How to install the Arduino Node

An Arduino Node is a device which is used to send data, the technology used is LoRaWAN and to send data you can modify the settings as you want. This Arduino Node is used for a project in the Østfold University College. This one transmits data like temperature sensors and also light sensors. If you want to understand how we make this kind of device, you can follow this little guide.

First of all, to use this guide you must have these different components:

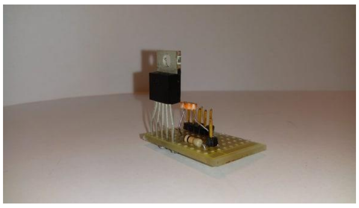
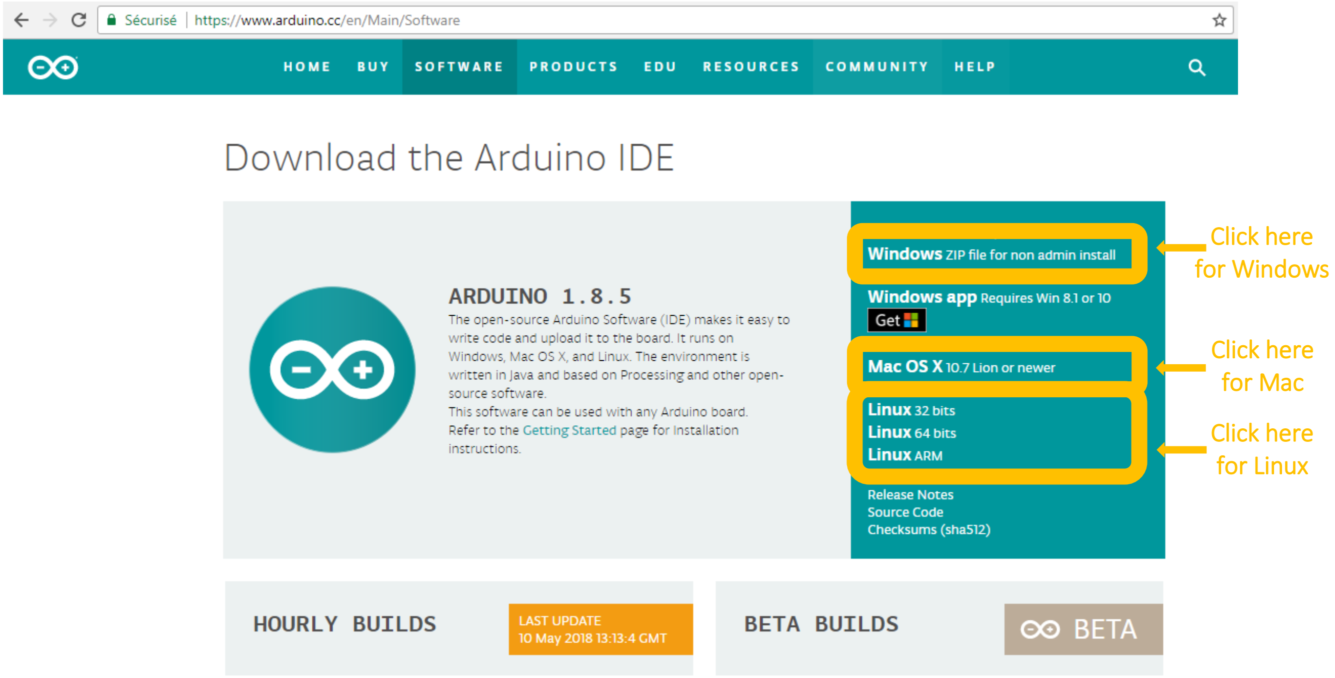
	Arduino Uno Rev 3
	SX1272 Embedded Shield
	Homemade electronic circuit
	Wire type A/B
	Wires

Table of contents

Now, if you have all these things you can start downloading the Arduino IDE and click on the OS that you selected and it will download it.



Download the Arduino IDE

ARDUINO 1.8.5

The open-source Arduino Software (IDE) makes it easy to write code and upload it to the board. It runs on Windows, Mac OS X, and Linux. The environment is written in Java and based on Processing and other open-source software.
This software can be used with any Arduino board.
Refer to the Getting Started page for Installation instructions.

