Heating Box Guide

Marcus Thelin Schmidt

Marcus.thelin.schmidt@gmail.com

0735951004

Table of Contents

[Background 3](#_Toc174969248)

[Parts 4](#_Toc174969249)

[Assembly 6](#_Toc174969250)

[Box 6](#_Toc174969251)

[Electronics 9](#_Toc174969252)

[Software 14](#_Toc174969253)

[How to Use 15](#_Toc174969254)

[Step 1 - Assembly 15](#_Toc174969255)

[Step 2 – Download Programs 15](#_Toc174969256)

[Step 3 – Install scripts from GitHub 15](#_Toc174969257)

[Arduino 15](#_Toc174969258)

[Python/PyCharm 16](#_Toc174969259)

[MATLAB 21](#_Toc174969260)

[Step 4 – Operate 23](#_Toc174969261)

[USB port 23](#_Toc174969262)

[Configure and Upload Arduino Code 25](#_Toc174969263)

[Run Python script 25](#_Toc174969264)

# Background

The idea of the heating box came from the need of having a temperature regulated room like the one in the lab, but portable for field work. This box can be used for various experiments where insects or items needs to be keep at a specific temperature for a prolonged time. So far it has been used for two different experiements. The first one involving the effect of tempature on the bee attennae’s response to smells .The bees were kept at 32 degrees for 2 hours before extracted from the box and tested with other instruments. This experiement took place in one of the labs at the university. The other experiment took place up in Abisko where the box was brought. Tests were conducted at various temperture between 30-40 degrees with a modified side board of the box with a rougly 2 cm hole. The idea was to keep bees attached to a small stick which is brought through the hole to then touch a small accelerametor inside the box to see the effect of temperature on buzz pollination.

The foundation of the box is designed for all purpose usage of a temperature regulated space. As shown further down the guide, the box is made up of different pieces to make them interchangeable with custom pieces (like the side piece with the hole). This makes it customazible for experiments with specific requirements.

If current pieces of the box can’t be modified to the users liking, new pieces might have to be crafted to meet the requirements.

# Parts

[Arduino Uno](https://www.digikey.se/sv/products/detail/arduino/A000066/2784006)

[USB-A to USB-B (Arduino to Computer)](https://se.rs-online.com/web/p/usb-cables/1862787?searchId=2841ce73-240f-48a9-99e1-e1d2efeebf47&gb=)

[Arduino Housing (3D-printed) (only used the base)](https://www.thingiverse.com/thing:713091)

[Light Dimming Module (RobotDyn 230V AC)](https://www.amazon.com/Genuine-Programmable-Light-Dimmer-110V/dp/B0893K6ZPK/ref=pd_rhf_se_s_pd_sbs_rvi_d_sccl_2_2/142-3971428-9762515?pd_rd_w=EzLsG&content-id=amzn1.sym.9c71db11-3b2f-49a1-9fef-afd524b20130&pf_rd_p=9c71db11-3b2f-49a1-9fef-afd524b20130&pf_rd_r=PSHDPSAAJWPTSBTCJG2D&pd_rd_wg=47jUT&pd_rd_r=21dd7f5c-6bcb-4f2b-b256-ce4b8200f937&pd_rd_i=B0893K6ZPK&psc=1)

[3-way Extension Cord](https://www.clasohlson.com/se/Grenuttag-3-vags-3-m-kabel/p/36-8901)

[Breadboard](https://se.rs-online.com/web/p/breadboards/1892277?searchId=d875f1f3-4e14-474f-bbc1-ec9745237753&gb=s)

[Temperature Sensor (DHT22)](https://www.digikey.se/sv/products/detail/dfrobot/SEN0137/6588461)

[Shrink Tube](https://se.rs-online.com/web/p/heat-shrink-tubing/3948315?searchId=e43b91ff-40f6-46fe-a619-6fa9b1380bd3&gb=s)

[Colored Thin Wires (22 AVG)](https://www.digikey.se/sv/products/detail/sparkfun-electronics/PRT-11367/5956251?s=N4IgTCBcDaIAoCUAqBaAjGgzANgOwgF0BfIA)

[230V Plugs (EU)](https://se.rs-online.com/web/p/mains-plugs/7050822?searchId=555f2de7-51ca-4640-96db-96d3bd88b19d&gb=s)

