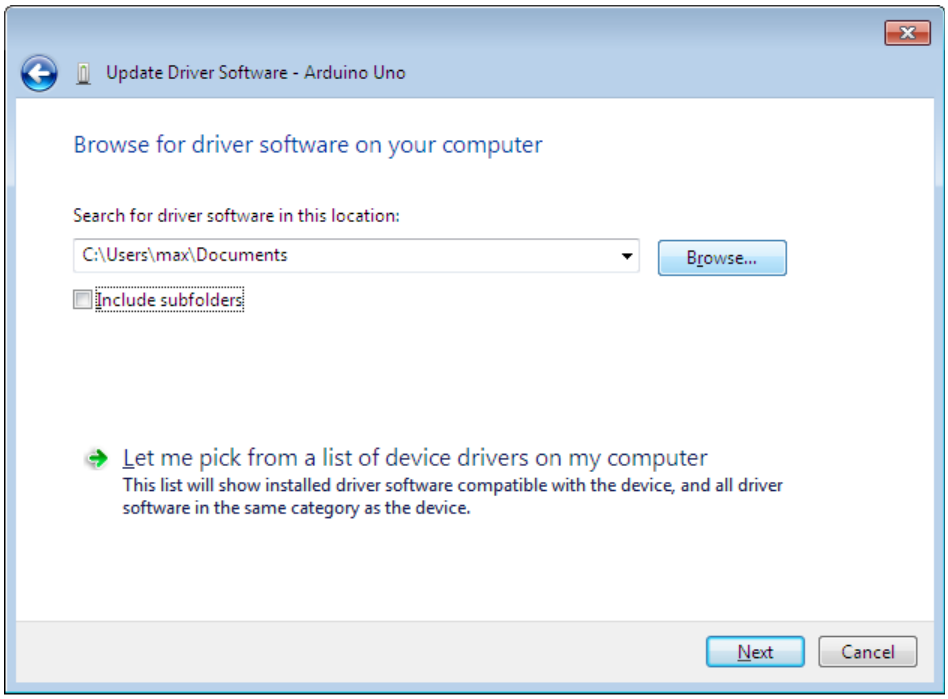

Locate the “Arduino UNO” entry as shown below. Right click on it and select “Update Driver Software”.



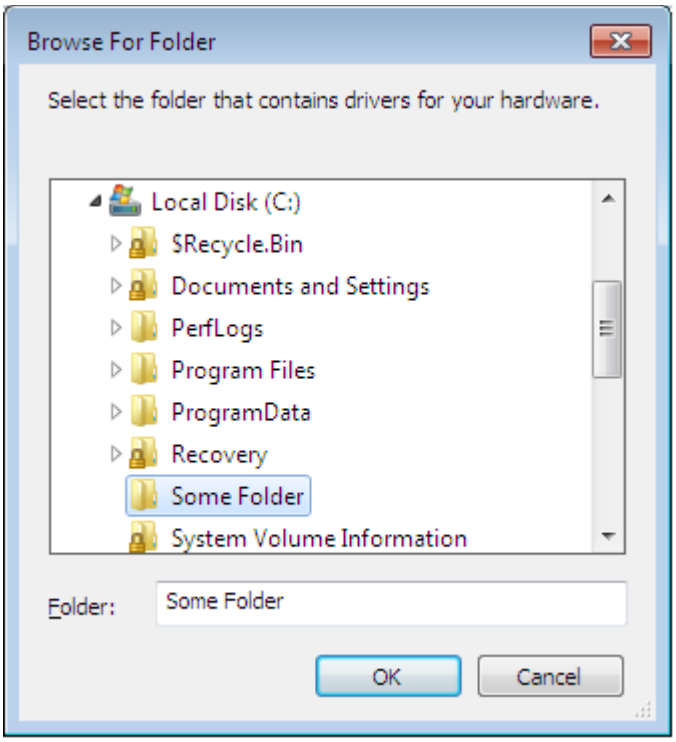
In the next dialog window, click on “Browse my computer…”



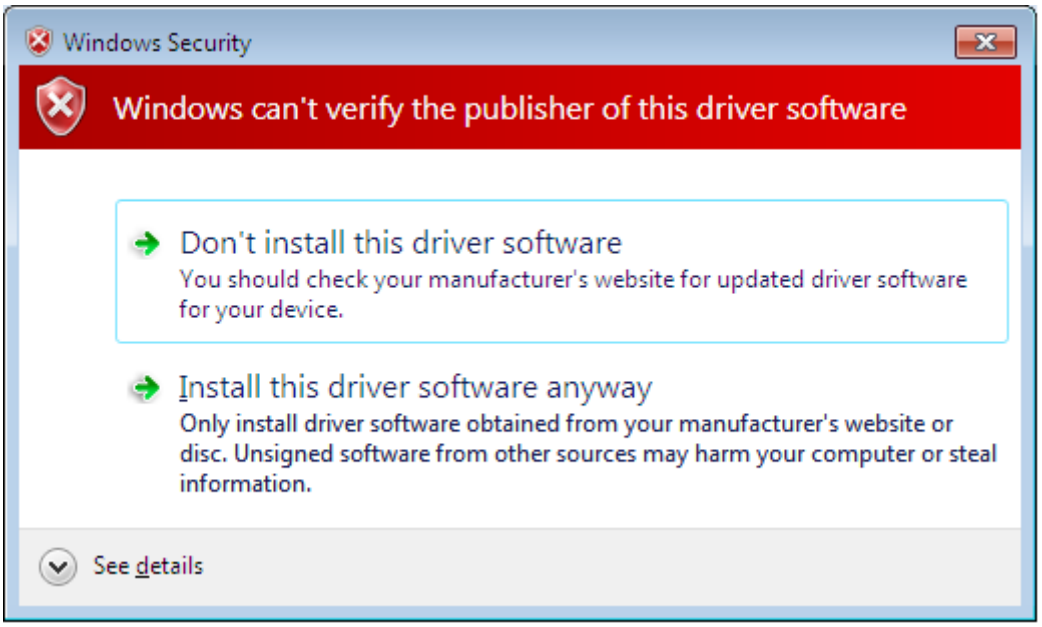

In the next window, un-check the “Include subfolders” option and then click on the “Browse…” button.



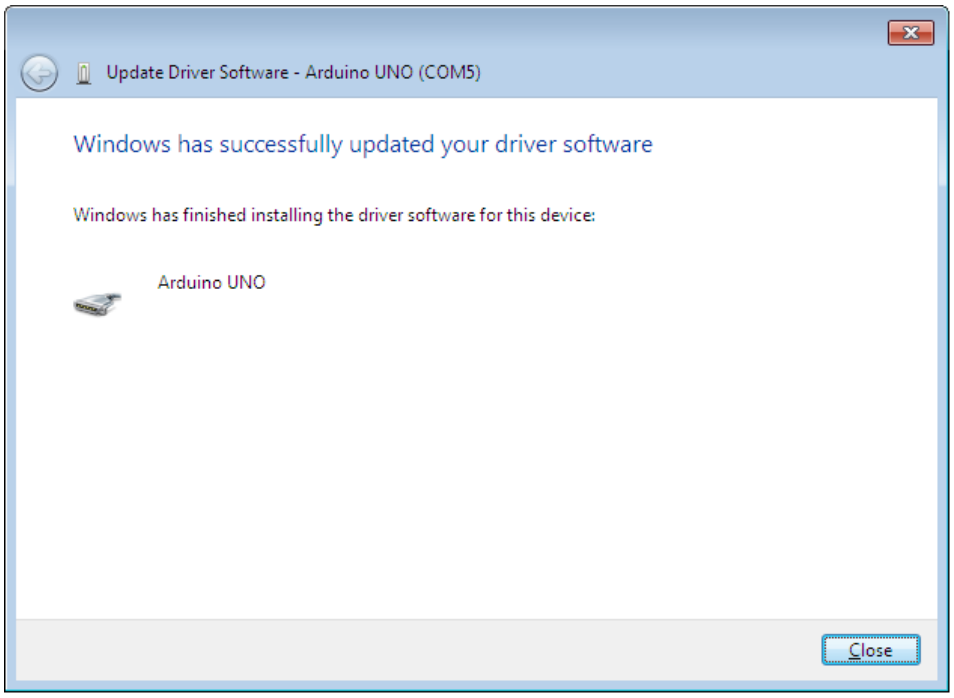
In the next window, browse to find the folder containing the “Arduino UNO.inf” file. In this example, the file is contained in the folder “Some Folder” on the C: drive.



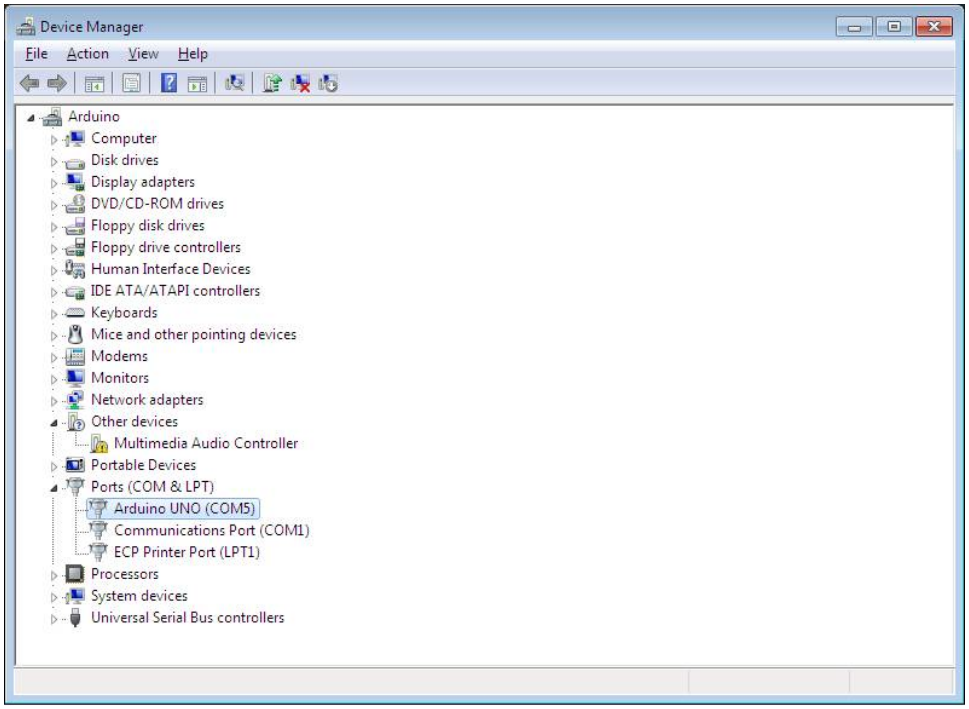

After clicking “Next” this window will appear. Again it is only a feint by Microsoft, so click on “Install this driver software anyway”.