Windows ZIP file for non admin install
Windows app Requires Win 8.1 or 10
Get
Mac OS X 10.7 Lion or newer
Linux 32 bits
Linux 64 bits
Linux ARM
Release Notes
Source Code
Checksums (sha512)

HOURLY BUILDS — LAST UPDATE 10 May 2018 13:13:4 GMT

BETA BUILDS — BETA

Click here for Windows

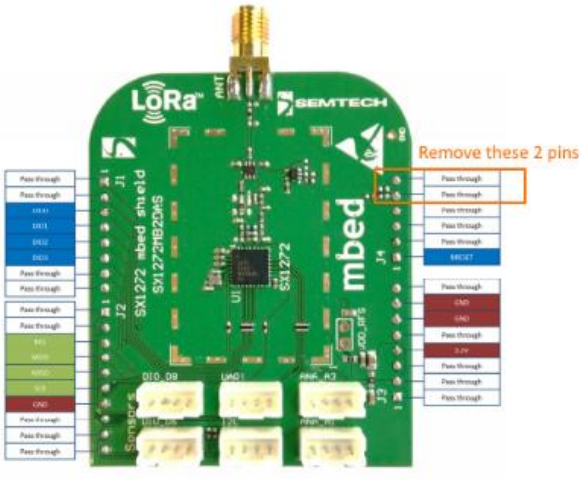
Click here for Mac

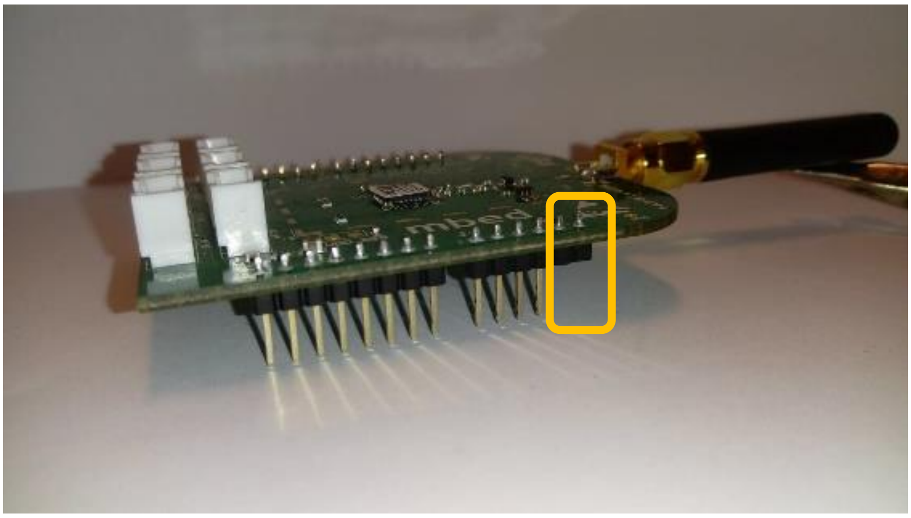
Click here for Linux

When this is done, run the file that you downloaded (.exe) and follow the instructions, it is very intuitive.
When the Arduino IDE is on your Desktop, you can follow the steps just below.