[Infrared Heat Lamp](https://se.rs-online.com/web/p/heat-lamps/7960117?searchId=5944971f-c24b-447d-a794-ff9aa9c37aab&gb=s)

[Lamp Holder](https://se.rs-online.com/web/p/lamp-holders/2678108?searchId=c9a46c65-eb9f-4a00-8b60-3107da5b84d8&gb=s)

[40mm Fans:](https://www.digikey.se/sv/products/detail/nidec-components-corporation/F410T-05LC/1165524)

[Fan Mounting (3D-printed)](https://www.printables.com/model/40322-simple-2020-40mm-fan-mount)

[Fine Insect Metal Mesh](https://www.vidaxl.se/e/vidaxl-insektsnat-aluminium-100x1000-cm-silver/8718475880639.html)

[Sturdy Metal Mesh (as support for the Insect Mesh)](https://www.clasohlson.com/se/Anslagstavla-metallnat-50-x-70-cm/p/44-5000)

[Plywood](https://www.byggmax.se/konstruktionsplywood-15-mm)

[Square Wood Column](https://www.byggmax.se/regel-45x45-p08145045)

[Acrylic Plate](https://www.byggmax.se/akrylskiva-3mm-p21190#1225=88061)

230V-approved wires: Salvaged from old extension cords and similar

[Arduino Crimps (Male)](https://se.rs-online.com/web/p/crimp-contacts/6881442?searchId=b2bf5832-e911-41f9-a393-e8617a3e6456&gb=s)

[Arduino Crimps (Female)](https://se.rs-online.com/web/p/crimp-contacts/8201529?gb=s)

[Arduino Crimp Holders (4-way)](https://se.rs-online.com/web/p/wire-housings-plugs/6812824P?gb=s)

[Fuse (2.5A)](https://www.digikey.se/sv/products/detail/littelfuse-inc/031202-5HXP/777236)

[Fuse Holder](https://www.digikey.se/sv/products/detail/littelfuse-inc/01500332H/3425489?s=N4IgTCBcDaIGYDYDsSCsBaAdgExAXQF8g)

[M4 inserts](https://se.rs-online.com/web/p/threaded-inserts/4257531?searchId=739c7d54-33a0-4a4e-9d50-15a13f68af5f&gb=s)

[M4 Bolts (recommend getting slightly longer ones)](https://www.ahlsell.se/products/infastning/maskingangad-skruv/maskinskruv/100676)

[M3 Bolts](https://www.byggmax.se/skruv-mrx-fzb-3x30mm-p2420628)

Miscellaneous:

* Screws (fastening Lamp Holders and Fans onto the floor)
* Bolts (M3 for fans, M4 for rest)
* Nuts (M3 for fans)
* Tape
* Sandpaper
* Epoxy
* Aluminum Foil

Tools:

* Wire stripper
* Soldering station (+tin & flux)
* Heat gun/lighter for shrinking tube
* Screwdriver
* Electric Saw (for wood and plastic)
* Manual Saw (for smaller wood parts)
* Bench Pillar Drill or Drill Stand (to get straight perpendicular holes)
* Crimping Tool (for Arduino Crimps)
* Multimeter

# 

# Assembly

The box is separated into several larger parts as well as all the electronics. This part of the guide will show how to first assemble the larger parts “Box” and then how to connect all the “Electronics”.

## Box

First step is to identify the bottom plate and the right plate of the box. The bottom plate is easy to see as it has gotten the lamp holders, fans and some cables attached to it. The right plate has a large “R” written on its’ sides, along with the text “Inner” on one side and “Outer” on the other. Screw on the Right plate onto the right side of the bottom plate (shown by the letter “R”).