If all goes well you should be presented with the successful installation window as pictured here.

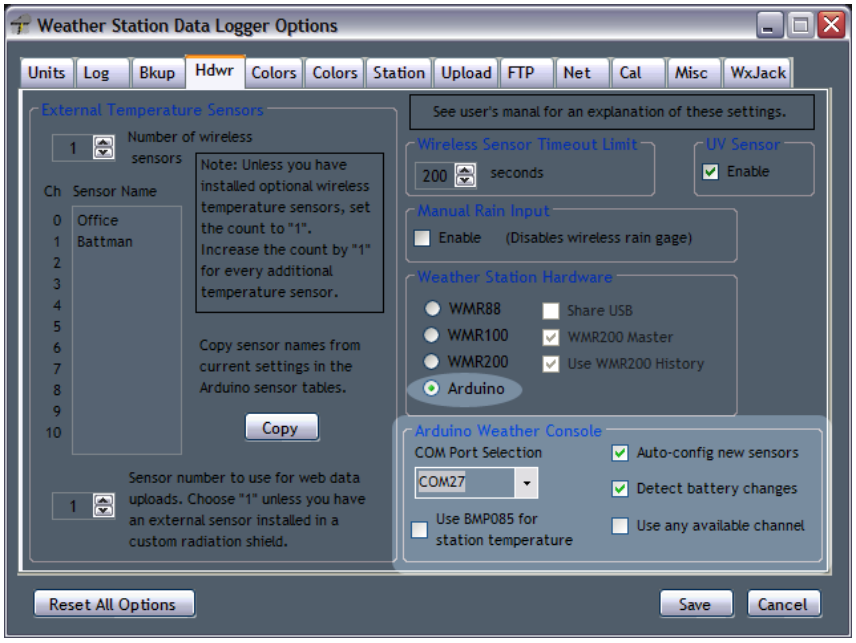



Close this window and look in the Device Manager again. Your Arduino UNO should now appear with a COM port number as shown here (COM5 in this example).



# Configuring WSDL

Start WSDL, open the option window and click on the “Hdwr” tab. Select the Arduino option button, which will activate several options for your new hardware. These new options are in the outlined box in the lower right corner of the window labeled “Arduino Weather Console”.




Select the Arduino COM port from the drop-down list. If the COM port you identified in the list is not available, you can also type in the value (such as “COM18” or “COM27” for example).

There are four check boxes that also need to be configured here. The first one will use the temperature reading from the Bosch BMP085 barometer if you do not have the optional Sensirion SHT15 remote temperature/humidity sensor. Be aware that the BMP085 temperature can read quite a bit higher than the actual room temperature due to heat generated by the WxShield. This is especially true if the WxShield is installed in a closed box (as recommended).

The other three settings depend on the array of wireless sensors you will be using. Determine which of the following scenarios describes your suite of sensors.

1. Standard network:
    a. All sensors transmit the same Oregon Scientific RF protocol (either 2.1 or 3.0, but not both), AND

b. The channel switch settings on all of your wireless temperature sensors are all different (no overlapping settings), AND

c. You have no more than one UV sensor.

2. Custom network:

a. You have a mix of different RF protocols (2.1 AND 3.0), OR

b. Channel switches on two or more of your wireless temperature sensors are set to the same number, OR

c. You have two or more UV sensors.

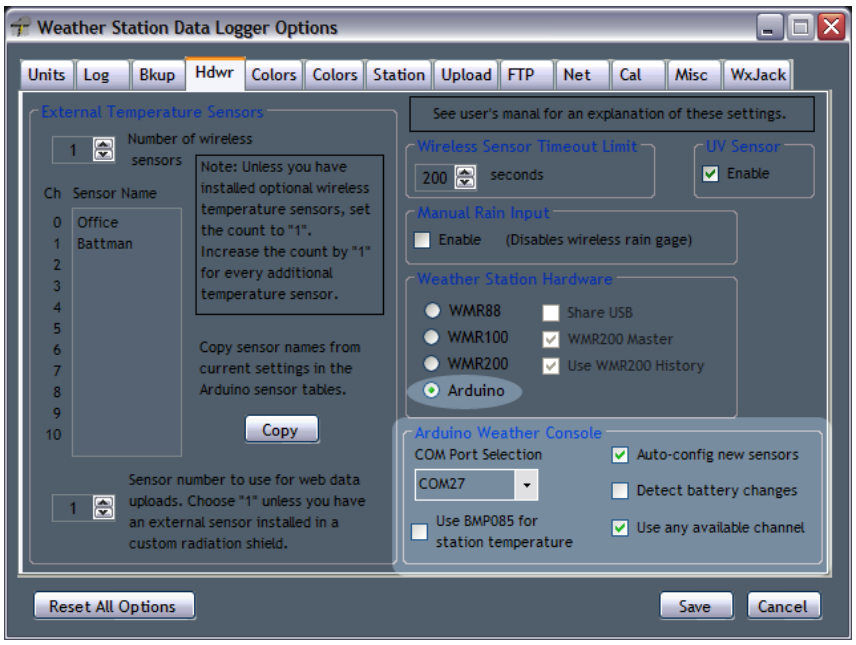
## Standard Sensor Networks

In this case the options to auto-configure new sensors and detect battery changes should be checked. Do not check the option to use any available channel. The option window screen capture above shows this setup.

This type of network is easy to manage and you won't need to learn a lot about wireless sensor management. You can change as many batteries at the same time as you want. That is, unless you start picking up a neighbor's wireless sensors; this problem is discussed below.

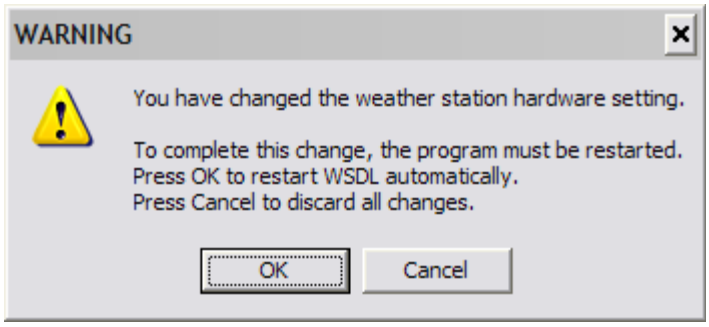
## Custom Sensor Networks

The options to auto-configure new sensors and use any available channel should be checked and the option to detect battery changes should be un-checked. This is a more complex scenario which is described in more detail later.

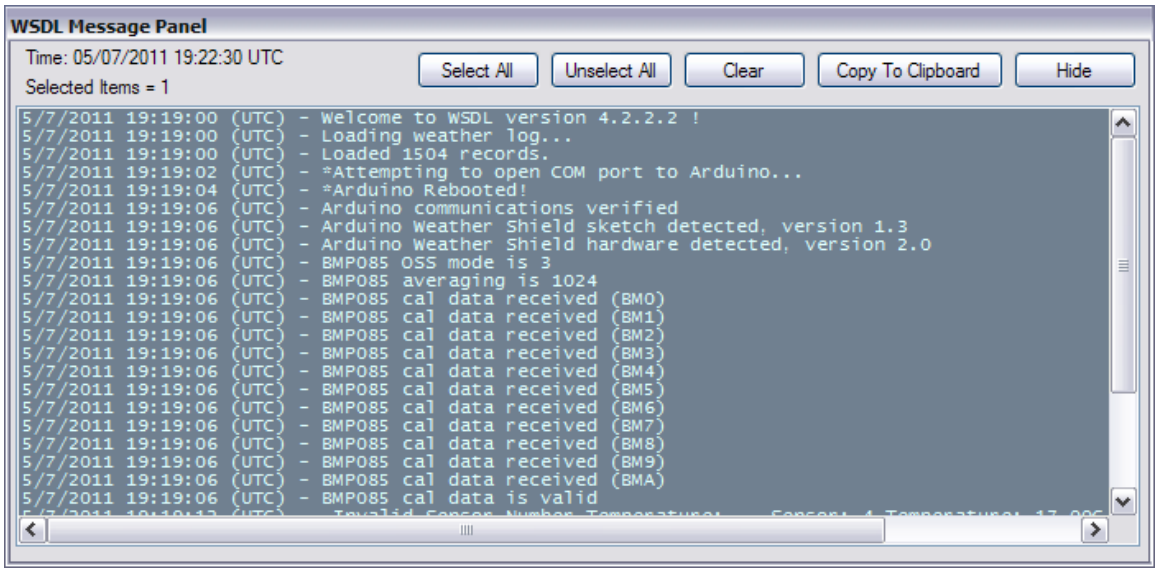


After all of your sensors have been configured, the option to use any available channel can be turned off and detection of battery changes turned on.

After making the desired settings, clicking the button to “Save” options will cause WSDL to restart. You should see the following dialog warning at this point. Click OK and WSDL should exit and automatically restart.




When WSDL restarts open the message panel (click on View…Message Panel in the main WSDL window) and you should see messages similar to the following.