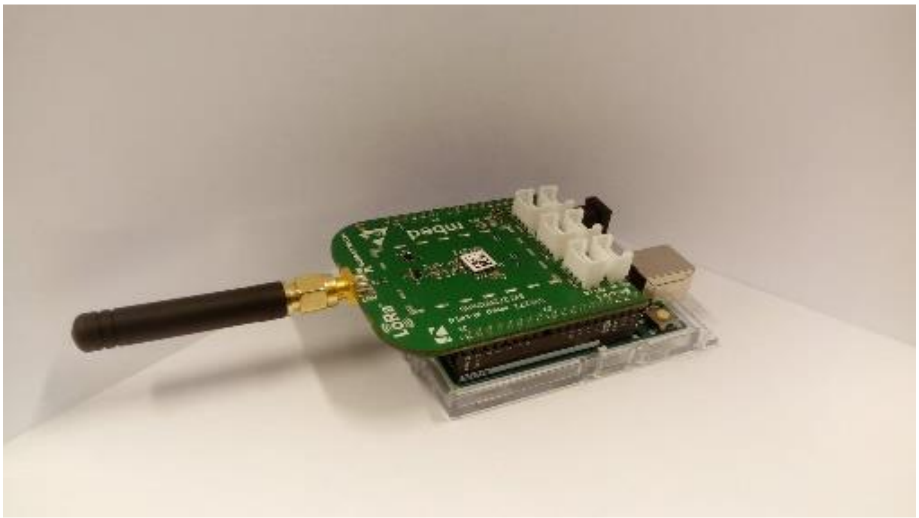
Step 1: Arduino + Shield

Unfortunately, two pins of the SX1272 Embedded shield have to be annulled in order to function with the Arduino Uno Rev 3. This is because the Arduino Uno Rev 3 can only use one I2C protocol at a time, but the SX1272 Embedded shield is expecting to use its two I2C protocols at the same time, thus separating the impulse which results in a voltage drop. This drop dips below what is needed to power the sensors plugged on the SX1272. Confronted to this compatibility issue between two pieces of hardware coming from different companies, we chose this straight and simple solution. So, in order to use our code please desolder or cut these two pins.



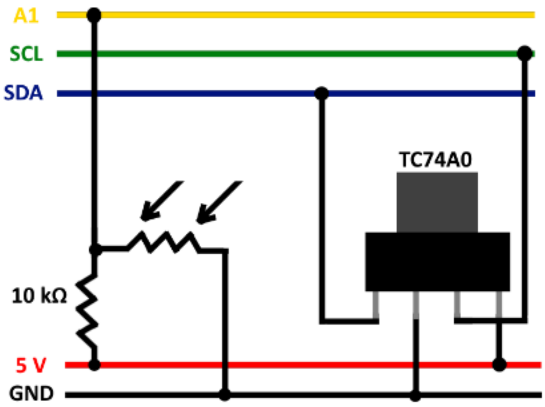


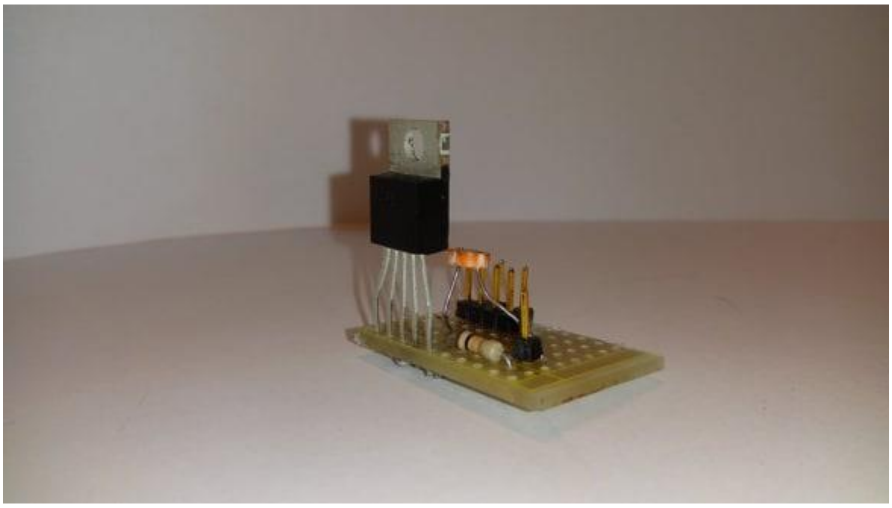
Fit together the Arduino Uno Rev 3 and the SX1272 embedded Shield.