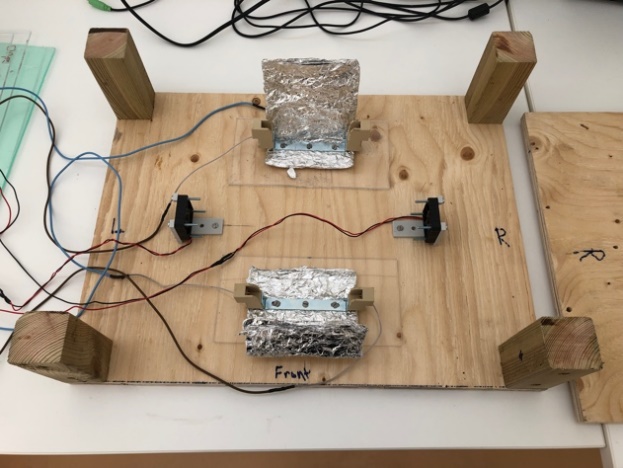


Figure 1

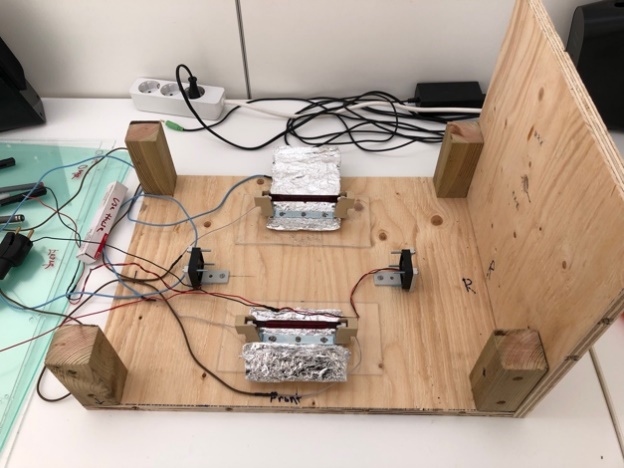


Figure 2

Make sure the side plate is firmly attached with all screws, but do not screw too hard as there is a risk of the screws just piercing through the wooden material of the side plate (figure 2).

At this step you can also connect the heating lamps into the lamp holders (figure 3).

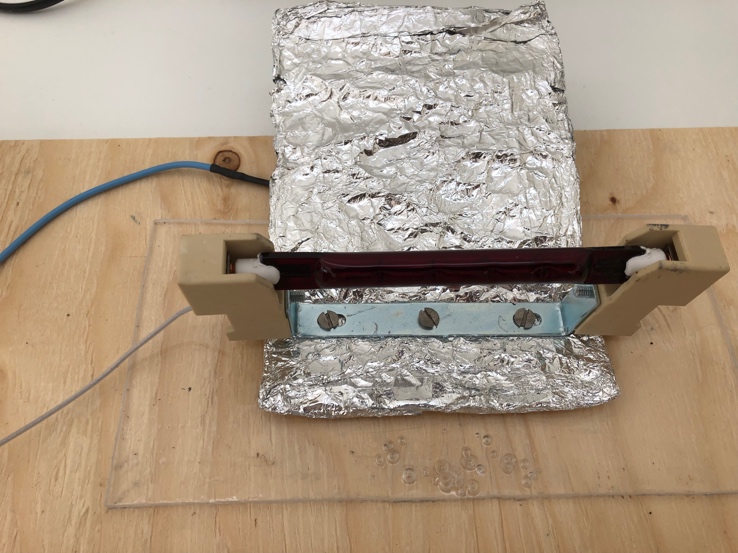


Figure 3

Fold the aluminum cover over the lamp to protect the bright light to get straight into the users’ eyes. Also, slightly fold the bottom part of the aluminum cover upwards to protect the plastic bottom under the lamp holder from overheating and melting by direct heat from the lamp.

Next step is the most crucial as far as time spent assembling is concerned. If this part is done incorrectly, it will take a lot of time to go back and fix it later. You want to have all the cables going in/out of the box to fit in the opening at the bottom of the left side plate (figure 5 and figure 6).

As figure 4 displays, organize all the cables towards the left. It is two set of cables from the lamp holders, two set of cables from the fans and two set of cables from the sensors you can use up to four at the same time).

A wooden board with wires and wires

Description automatically generated

Figure 4

When all cables have been found and organized you want to attach the left side plate marked with an “L” as well as “Outer” and “Inner” on respective sides (figure 5).

A wooden box with wires and wires

Description automatically generated

Figure 5

A group of wires sticking out of a hole in a wood surface

Description automatically generated

Figure 6

After this, you want to find the transparent plastic plate for the back of the box (figure 7). It has “Back”, “Inner” and “Outer” written on it. It is important it is the correct plate for each side even here as the screw holes are a tight fit customized for each side.