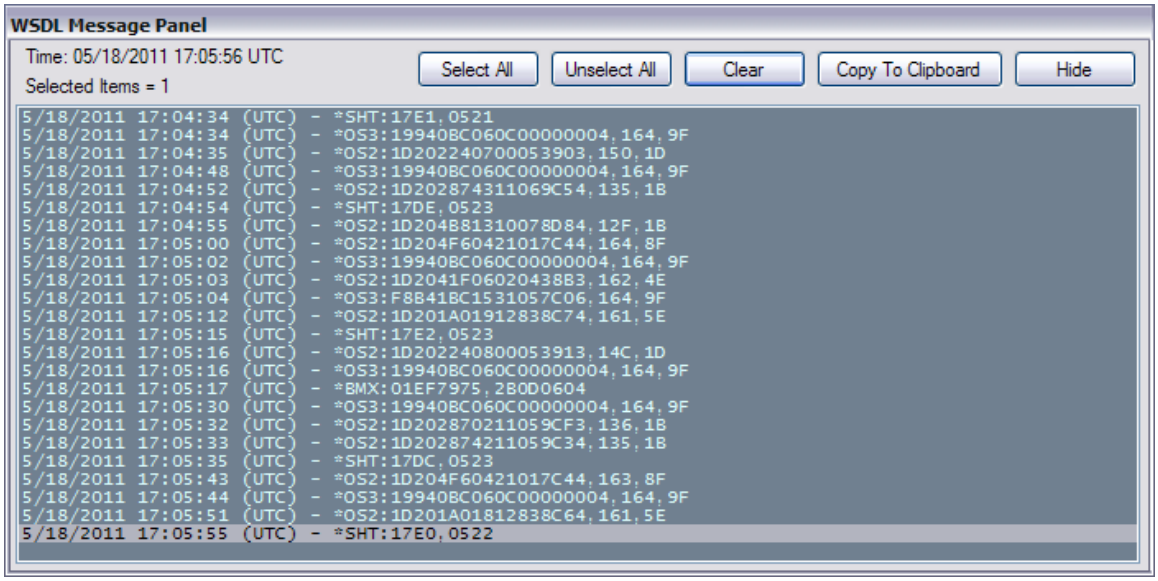



In the main WSDL window, select “View…Enable…Undecoded Messages”; make sure that option is checked. Go back to the message panel and let this run for a while. You should see additional messages appear. Look for at least one each of the following message types.

- OS2:/OS3: These are received and partially decoded messages from Oregon Scientific wireless sensors. The difference between OS2 and OS3 is the RF protocol version of your sensor. For now, just confirm receipt of at least one of these message types.

- SHT: This is data from the SHT15 remote sensor board. If you're not getting that and just realized you forgot to plug in the remote board – wait! Be sure to power down the WxShield before plugging in the remote board.

- BMX: This is temperature/pressure data from the Bosch BMP085 barometer. These messages only appear about once a minute so be patient.



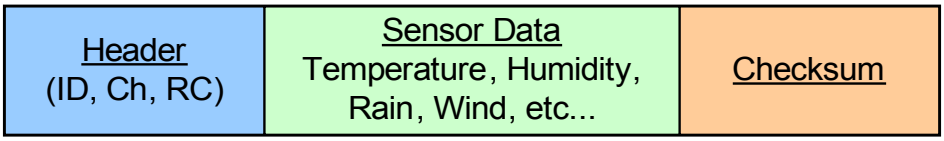

If you are receiving all three types of messages, then all three sub-systems on the WxShield are operational.

What comes next depends on your sensor array.

# Sensor Identification

Channel switches on your wireless sensors allow no more than ten different settings. In reality sensors can effectively have anywhere from 256 to 2,560 “virtual” channels. To understand how this is possible, a discussion of the data transmitted by sensors will help.

The drawing below depicts the information contained in every transmission from version 2.1 and 3.0 wireless sensors. Version 1.0 sensors do not include the “ID” field in the header.




All transmissions begin with a “header” which contains identifying information about the sensor. This is followed by the current weather data measured by the sensor. Finally a “checksum” is transmitted which provides a check on the integrity of the entire message. The checksum helps to identify garbled messages so they can be discarded.

For the purposes of this discussion, we are only concerned with the data contained in the header portion of the message. There are three different numbers contained in the header:

- An “ID” number which identifies the sensor’s model. This is an integer value between 0 and 65,535. In some cases this value is unique to a particular sensor model. For example the PCR800 rain gauge is identified by the value 10,516. In other cases, several different sensor models may share the same identification number (e.g. the THGN123N and THGR122NX both share the number 7,456). WSDL’s Arduino Sensor Manager displays these numbers in hexadecimal format, so the value “7,456” will be shown as the hexadecimal value “1D20”.

- A “Ch” value that depends on the channel switch setting. With some sensors this number is the same as the channel switch setting. For example when the channel is set to “7”, the number “7” is also transmitted. For other sensors, this number may be different than the channel number selected on the channel switch. For example, channel switch setting “1” may be sent as “1”, while selecting channel “3” causes the number “4” to be sent.

- A special “RC” number which stands for “Rolling Code”. This is an integer value between 0 and 255. This number allows WSDL to create far more “virtual” channels than the actual number of channel switch settings. The next section explains how this works. WSDL displays rolling codes in hexadecimal format. For example, the decimal value “128” will be shown as hexadecimal “80”. In version 1.0 sensors, the rolling code is a number between 0 and 16.

## Rolling Codes

When batteries are first installed in a sensor, the sensor will set its rolling code to a random integer between 0 and 255. If batteries are changed, the sensor will once again pick a (usually different) random value for the rolling code. Every time the sensor’s reset button is activated, the random code will change to another (usually different) random value.

There is a 1-in-256 chance that after a battery change your sensor will choose the same rolling code again, so this will not happen very often. But it will happen on occasion, and you need to remember this.

The rolling code was probably added to allow two neighbors to operate OS weather stations without interfering with each other. As long as their individual consoles (such as the WMR100) keep track of the rolling code for their own sensors, the neighbors’ sensors will not get mixed up.

As mentioned above, there is a 1-in-256 chance that two neighbors’ sensors will wind up with the same identical rolling code. When this happens, there is no way to tell them apart and one neighbor will need to reset their sensor for a new rolling code.

These rolling codes are the mechanism that allows WSDL and the WxShield to work with multiple sensors set to the same channel number. An example will help to understand how this works.

Consider two THGR122NX sensors that are both set to channel “1”. The header from one sensor will contain the sensor’s ID (hexadecimal “1D20”), the channel number (“1”) and a random rolling code (e.g. “3F”). We’ll represent this sensor’s header data with the notation “1D20,1,3F”.

The header from the second THGR122NX sensor will contain the same ID and Ch values, but the rolling code will typically be different – something like this: “1D20,1,B3”.

WSDL is then receiving transmissions from a sensor with the header “1D20,1,3F” and another sensor with the header “1D20,1,B3”. Clearly there is no problem telling these two sensors apart.

Since there are 256 different possible rolling codes it is theoretically possible to have 256 different THGR122NX sensors all set to channel 1 and we can still tell them apart. This turns out not to be practical for several reasons, but having three of these sensors on the same channel setting is not a problem.

What makes this a bit more difficult is the fact that sensors do not display their rolling codes in the little LCD window.

During the initial configuration of WSDL, you must determine which rolling code goes with which sensor for those sensors that share channel numbers. After this initial setup if you are careful to only change batteries (or reset) one sensor at a time, WSDL can keep track of things without further assistance.

The easiest way to figure out the rolling codes is to remove the batteries from all sensors which have channel number conflicts. Then install batteries in one sensor at a time and watch for WSDL to pick up the new sensor.

## Sensor Management

It should be clear now that in order to keep track of wireless sensors, WSDL needs to associate the header data (ID, Ch, RC) for each sensor with a WSDL channel number (even with all these virtual channels, there are still only ten WSDL temperature/humidity channels).

This is done by creating two tables. The first table assigns a “serial number” to each sensor’s header information (ID, Ch, RC). The figure below shows how the table works.

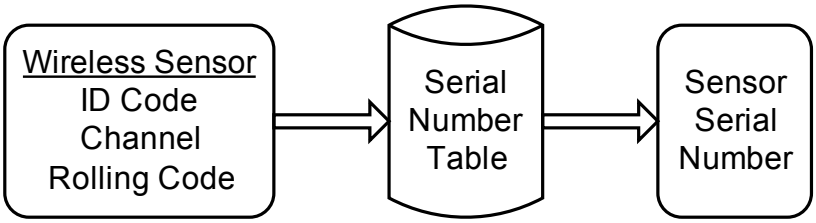



**Figure 1. Sensor Management: Serial Numbers**

A second table is used to connect a particular physical sensor (identified by serial number) to a WSDL channel. This table is called the Sensor Information Table.

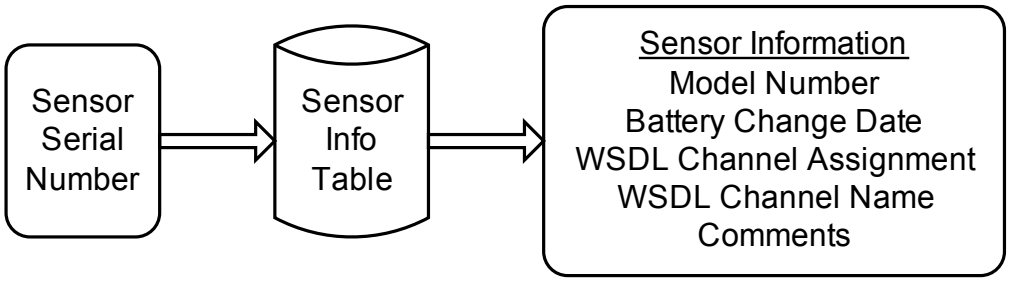



**Figure 2. Sensor Mangement: Sensor Information**

In addition to assigning a WSDL channel number to the sensor, this table keeps track of the sensor's model number, battery change date, channel name and has room for miscellaneous user comments.