Step 2: Homemade circuit

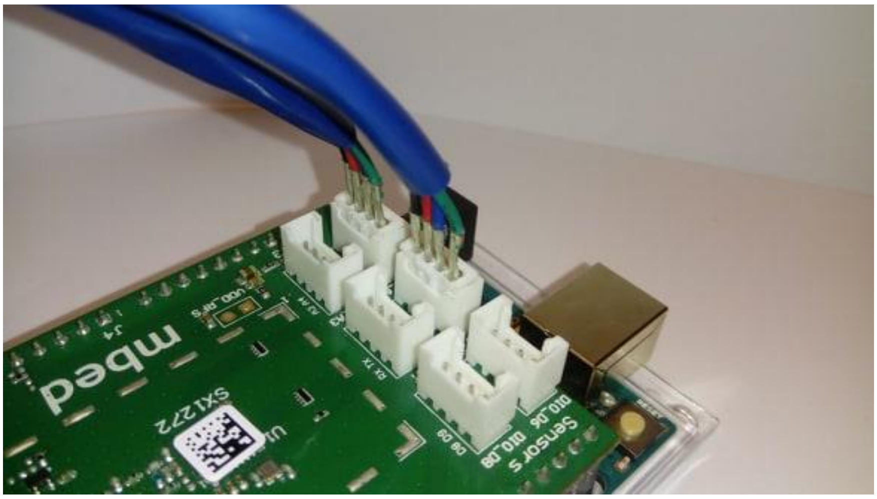
Create this sensors circuit in order to be able to measure brightness and temperature. Light sensor: LDR 12 mm and Thermal sensor: TC74A0.