Figure 7

A wooden box with wires and foil on it

Description automatically generated

Figure 8

Now do the same for the front (figure 9 and 10).

A white table with red writing on it

Description automatically generated

Figure 9

A wooden box with wires and wires

Description automatically generated

Figure 10

Only remaining part of the box is the top plate. This part does not attach with screws or anything similar. This is intentionally done to have it easily removable as the user most likely want to get access into the box on a regular basis during experiments. If a tighter fit is sought after, the user can use silver tape to secure the top plate. The top plate is marked with “Top” (figure 11 and figure 12).

A white table with a clear surface

Description automatically generated

Figure 11

A wooden box with wires and wires

Description automatically generated

Figure 12

How to attach the sensors inside the box will be shown in the next chapter “Electronics”.

## Electronics

The electronics setup consists of:

* Arduino
* Dimming Module
* 3-way extension cord
* Breadboard
* 1-4 sensors
* 2 lamp holders
* 2 fans
* USB connector

As shown during the box assembly the two fans, lamps holders and sensors are within the box with their cables going out (figure 5). The other parts are packed together in a small box outside (figure 13).

A plastic box with wires and wires

Description automatically generated

Figure 13

First step is to pull the cables from the lamp holders, fans and sensors through the opening of the small electronics box (figure 14).

A group of wires in a plastic container

Description automatically generated

Figure 14

Then connect the two lamp holder plugs into the extension cord (figure 15).

A plastic container with wires

Description automatically generated

Figure 15

Next step can be a bit tricky as it requires the user to accurately put all the small cables into the correct slots on the Arduino, breadboard and dimming module. If something is not working correctly with the box when trying to run it later on, this is a part of the guide to go back to and reassure everything is connected the way it should.

Arduino to Breadboard connections

The red wire goes from 5V slot on the Arduino to the left column on the breadboard and the black wire from “GND” on the Arduino to the right column on the breadboard, see figure 16.

A circuit board with wires

Description automatically generated

Figure 16

This connection allows 5V to be accessed from the entire vertical row up from the red wire on the breadboard. Likewise, ground (GND) can be accessed from the entire vertical row up from the black wire on the breadboard. When these are labelled on connections below you can connect it to any of the slots on the respective rows.

Fan connections

*Fan 1:*

Red 🡪 4 (Arduino)

Black 🡪 GND (breadboard)

*Fan 2:*

Red 🡪 7 (Arduino)

Black 🡪 GND (breadboard)

Sensor connections

Up to 4 different sensors can be used at the same time. In the pictures presented in this guide, only two have been in use as it was needed for the latest experiment. It’s important to keep in mind which sensor is connected into which slot on the Arduino as this will determine which sensor is represented as 1,2,3 or 4 in the code. It is labelled as below:

*Sensor 1:*

Red 🡪 5V (breadboard)

Black 🡪 GND (breadboard)

Green (signal) 🡪 12 (Arduino)

*Sensor 2:*

Red 🡪 5V (breadboard)

Black 🡪 GND (breadboard)

Green (signal) 🡪 10 (Arduino)

*Sensor 3:*

Red 🡪 5V (breadboard)

Black 🡪 GND (breadboard)

Green (signal) 🡪 11 (Arduino)

*Sensor 4:*

Red 🡪 5V (breadboard)

Black 🡪 GND (breadboard)

Green (signal) 🡪 8 (Arduino)

Extension cord to Dimming module connections

Brown 🡪 “IN”

Blue 🡪 “N”

Black 🡪 “Out”

See figure 17.

A close-up of a circuit board

Description automatically generated

Figure 17

Dimming module to Arduino connections

White (PSM) 🡪 3 (Arduino)

Green (Z-C) 🡪 2 (Arduino)

Blue 🡪 GND (breadboard)

Red 🡪 5V (breadboard)

Arduino to Laptop connection

Connect the large black USB-A to USB-B cable from the Arduino to the laptop that will be used during experiments.

To attach the sensors in the desired location within the box, it has previously been used double-sided silver tape on the back of the sensor and then put on the inside box’s side plates. This way it’s easily removable and adjustable (figure 18).