## The Arduino Sensor Manager

Both of the tables discussed above can be viewed and edited with the Arduino Sensor Manager (click Tools…Arduino Sensor Manager in the main WSDL window).

Users with standard sensor networks will typically not need to use this tool.

## The RF Codes Table

When WSDL receives a wireless transmission, it searches this table for the sensor's header information. If the sensor's header is not in the table it will be added automatically. The serial number will either be set to "-1" (disabled) or a valid serial number depending on the auto-config option settings.

Users cannot add entries to this table, but can delete entries and change serial numbers.

In reality, WSDL needs to receive two transmissions from a new sensor within a short period of time before it is added to the RF Codes Table. This prevents garbled messages from masquerading as new sensors.

## The Sensor Information Table

Two conditions must be met before a sensor's data will be accepted by WSDL:

1. The sensor's header must be found in the RF Codes table and the serial number must be positive.

2. There must be an entry in the sensor information table with a matching serial number.

The matching entry in the sensor information table is then used to determine the WSDL channel number for display and logging of sensor data.

## Channel Numbers

You may be used to thinking of channel numbers for wireless temperature sensors, but other sensors (like a UV sensor for example) also transmit channel numbers. These cannot be changed but they are still part of the sensor's header data.

The sensor information table also contains WSDL channel number assignments for sensors like rain gauges. Normally, these should be set to channel "1" for sensors other than temperature and/or humidity. The only exception is with UV sensors – if you are using two or more UV sensors, each sensor should have a different WSDL channel number.

## Automatic Sensor Configuration

You should have enough background now to understand how sensor configuration is performed by WSDL. First, consider a standard network where auto-configuration is enabled. WSDL will start with an empty sensor database.

Assume there is only one sensor (e.g. the anemometer). The first time WSDL hears the anemometer it makes a note but does not do any configuration. The second time however, the anemometer's header is added to the RF Codes table. Since auto-config is enabled, it will do two additional tasks.

1. Assign the lowest available serial number to the anemometer in the RF Codes table.

2. Add an entry to the Sensor Information table with the same serial number. The WSDL channel number will be set to "1" since this is an anemometer.

If a different anemometer is discovered, it will be added to the RF Codes table with a serial number of "-1" and the Sensor Information table is not modified. This is because WSDL can only handle data from a single anemometer.

Now, say two transmissions are received from a wireless temperature/humidity sensor with the channel switch set to "3".

1. The sensor is added to the RF Codes table using the lowest available serial number.

2. An entry is added to the Sensor Information table for the new serial number and the WSDL channel number is set to "3" – the same channel number that is set on the sensor's channel switch.

There is one caveat here however. Some sensors (e.g. THGR122NX) transmit a channel number in the header that is different than the number on the channel switch. Channel numbers {1,2,3} might transmit as {1,2,4} for example. As a

result, the WSDL channel number assigned during auto-config may be different than expected.

Any additional temperature sensors are added in a similar manner. Since this is a standard network, there will be no conflicting sensor channel numbers.

## Auto-Configuration in a Custom Network

Custom networks provide a great deal of flexibility in use of sensors with differing RF protocols and conflicting channel settings. The price to be paid for this is added complexity in configuring your sensors. It is necessary to understand more about your sensors' RF transmissions.

What happens in the previous example if two temperature sensors have the same channel number? When the second sensor is discovered, WSDL detects the channel number conflict. As a result, the second sensor will be assigned a serial number of "-1".

That is, unless the hardware option to "Use Any Available Channel" is turned on. In this case, WSDL will assign the lowest unused WSDL channel number to the new sensor and add an appropriate entry to the Sensor Information table.

This makes it easier to get an initial configuration setup for a custom sensor network, the actual channel assignments are not predictable and the user must still determine which sensors have been assigned to which channel.

This initial setup of a custom network is the most difficult part. This task is easier if you pull the batteries from all of your temperature sensors before beginning the configuration task. Then, install batteries in on sensor at a time and wait for that sensor to be auto-configured. This way there is no doubt as to which sensor is which.

## Changing Batteries

When you change sensor batteries, the sensor's RC usually changes (there is a 1-in-256 chance that it won't change). When this happens, WSDL will think this is a new sensor (not the old sensor with new batteries). The new sensor's header will be added to the RF Codes table and it will be auto-configured (if possible) based on option settings. This is probably not what you want.

That's why there's an option to detect battery change events in the options window. Here's how a battery change is handled if the auto-config and battery change options are enabled and the use any channel option is disabled.

1. A new sensor is discovered (requires 2 transmissions).

2. WSDL checks to see if there are any other sensors with the same ID and Ch values but a different RC. For each such (possibly old) sensor,

3. WSDL looks at the last time a message from the (possibly old) sensor was received, and the first time the new sensor was heard from.

4. If the (possibly old) sensor has not been heard since the new sensor appeared, it is assumed that the new sensor is the same as the old sensor after a battery change.

When a battery change is detected, the old sensor's serial number is changed to "-1" in the RF Codes table and the new sensor is given the old sensor's original serial number. The battery change date for that sensor in the Sensor Information table is also updated.

WSDL only performs these checks every couple of minutes, and it can take up to 8 minutes or so for a battery change to be fully processed. If you change batteries in more than one sensor at a time, and those sensors have overlapping channel settings then WSDL may get confused. This can result in sensors getting swapped to different WSDL channels.

As a result, it is recommended you only change one set of batteries at a time and wait for WSDL to process the change before changing more batteries. You can see the change happen by watching the RF Codes table for changes in the Arduino Sensor Manager. Be sure to hit the refresh button once in a while as this window does not update automatically.

## RF Transmission Intervals

Each sensor transmits data about once a minute. It is easy to realize that if all sensors transmitted data at the same interval (say 60 seconds for example), you could wind up with a situation where all sensors were talking at the same time and at best only one sensor (the loudest one) could be received.

OS solves this problem by changing the transmission interval as a function of the channel number setting. For example, a sensor set to channel 1 might transmit every 49 seconds while another sensor on channel 2 might send data every 57 seconds. This way, even if sensor transmissions collide with each other, this only happens once in a great while.

The above discussion should make it obvious that problems can occur when two sensors are set to the same channel number. Although WSDL can tell the sensors apart by examining their rolling codes, these sensors are sending data at the same intervals.

How much of a problem this is depends on how accurately each sensor defines its transmission interval. Each sensor has an internal clock which is used to generate the transmission interval (of say 57 seconds for example). These internal clocks are not perfectly accurate however, so one sensors might transmit data every 56.994 seconds while a different sensor is sending data

every 57.007 seconds. Furthermore, these internal clocks tend to drift as the temperature changes so on a colder day, the sensors might be sending data every 56.998 and 57.015 seconds respectively.

Transmissions from version 3.0 sensors last about 0.1 seconds and version 2.1 sensor transmissions last about 0.4 seconds.

As a result of these sensor timing issues, two sensors with the same channel number can wind up talking on top of each other more frequently than when using different channel numbers. This can manifest itself as a period of time (lasting as long as several minutes) where either one or both sensors go missing, even though both appear to have good signal strength. It is even possible (although unlikely) that this time period could last half an hour or more.

If you notice that one or two sensors appear to drop out every now and then and they are both set to the same channel number, this may be the explanation.

## Version 1.0 Sensor Differences

As noted earlier, version 1.0 sensors do not include an ID field in the message header. As far as is known, all version 1.0 sensors are temperature-only units and all these messages contain the same data in the same format.

There is only a single nibble (4-bits) in the rolling code for these sensors so there are only 16 possible values instead of 256. Furthermore, at least some of the version 1.0 sensors do not always change the rolling code when reset or when batteries are changed. It can take several reset operations before the code changes. Trial and error is the best approach here.

# Barometer Accuracy and Offset

The Bosch BMP085 barometer has a specified absolute accuracy of +-2.5mb. Although Bosch also quotes a "typical" accuracy of +-1.0mb this is not guaranteed and should not be relied upon. Furthermore, Bosch states that the process of soldering the barometer typically results in an additional offset of around 1mb.

This means that uncorrected pressure readings from your barometer can be in error by as much as 3.5mb. A large majority of the WxShields tested to date seem to have an offset of 2.5mb or less, although our measurements are in no way guaranteed.

That's not the end of the story, however. The more important specification is the one called "relative accuracy". Let's say a particular barometer has an offset of +2.1mb when the true station pressure is 1012.0mb. The barometer will be reading 2.1mb higher (or 1014.1mb) and if we subtract the offset (2.1mb) then we get the true station pressure (1014.1 - 2.1 = 1012.0).

Now what happens tomorrow when the true station pressure has dropped to 999.0mb? We are of course tempted to say that the barometer will read 2.1mb higher than this, or 1001.1mb. In fact this is not generally the case – the barometer might for example actually read 1001.12mb. Now if we subtract the 2.1mb offset from this we get 999.02mb so our error is now 0.02mb.

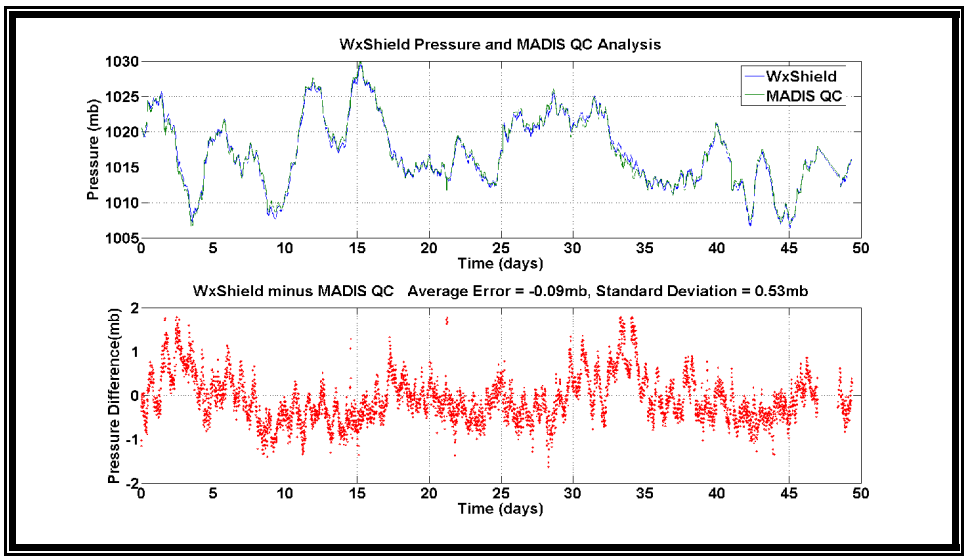
The above example illustrates what is meant by relative accuracy. If you remove the offset at one pressure reading, relative accuracy specifies how much the barometer can be off when the pressure changes. The BMP085's relative accuracy is +-0.2mb and this is valid over a pressure range from 700mb all the way to 1100mb. In other words, if an offset is measured at one pressure (say 1012.0mb) then the barometer will be in error by no more than 0.2mb at any other pressure between 700mb and 1100mb.