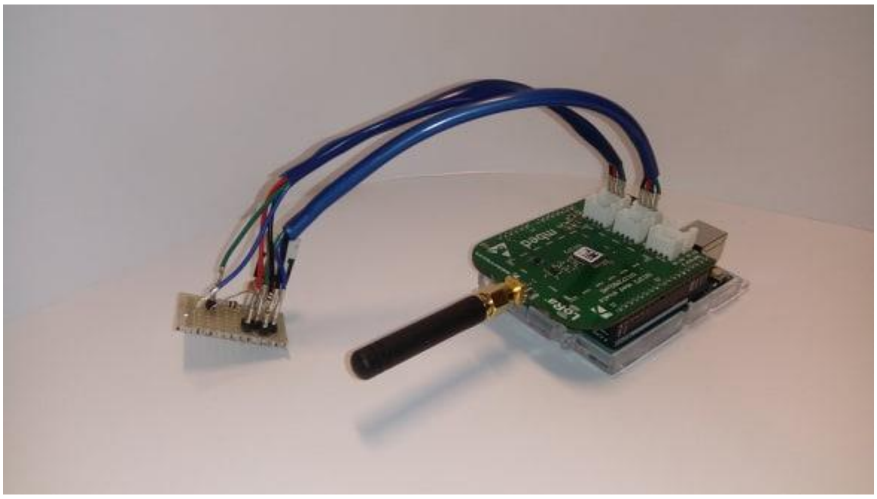




Step 3: Plug the Homemade circuit

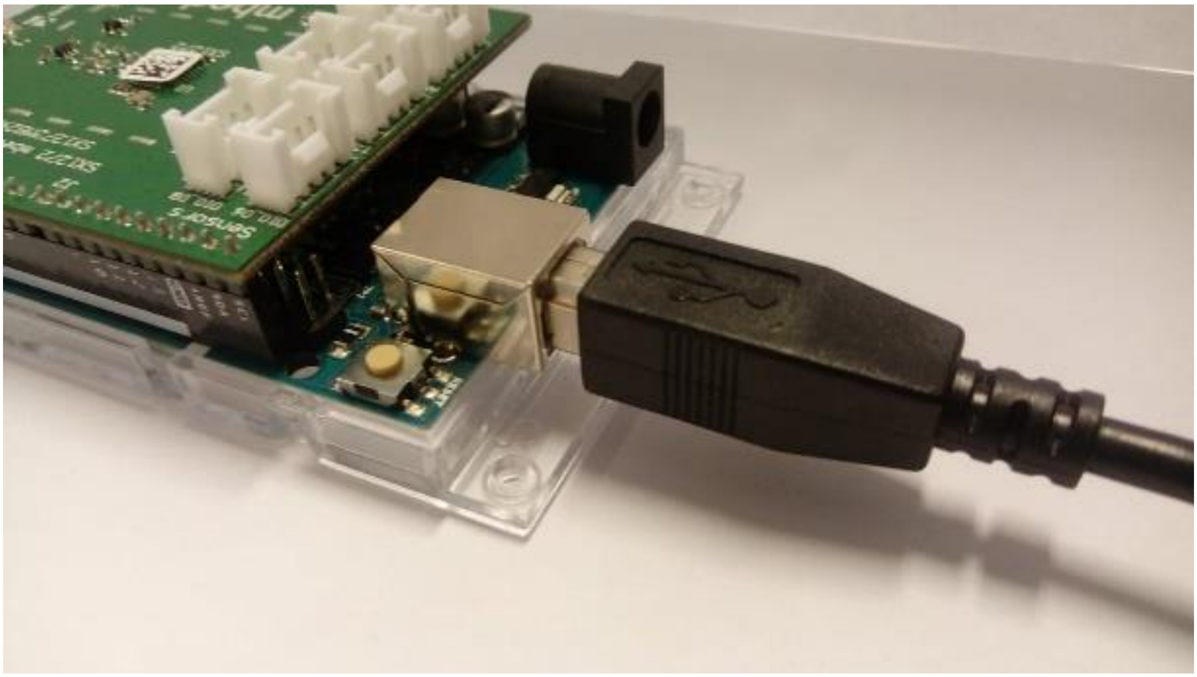
Use wires to plug the sensors circuit board to the SX1272 Embedded shield.





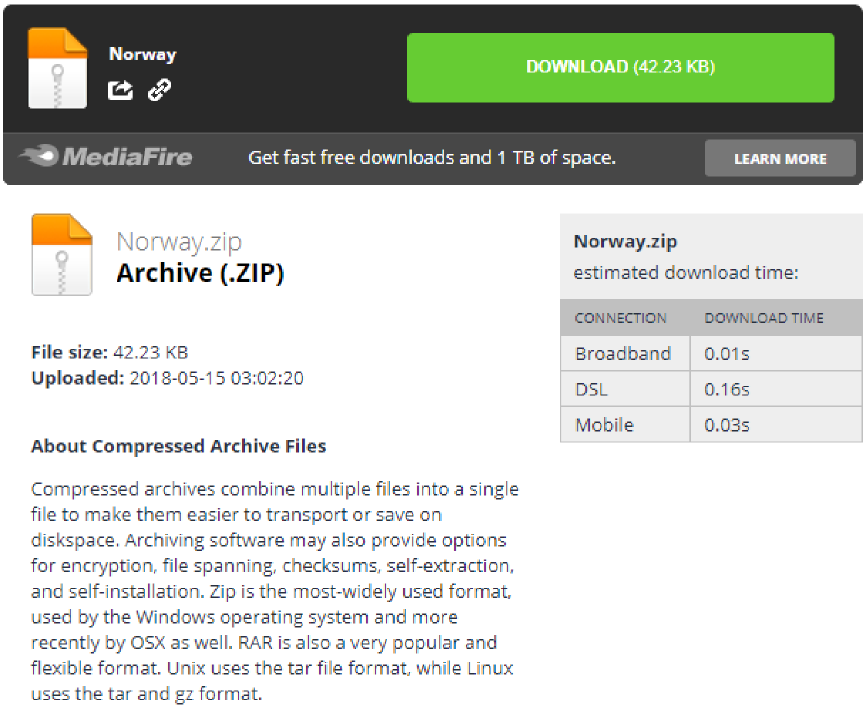
Step 4: Connection between Arduino and a PC

Plug the Wire type A/B into the Arduino Uno Rev 3 and also plug the Arduino Uno Rev 3 into an USB port of your computer.



Step 5: Download the code

Download the Arduino called Norway.