A wire on a wood surface

Description automatically generated

Figure 18

Everything is now assembled in preparation for operation. Go to chapter “How to Use” to follow the subsequent steps to get it up and running.

# Software

The logic of the temperature regulation is controlled by the Arduino. This Arduino code is the only software necessary to run the box at a base level. But to have a better overview of the temperature at the different sensor locations and power output of the heating lamps in real time, there is a Python script to be run. It is highly recommended to use this Python script while running experiments for better control and precision of when to conduct the tests. The only downside of using this Python script is the necessity of having a laptop connected to the Arduino while doing so. This means that you can use the box and its features without a laptop connected, but you will not be able to see the current temperature of sensors and power output of the lamps. You will simply only be able to upload the Arduino code with the temperature set point you want to have inside the box and then wait roughly 20 minutes for it to reach and stabilize at that temperature.

The repository can be found at: <https://github.com/MarcusTSgithub/BiLab_Heatbox>

Please read chapter “How to Use” to get a thorough step by step guide on how to operate it.

# 

# How to Use

## Step 1 - Assembly

Follow the steps shown in the chapter “Assembly”.

## Step 2 – Download Programs

You will need the following programs to operate the heating box:

* Arduino IDE (<https://www.arduino.cc/en/software>)
* PyCharm, Community Edition (<https://www.jetbrains.com/pycharm/download>, Scroll down for Community Edition)
* Python (this will be downloaded later when we install the scripts)

## Step 3 – Install scripts from GitHub

Go to <https://github.com/MarcusTSgithub/BiLab_Heatbox>.

Click on the green button “<> CODE” A green rectangular button with white text

Description automatically generated and choose “Download ZIP”. Where the ZIP file is downloaded and unzipped does not matter as the files within the folder will be moved later.

### 

### Arduino

After unzipping the files there should be three subfolders: Arduino, MATLAB and Python. First step is to move all the Arduino files into the correct location on the computer. Locate the original “Arduino” folder on the computer which was created when “Arduino IDE” was downloaded. It is usually located in “C:/Users/”user name”/Documents” on Windows or “Documents” on Mac. Replace that folder with the new one from GitHub.

IF there already are other Arduino projects and/or libraries within the original folder the user has to go into the original folder and first copy over the file called “BiLAB\_Heatbox.ino” from the github folder. Secondly, go into the original folder->libraries and copy over all the content from github folder->libraries.

Now all necessary Arduino files have been moved into the correct location and the folder that was unzipped from Github can be closed down.

Open the file called “BiLAB\_Heatbox.ino”. It will prompt the user telling it that a folder for that project is needed, click “create folder” and proceed. The Arduino IDE application should now be open with the code shown in the window.

Lastly, a test will be performed to see if the script copied over correctly. Click on the dropdown menu that says “Select Board”. Scroll down until “Arduino UNO” is found and select it. Afterwards click on the top left circular button called “Verify”. Wait until the compilation is done. If it’s successful it will say “Done compiling”. By doing this now a verification is given to see if the libraries needed to run the script got moved into the correct folders (figure 19).

A screenshot of a computer

Description automatically generated

Figure 19

### 

### Python/PyCharm

Start the PyCharm application. If you get prompted to choose weather or not to import settings, click “Do not import settings”. The program should now be open and the window should look like figure 20.

A screenshot of a computer program

Description automatically generated

Figure 20

Click on “New Project”.

Give the project an appropriate name, e.g “Heatbox”, see top of figure 21.

A screenshot of a program

Description automatically generated

Figure 21

Click “Create”.

If you don’t already have Python installed on the computer from before, this is where you will get prompted to download it which you need to do.

After all have been loaded the new window should look like figure 22.

A screenshot of a computer

Description automatically generated

Figure 22

If Python installed correctly, you will see the Python version down at the bottom right. I have Python version 3.12 as seen figure 23. Having the exact same version is usually not necessary, but I would recommend getting version 3.8, 3.10, 3.12 or something newer depending on when you read this.

Now expand “External Libraries”🡪 “< Python 3.12 (Heatbox) >” 🡪 “site-packages”. It should look like figure 23.