There is one other detail -- this particular relative accuracy specification only applies if the barometer is kept at a constant temperature (25C). If the barometer is being subjected to large temperature swings, then another relative accuracy specification comes into play, which is +-0.5mb (for any temperature between 0C and 60C). For many WxShield users, the indoor temperature will be relatively constant and this specification will not cause too much trouble.

Each WxShield is fully tested, and a suggested barometer offset is recorded on the test slip attached to the anti-static bag the WxShield is shipped with. We hope these offsets are close to the right value, but they are in no way

guaranteed to any level of accuracy. Please treat the suggested offset as only a suggestion.

Below is a plot showing the MADIS QC analysis of one WxShield's BMP085 barometer over a period of about 12 weeks. The average error is only 0.09mb (or 0.0025inHg) and the barometer has not required periodic tweaking after the initial offset was determined.




The bottom plot shows the difference between the WxShield's barometer and MADIS QC analysis values. Although the average is nearly zero, there are significant excursions (as much as 2mb) from zero. There are most likely two factors at work which cause this behavior.

1. Wind. Whenever wind is blowing it is due to pressure differences between where you are and where the wind is coming from (and where its going to). MADIS may or may not account for this, but even if it does, it will not get it right all the time.

2. Non-standard atmospheric conditions. This particular station's elevation is significantly different than all neighboring stations so MADIS must make some assumptions about the vertical atmosphere profile in order to compare stations. If MADIS gets that wrong, there will be errors. A pressure difference of 2mb occurs with an altitude change of only 20 meters (60 feet) so it does not take much to cause this amount of error.

# Signal Strength

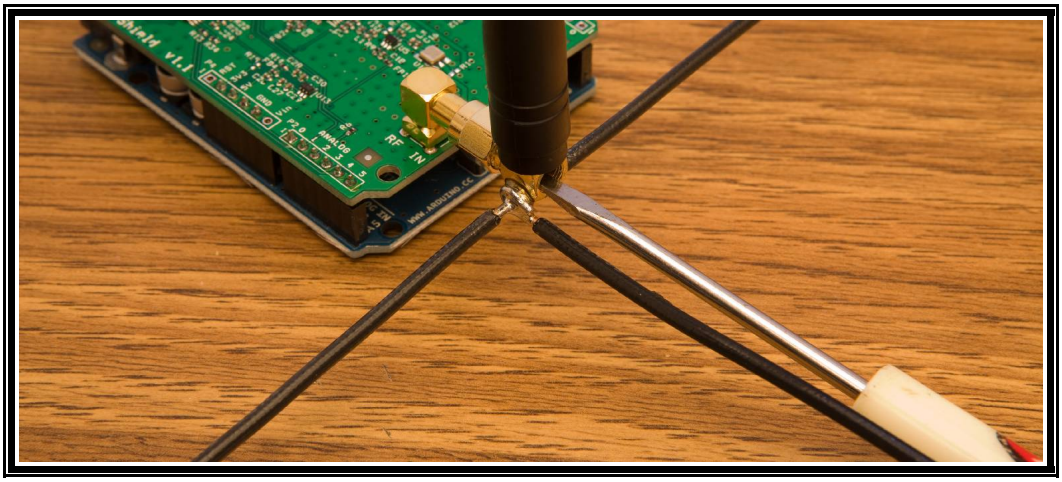
In the main WSDL window, clicking on the “Details” button in the battery status area will bring up a window that includes signal strength information on your wireless sensors.

WSDL displays approximate received power levels in units of “dBm” which is short for “decibels relative to one milli-watt”. See the following Wikipedia article for a brief introduction to dBm:

http://en.wikipedia.org/wiki/DBm

First, a caveat is in order regarding signal strength numbers. The Freescale receiver IC provides signal strength information but the accuracy of these readings is not specified by Freescale and may vary widely from unit to unit. Therefore, comparing signal strengths reported on two different WxShields may not be all that meaningful. These numbers ***are*** useful as a relative comparison for comparing signal strength between sensors and trends over time.

The original antennas we purchased were only $2.20US plus shipping and you get what you pay for. Although the hacked antenna works very well, there is one problem you may encounter and need to be aware of.




The center pin in the right-angle SMA connector is sometimes loose (or can become loose over time). When this happens, the center pin can get pushed inwards and towards the threaded plug at the rear of the SMA connector. This happens when the connector is threaded onto the WxShield and tightened. It sometimes is pushed far enough to short out against the connector body. When this happens signal strengths will drop by about 30dB or so.

If you experience low signal strengths or reception problems, carefully remove the threaded plug at the rear of the connector. The plug is very small so be

careful not to lose it. Use a small tool (e.g. a common screwdriver or the shank of a small drill bit) and firmly push on the center pin to seat it properly in the connector. If that was the problem you should feel it give way and move a little bit.

Afterwards, it is not mandatory to replace the end plug but you may get very slightly better reception with it installed. For more information see the section below on hacking the antenna.

WxShields delivered in late 2012 may have a small round plastic insulator underneath the threaded plug to help with the center pin shorting problem. Be careful not to lose the insulator if it is installed and you remove the plug.

# Updating the WxShield Firmware

From time to time, it may be necessary to update the program code stored in the Arduino UNO board that is part of the WxShield assembly. One example would be to add support for Oregon Scientific version 1.0 sensors. This is relatively easy to do if you follow the steps below. The current shield version's firmware was built using Arduino software version 022. If possible, use this version to build new firrmware versions; if version 022 is not available, a newer version should work too.

1. Go to the Arduino web site (http://www.arduino.cc) and download the version 1.0 software. This is a large file (tens of MB in size).

2. There is no regular installer for Arduino software. Instead the zip file is simply extracted into a new directory on your computer's hard drive. This will create a folder named “arduino-1.0”. From here on out, we'll refer to the “arduino-1.0” directory as the “home directory”. The zip file can be extracted anywhere you like:

   a) On your desktop.

   b) Somewhere in your My Documents folder.

   c) In the Program Files folder (use Program Files (x86) on Windows Vista or Windows 7) if you have a 64-bit system.

   d) Just about anywhere else.

   e) If you already have the Arduino software on your computer, it might be a good idea to extract a second copy into a different directory. This way, if the WSDL source code over-writes any of the built-in libraries this won't cause any problems with your existing Arduino software.

3. Get a copy of the current WSDL WxShield firmware source code. This is also a zip file and will have a name something like this: “WxShield-Source-Version-1.6.zip”.

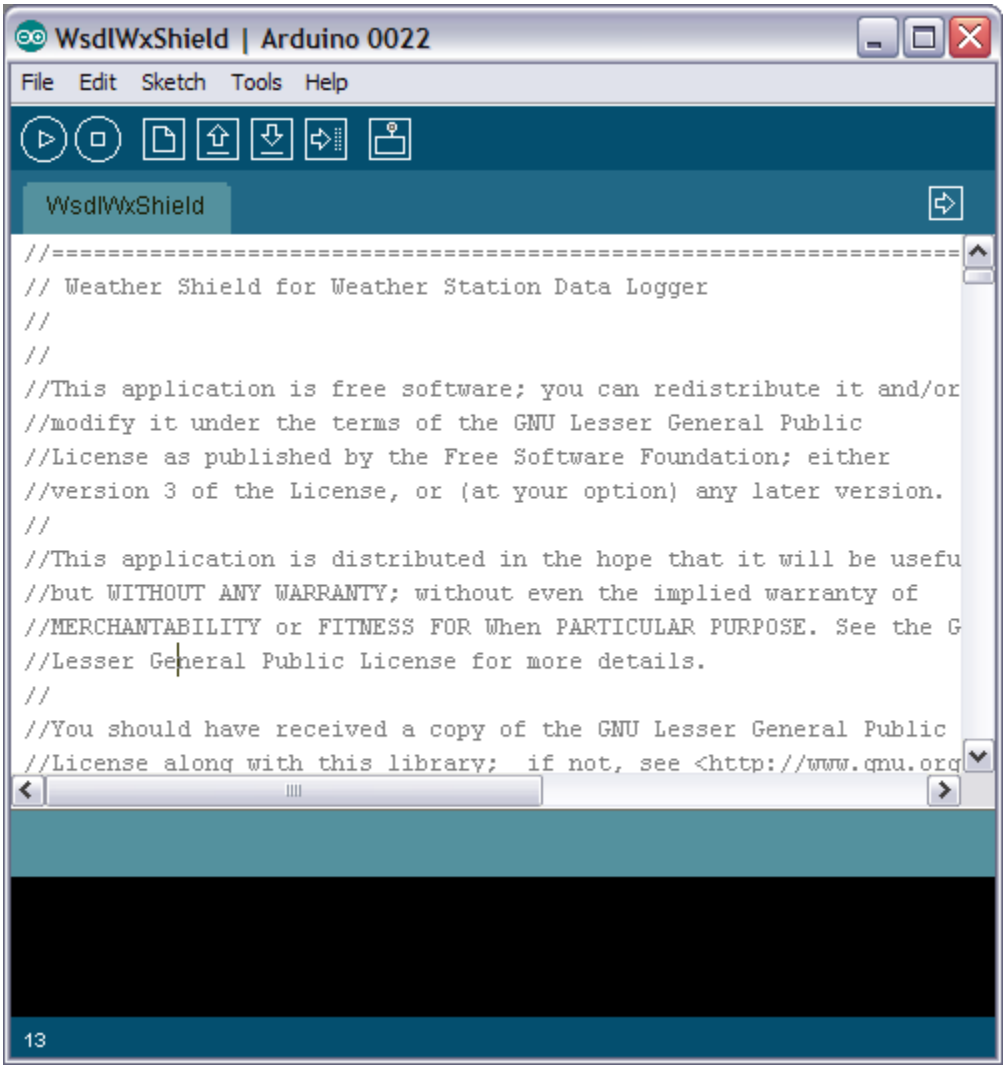
   Place this file into a different directory and extract it there.