A screenshot of a computer

Description automatically generated

Figure 23

This is where you will see all the libraries for the project. Currently it’s almost empty because we haven’t downloaded any yet. Now you want to right click “site-packages”🡪”Open In” 🡪 Explorer(Windows)/Finder(Mac). This will open the folder where all the libraries will be copied to. On the new Explorer/Finder window that just opened go into the folder “site-packages”. Now open the unzipped folder from Github and go into the subfolder called “Python”. You will find two folders called “serial” and “serial-0.0.97.dist-info”, copy these into the “site-packages” folder. If it was copied properly, they should now pop up in the PyCharm window on the left under “site-packages” (figure 24).

A screenshot of a computer program

Description automatically generated

Figure 24

The remaining libraries will be downloaded through the PyCharm program, but we need to get this specific library this way because the one found on PyCharm doesn’t work properly.

To get the remaining libraries we go into PyCharm again and go to “Settings”. Then go to “Project: Heatbox”🡪 “Python Interpreter” (figure 25).

A screenshot of a computer program

Description automatically generated

Figure 25

Here you want to click on the plus-symbol right above the text “Package” (figure 25). This will open a window called “Available Packages” (figure 26). Now click on the symbol with two rotating arrows next to “Description”. This will load all available packages you can download.

A screenshot of a computer

Description automatically generated

Figure 26

Here you want to search for and download the following packages:

* openpyxl
* matplotlib
* pandas

After these are downloaded you can close the window and click “OK”. You should now see a bunch more libraries pop up under “site-packages” (like figure 24, but more).

Next step is to create the file for the script. Since I’m no expert in working with projects on GitHub I’ve found the easiest way for me to get the script to work is to make a new “Scratch” file and copy over the script code onto this file.

To do this you need to go the left panel and right-click “Scratches and Consoles”🡪 “New”🡪 “Scratch File”, see figure 27.

A screenshot of a computer

Description automatically generated

Figure 27

In the “New Scratch File” window scroll down until you find “Python” and choose it. A new file called “scratch.py” will now be created under “Scratches and Consoles”🡪 “Scratches”. Now go back to the unzipped github folder🡪 “Python” and right click “BiLAB\_Heatbox” and open with Notepad or similar text-program. It should look like figure 28.

A screenshot of a computer

Description automatically generated

Figure 28

Select and copy all the text and then paste it into the scratch-file window that was created in PyCharm (figure 29).

A screenshot of a computer program

Description automatically generated

Figure 29

I would now recommend renaming the scratch file to something appropriate like “Heatbox Script.py”. Do this by right-clicking “scratch.py” on the left panel 🡪 “Refactor” 🡪 “Rename”. Make sure you keep the “.py” at the end of the name.

If all the libraries/packages were downloaded correctly and the code copied correctly there should be no errors in the code, only some “warnings”/”weak warnings”. You can see this by hovering the symbols at the top right of the code window, seen in figure 30.

**A screenshot of a computer screen

Description automatically generated**

Figure 30

The only remaining part of installing the Python scripts is finding the name of the correct USB-port used for Arduino on the computer. This port name is given to the variable “SERIAL\_PORT” seen at row 9 in the script. I will show how this is done in the next chapter where we will learn how to operate the heating box.

The installation of the Arduino and Python scripts are now prepared and ready for operation.

### MATLAB

There is a subfolder called MATLAB in the GitHub repository. This contains data gathered from the Python script from previous experiments that the writer used in MATLAB to analyze various parameters. This is NOT necessary to download and install to operate the box. It is specifically used for post analysis by the writer. The user can look at it for inspiration on how the files created by the Python scripts can be used in MATLAB, but it is not recommended to follow strictly.

## Step 4 – Operate

### USB port

Connect the USB cable from the Arduino into the laptop if this is not already done.

Open the Arduino script in the Arduino IDE. Now we want to make sure the laptop recognizes the Arduino connection and is properly connected to it. Usually the name of the USB port will be shown in the tab where the Arduino type can be seen in the window, as shown in figure 31.

A screenshot of a computer

Description automatically generated

Figure 31

On my computer it is called “/dev/cu.usbmodem14101” which is common for Mac computers, on a windows computer it can be something similar to “COM3”.