4. In the home directory there is an executable file named “arduino” or “arduino.exe” – double-click this file to start the Ardunino software. In the window that comes up, click on File...Preferences. A new window will open; change the Sketchbook location to the directory where you

extracted the WxShield source code zip file. The directory you specify here should have sub-directories named sketches and libraries.

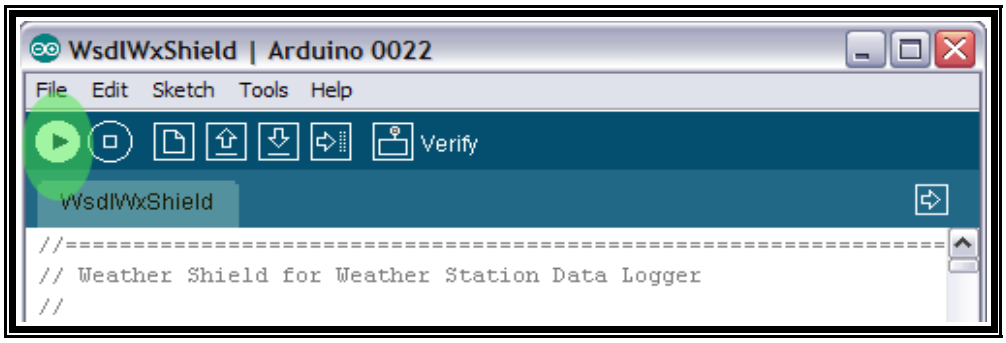
5. Now, click on File...Sketchbook...sketches...WsdlWxShield. If you do not see this in the menu it means the Sketchbook location preference is not set correctly.

   This will open a new Arduino window and the original window can now be closed. This new window should look something like this:

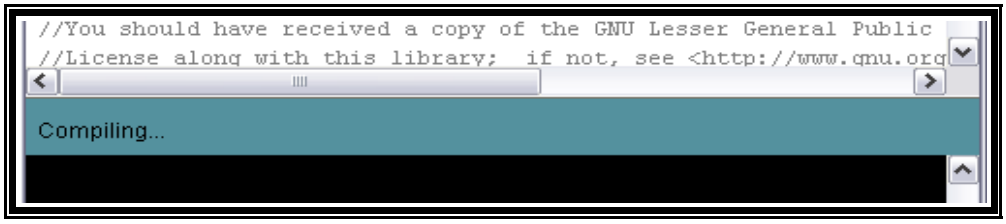



6. Click on Tools...Board... and select the Arduino board you have. The first pre-assembled copies of the WxShield were shipped with the Arduino Uno board. Look on the bottom of your WxShield if you are not sure which board you have.

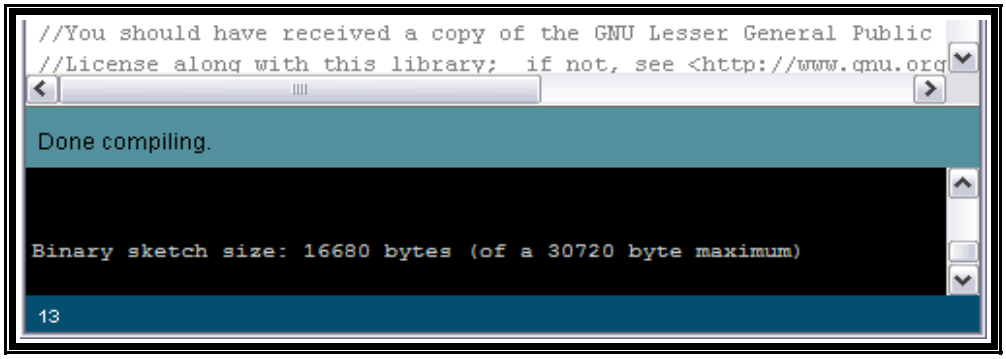
7. Now click on the “Verify” button, which is the one in the upper-left corner and highlighted in green below:




8. This should cause a status message, “Compiling...” to appear near the bottom of the window, like this:

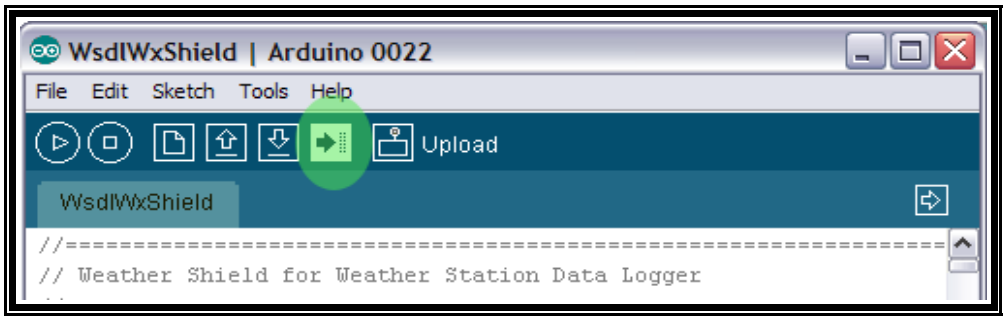



9. In less than a minute the status should change to “Done Compiling”:

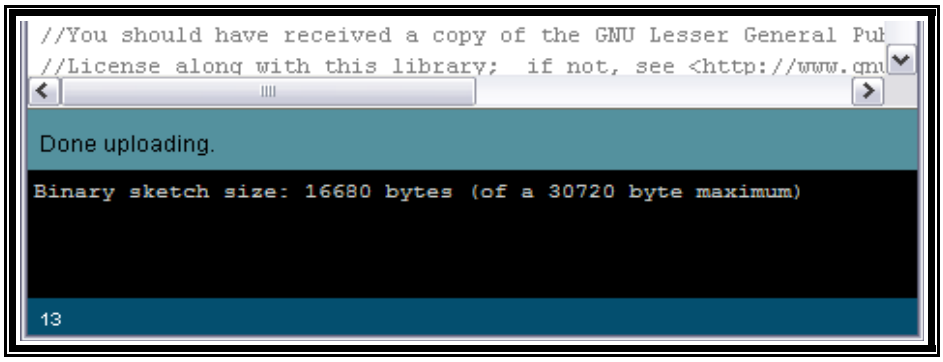



10. This verifies the WxShield sketch is valid and ready to be programmed into your Arduino board. Plug your WxShield into the computer's USB port and make sure that WSDL is NOT running. Now click on Tools...Serial Port... and select the COM port corresponding to your WxShield.

11. Now click the “Upload” button, which is highlighted below in green:




12. This should result in an “Uploading...” status message. The sketch will be compiled again and then uploaded to your Arduino board. When finished you should see something like this:




13. If there are cryptic error messages instead, make sure you have selected the correct Arduino board model and serial port in the Tools menu.

That's it; your Arduino firmware is now updated and ready to connect again with WSDL. Close the Arduino window and open WSDL.

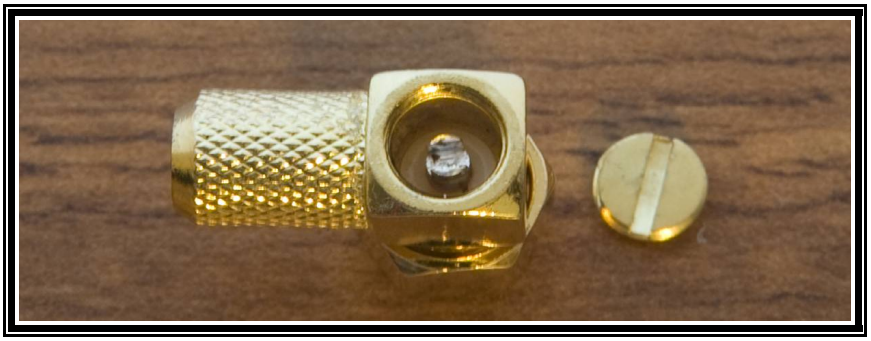
# Hacking the Antenna

The inexpensive antenna shipped with the WxShield can be easily modified as described here and becomes a very effective little antenna. You may have received a WxShield with an antenna that has already been modified. In that case, this section can be skipped.

The photo above shows the original antenna as delivered. Start by separating the black plastic cover from the metal connector. Grab the connector in one hand, then twist and pull on the plastic cover to remove it. It may take a bit of force to break the cover loose, and you may stretch or even break the little internal coil antenna in the process; this is okay.

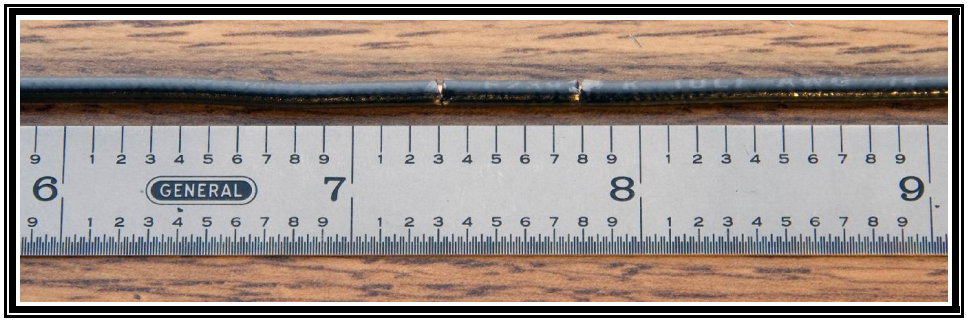
The above photo shows what's inside after removing the plastic cover. The next step is removing the coiled wire from the connector.