If nothing is shown here, there is another way to look for the name of the USB port name with PyCharm. We will use a simple script and run it in a new Scratch file.

Create a new scratch file the same way as one done earlier for “Heatbox” (figure 27). Then copy this script into the new Scratch window:

from serial.tools import list\_ports  
  
ports = list\_ports.comports()  
for p in ports:  
 print(p)

Then right-click anywhere on the window and click “Run scratch\_1”. The console window should pop up with at the bottom of the PyCharm window with the name of the USB port where the Arduino is connected as shown in figure 32.

A screenshot of a computer

Description automatically generated

Figure 32

Here you can see “/dev/cu.usbmodem14101” show up again.

Now go back to the main script “Heatbox” and scroll down to row 9 where you have the variable called “SERIAL\_PORT”. Replace the string to the right of the equal sign to the name of your USB port, and remember to keep the quationmarks at the start and end of the port name (this should turn the entire name green as seen in figure 33).

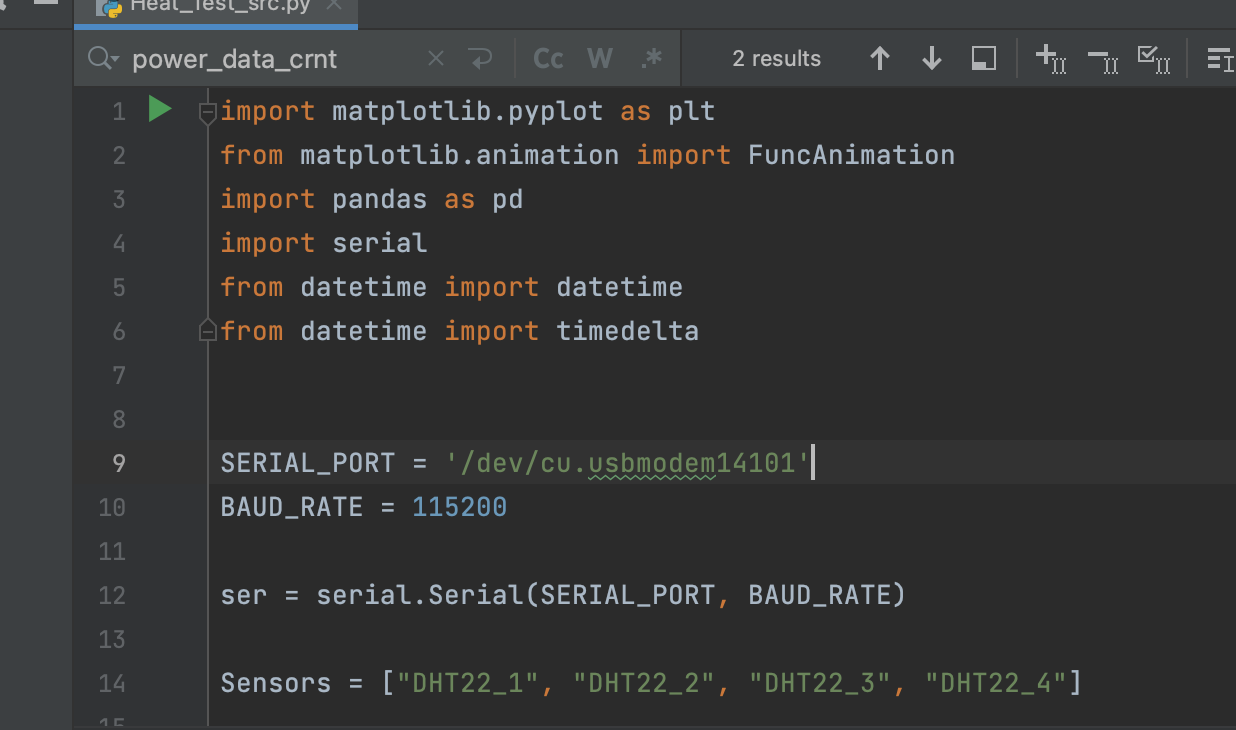


Figure 33

This port name is important as we now can tell our main Python script (“Heatbox”) where to get its continuous information from during operation.

### Configure and Upload Arduino Code

Final step before starting the heating box is uploading the code to the Arduino. Open the Arduino code in Arduino IDE. Scroll down to row 21 where you see “const double setPoint= 32;”. This is the variable that has to be changed by the user to specify the desired temperature set point for the experiement.