Remove the threaded plug on the rear of the connector. Inside is the connector's center pin post to which the coil wire is soldered. Use a small soldering iron to remove the wire coil from the connector.



In the above photo, some solder-wick has been used to clean up the end of the post. This is mostly done here for illustrative purposes and is not a necessary step in general.

Now cut three lengths of 14AWG copper wire; one 15-inches long and two 7-½ inches long. At least one of the shorter pieces must be insulated but the remainder can be bare if desired.

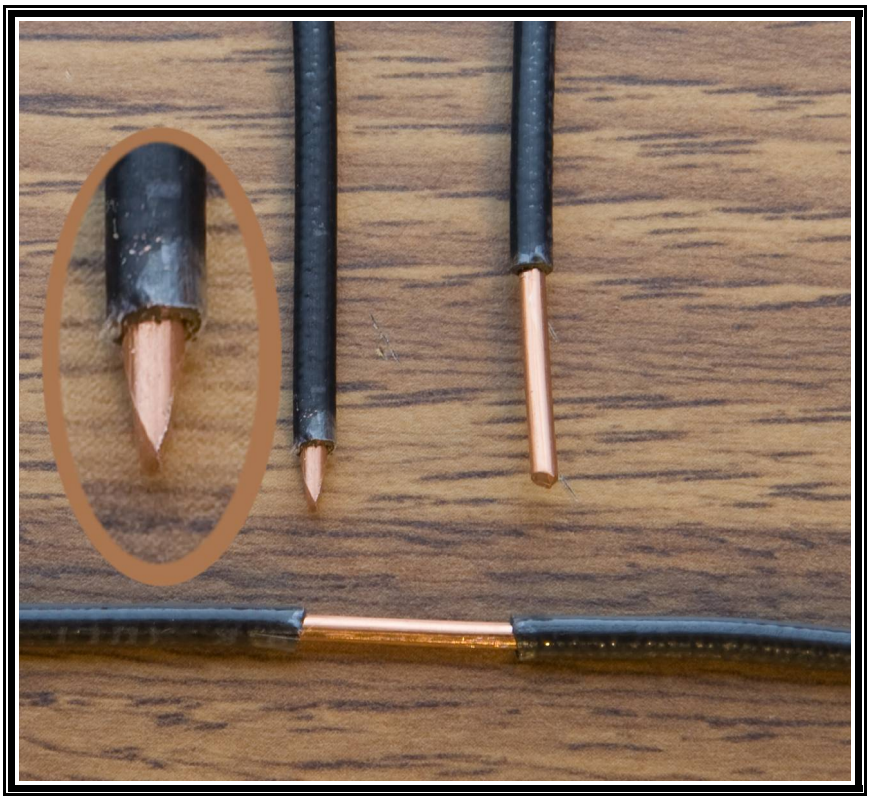



With a wire stripper, make two cuts in the insulation of the longer piece, at 7.3 and 7.8 inches from one end, as shown above. Take a sharp knife and cut longitudinally along the insulation between the two cuts as shown below.



This will allow the short piece of insulation to be removed from the wire, leaving a ½-inch long bare section in the longer piece. The bare section is intentionally not quite in the center of the wire.

Strip back the insulation on one end on each of the remaining two pieces of wire. Strip ½-inch on one piece and only about 1/8-inch on the other on. If you are using bare wire for ground radials, then the insulated piece should be stripped back about 1/8-inch.



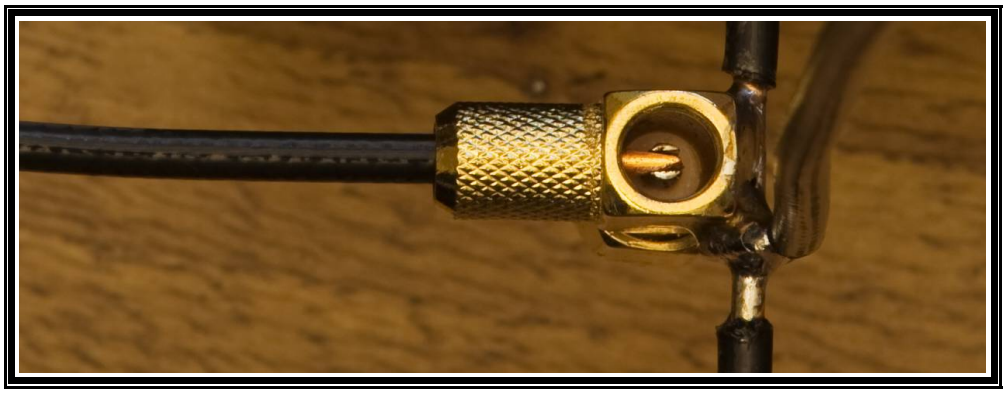
After stripping the 1/8-inch from one short piece, use a file or grinding wheel to shape the end of the wire into a two-sided, chisel-like point. The exploded inset in the photo above shows the filed point in more detail.

Now the longer piece is ready to be soldered to the connector body, forming two of the three ground radials.

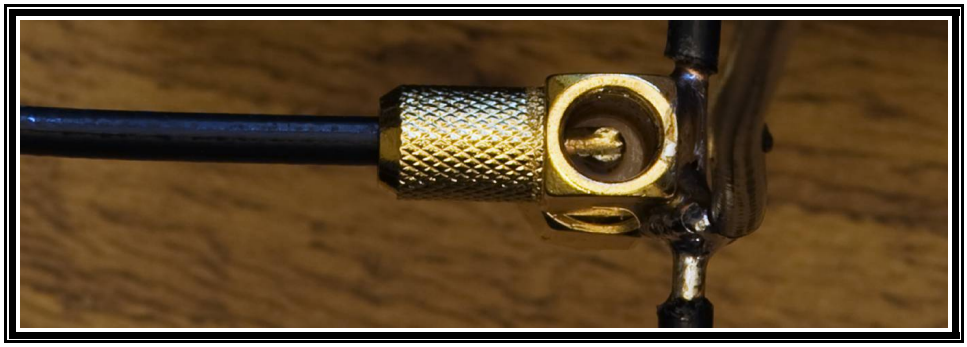
The photo above shows the longer wire laid in place on the connector body where it will be soldered. The middle of the wire is aligned with the middle of the connector body. However since the bare section is not centered, it appears offset to one side. This extra exposed wire on one side will be used to attach the third radial.

Start by tinning the bare section and bottom of the connector body with solder. Then bring the two together and apply heat until they are soldered together. Avoid using too much solder as some may wick onto the connector nut and lock it into place. The photo above shows the finished assembly.

While taking photos for this document, the third radial was soldered in place next. However, that got in the way of attaching the vertical antenna element making that step more difficult. The next step should then be to attach the vertical element and we'll show that next. You'll see the third radial already soldered on but if you follow this recommended order, the third radial will not yet be attached.



The photo above shows the chisel-pointed end of the vertical element slid into the connector body through the top opening. The pointed end has been positioned between the opening in the center pin post.

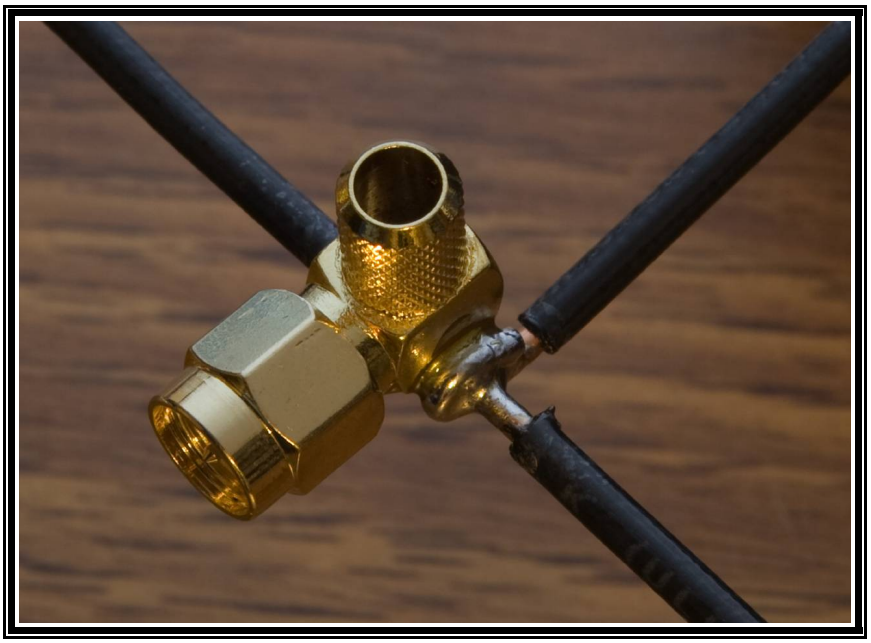


Using a smaller soldering iron, the center element can now be soldered onto the center pin as shown below. Be careful not to get solder onto the threads for the rear plug as this will prevent the plug from being re-installed. If solder does get onto the threads, it can usually be cleaned up with some solder wick.

Now the third ground radial can be attached. Bend a tight loop in the end of the short piece that was stripped back ½-inch. Hook it over the exposed bare section next to the connector body and crimp snugly with a pair of needle-nose pliers as shown below.

It may help to secure the third radial horizontally in a vise while soldering it to the long cross-radials. Applying heat mostly to the loop bent in the third radial will help to prevent melting of the previously solder joint between the long radial and connector body. If you have trouble melting this joint and it all comes apart, it is permissible to move the crimped loop about 1/8-inch away from the connector body and solder it in place there.