When the desired set point has been changed, click on the “Verify” button at the top left to make sure the code still compiles. When this is complete, click on the button next to it called “Upload” (make sure the USB cable is connected). When this is complete, the code is now uploaded to the Arduino and ready to run.

When the Arduino is plugged into a laptop and gets powered up it will start running the code by itself as soon as it’s connected. This is the way to do it if the user prefers to run the heating box at a base level without the Python script overview. Follow the next sub chapter “Run Python script” to learn how to run it through the Python script.

### Run Python script

Before running the Python script, make sure the Serial Monitor in Arduino IDE is off as this would interfere with the readings from the Arduino to PyCharm. If you can see the “Serial Monitor” tab as in figure 34, it is active and has to be closed.

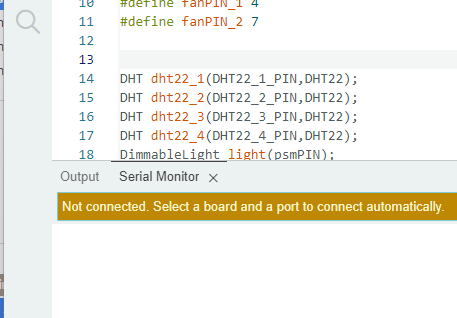


Figure 34

Open the Python script in PyCharm. This will most likely open automatically when you open PyCharm if it was the latest project you worked on. As mentioned before, the heating box should automatically have started as soon as the USB cable got plugged into the computer. If the extension cord is already plugged into an outlet (which it shouldn’t yet) the lamps should be on, but if it’s not you can see that it is on by looking at the fans which should be rotating. But now we want to restart the process and starting it up through the Python script instead.

Right-click anywhere on the PyCharm window where the code is 🡪 click “Run Heatbox”. The console window at the bottom of the PyCharm window will now open up if it isn’t already there. It will print some rows of text that should look similar to figure 35.

A screenshot of a computer

Description automatically generated

Figure 35

At the bottom it will prompt the user to answer with “y” or “n” depending on if the user desires the data from the test to be saved onto an Excel file (figure 35). This answer needs to be entered in the console before the heating box starts. Simply click at the same row the question is printed, type one of the options and press Enter. Once it’s done the code will fully start the process and a plot will show up with the different sensor readings and lamp power output (figure 36).

When the plot shows up like figure 36 below, this is when it’s recommended to plug in the extension cord into the outlet and power up the lamps.

A screenshot of a computer

Description automatically generated

Figure 36

The “(C)” that can be seen on the legend in the plot is which sensor is chosen as the reference when the computer is trying to stabilize the temperature (figure 36). I.e. the Arduino is trying to get the temperature of this sensor reading as close to the set point as possible, while the other sensors as only reading the temperature at their respective location.

If the user wants to change the reference sensor it will need to go into the Arduino code and change a few variables. Let’s say we want to change from sensor 1 to sensor 3 the following changes need to be made:

At row 37-38:

int chosenSensor = 1; // [1 = DHT22\_1, 2 = DHT22\_2, 3 = DHT22\_3, 4 = DHT22\_4]

double TempSensor = dht22\_1\_t;

change to

int chosenSensor = 3; // [1 = DHT22\_1, 2 = DHT22\_2, 3 = DHT22\_3, 4 = DHT22\_4]

double TempSensor = dht22\_3\_t;

At row 169-170:

TempSensor = dht22\_1\_t; //needs to be changed for specified sensor

if (readSuccess\_dht22\_1\_t){ //needs to be changed for specified sensor

change to

TempSensor = dht22\_3\_t; //needs to be changed for specified sensor

if (readSuccess\_dht22\_3\_t){ //needs to be changed for specified sensor

If another reference sensor is desired change the “3”’s to the respective sensor number instead.

When the box has been active for a while the plot will look similar to figure 37. Here you will see how the temperature stabilizes over time and when it gets steady enough for the experiments to be conducted.

A screenshot of a computer

Description automatically generated

Figure 37

When the test is done, simply close down the plot and the data will be automatically saved onto a new Excel file in the same folder as the Python script.