That's all of the soldering. Drill a hole in the end of the plastic cover the same size as the wire insulation. A 7/64-inch drill was used in this example.

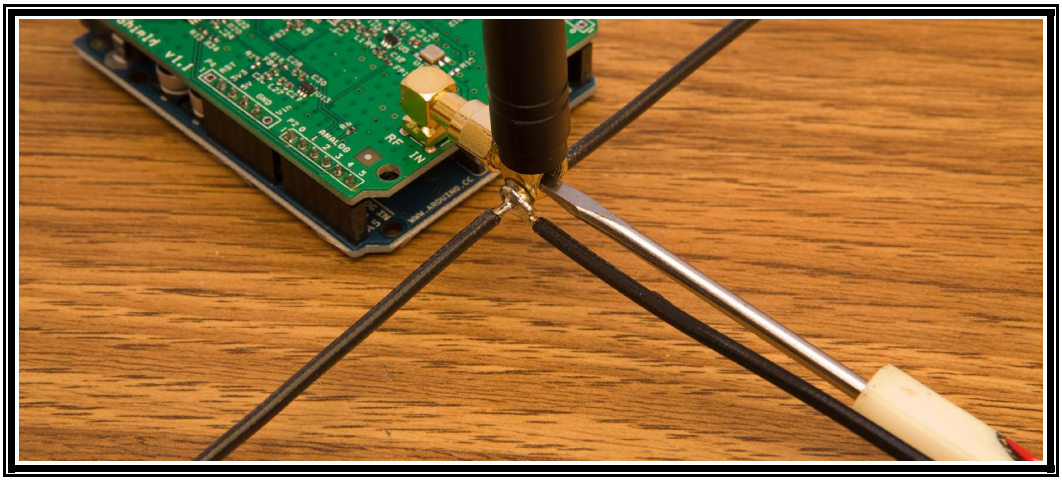


Slip the plastic cover down over the vertical wire element and work it back down in place as shown below.

Finally, trim the length of the vertical element. Measure this from the bottom surface of the connector to the tip of the vertical wire. The trimmed length should be 6.60 inches, 6-19/32 inches or 167.5mm. Don't include the diameter of the wire soldered on the bottom of the connector in this measurement. Try to get this as accurate as possible – the antenna's efficiency is quite sensitive to the length of the vertical wire.

This completes the antenna assembly. It is a good idea to wait until the antenna is installed on the WxShield before re-installing the small threaded plug.

# Fixing the Antenna Shorting Problem




On some of the antenna connectors, the center pin can be a bit loose. Tightening the connector nut can cause the center pin to get pushed out toward the threaded plug. If the plug is installed it can short against the plug causing loss of signal strength. If the plug is not installed, it is also possible for the vertical wire to short against the connector body if the threaded plug is removed.

If this problem occurs, remove the threaded plug and use a small screwdriver or other small drill bit or something similar to firmly push the center pin back in after tightening the connector nut.

Some of the antennas delivered in late 2012 may have a small round plastic insulator under the threaded plug to help prevent this problem.

In reality, it is not all that important to install the rear plug into the connector body. Signal strength may be very slightly improved with the plug in place. However, if the antenna is frequently removed and re-installed, it will be easier to leave the plug out to provide access to the center pin.

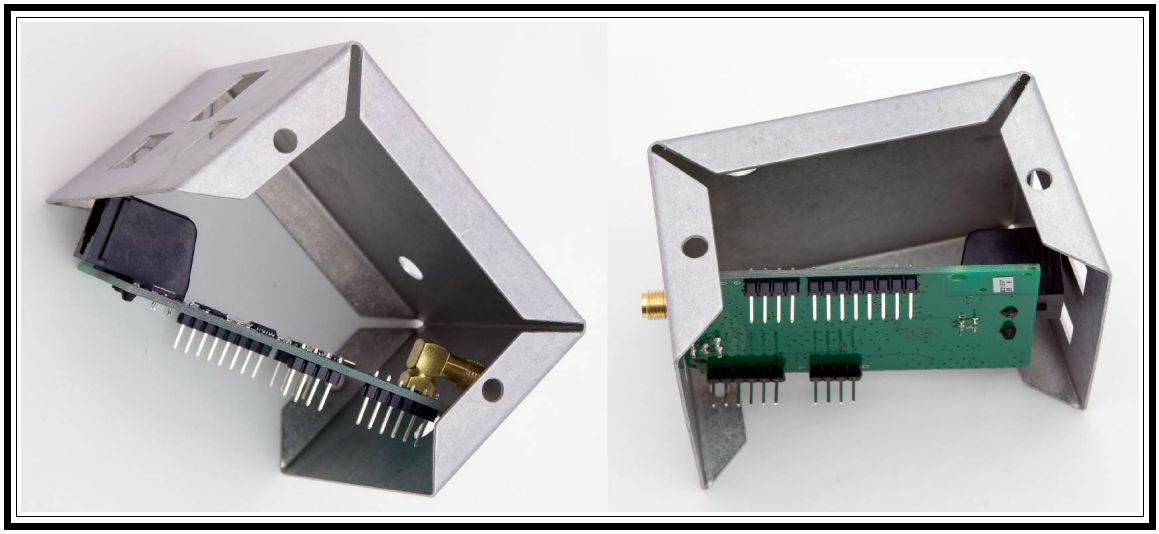
# The WxBox

As mentioned in the introduction, some WxShield units are provided with an optional custom aluminum enclosure. This is a very inexpensive aluminum box made from 30 mil (0.76mm) thick 5052 aluminum. It has a rough finish and a few holes to accept the various connectors on the WxShield (USB, SHT15, DC power and RF input). There is also a hole in the top for viewing the red/green LEDs that indicate when signals are being received.

The WxShield is only held in the WxBox by connectors protruding through holes in the box. There are no mounting screws (other than four #6-32 x 1/4" screws used to hold the two halves of the box together).

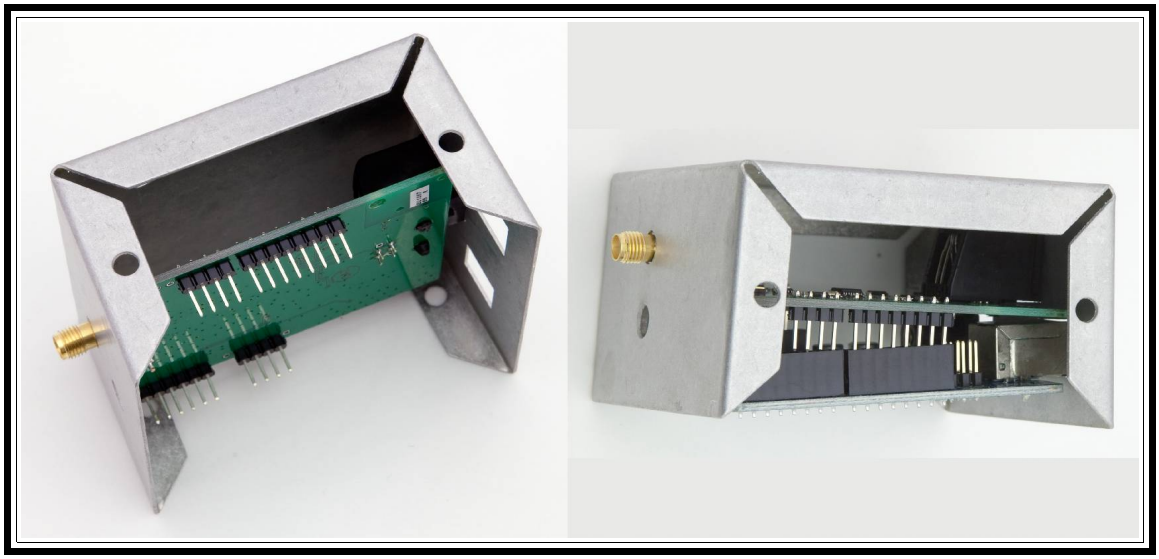
Installing the WxShield into the WxBox requires the half of the box with mounting holes to be slightly bent open. This task is easier to accomplish with two people – one person gently bends the box open while the other guides the WxShield into the mounting holes.

To start with, separate the WxShield receiver PC board from the Arduino processor PC board. Then place the receiver board's RF input connector against the round hole in the end of the box as shown in the photograph below on the left (an errant hole is also visible there which does not exist on the current WxBox design).



Next, gently spread the ends of the box apart while guiding the receiver board into place as shown below. At some point, as the RJ14 connector is moved towards its mounting hole, the RF connector will kind of "pop" into place and the RJ14 connector can then be easily aligned with the correct square hole. This state of affairs is seen in the photograph above on the right.

Now the Arduino processor can be re-assembled to the WxShield receiver board. Start by inserting the Arduino's USB connector through the matching hole in the box. At the same time, align the pins on the receiver with the mating socket on the Arduino processor.



This is shown in the photographs above-right and below-left, from opposite sides of the box. Be careful to get the the pins aligned correctly as shown; it is very easy to mis-align the pins and this may destroy some parts on the PC boards when it is powered up again.

Next, slip the two halves of the box together as shown in the photograph above-right. It may be necessary to gently bend the outer box half together to get the mounting holes to line up with the inner box half.

The two box halves are secured with four #6-32 x 1/4” machine screws. They are force-threaded into the smaller holes of the inner box half. Be careful not

to strip these threads if the screws have already been installed once. Tighten them only lightly.

The hole locations on inner and outer box halves are designed so the mounting screws will pull the outer box half tightly against the inner box half and it may be necessary to gently hold the outer box half tight against the inner half while installing the screws.

Dis-assembly is the reverse of the assembly process describe above. As with assembly, it is much easier to have a second person help with removal of the receiver board from the box. One person carefully bends the box ends apart while the second person gently coaxes the receiver board out. Do this by moving the RJ14 connector upwards (away from the box) and at the same time moving the RF input connector out of its mounting hole.