Norway

DOWNLOAD (42.23 KB)

MediaFire Get fast free downloads and 1 TB of space. LEARN MORE

Norway.zip
Archive (.ZIP)

File size: 42.23 KB
Uploaded: 2018-05-15 03:02:20

About Compressed Archive Files

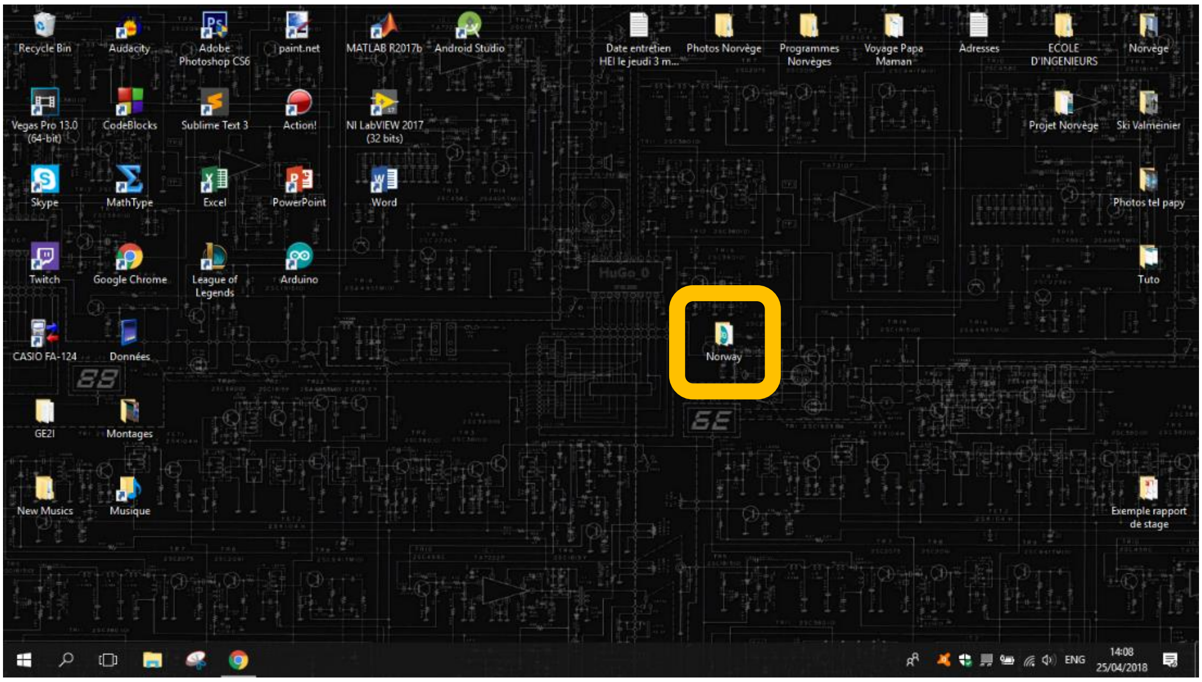
Compressed archives combine multiple files into a single file to make them easier to transport or save on diskspace. Archiving software may also provide options for encryption, file spanning, checksums, self-extraction, and self-installation. Zip is the most-widely used format, used by the Windows operating system and more recently by OSX as well. RAR is also a very popular and flexible format. Unix uses the tar file format, while Linux uses the tar and gz format.

Norway.zip
estimated download time:

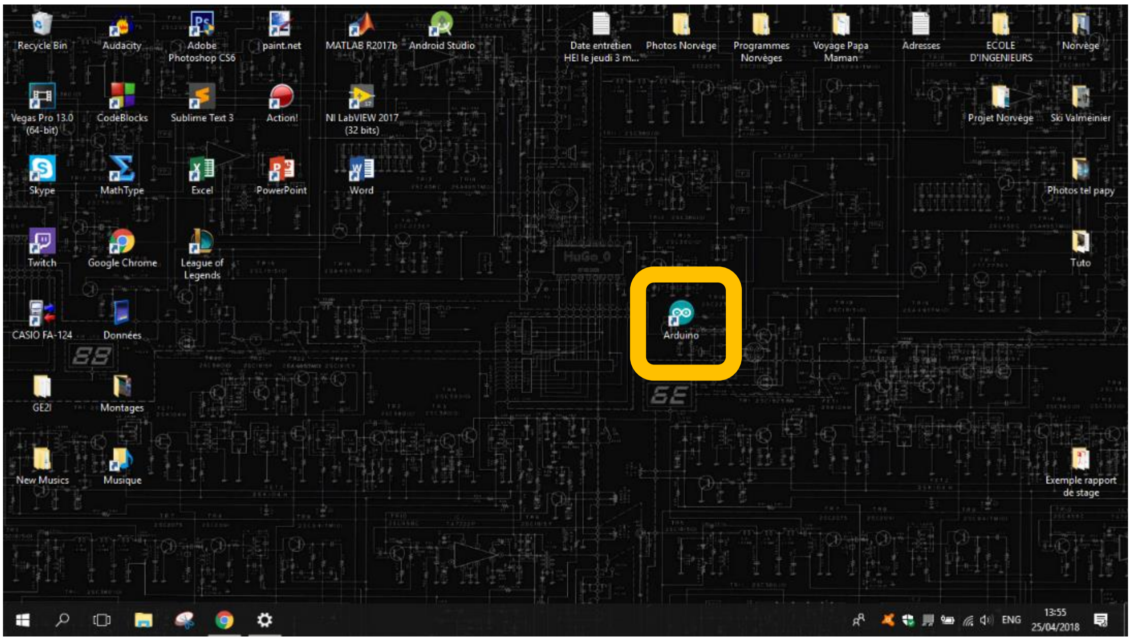
CONNECTION	DOWNLOAD TIME
Broadband	0.01s
DSL	0.16s
Mobile	0.03s

Click here to download

It is a ZIP file so you must extract the folder on your Desktop.

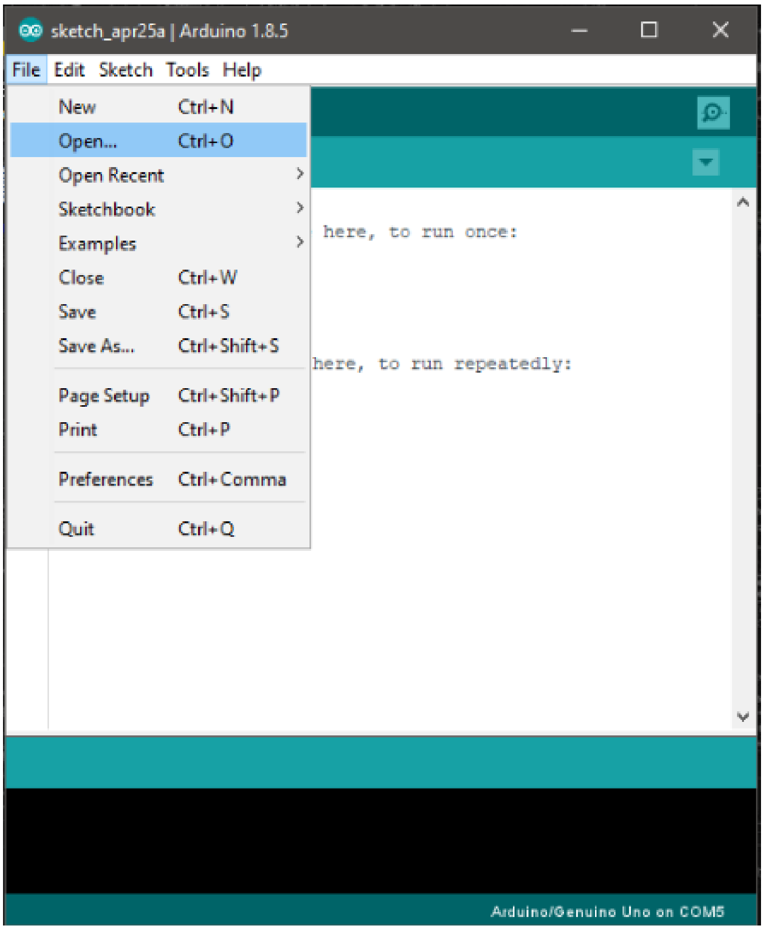


Next, you need to turn on the Arduino IDE.



Step 6: Insert the code into Arduino IDE

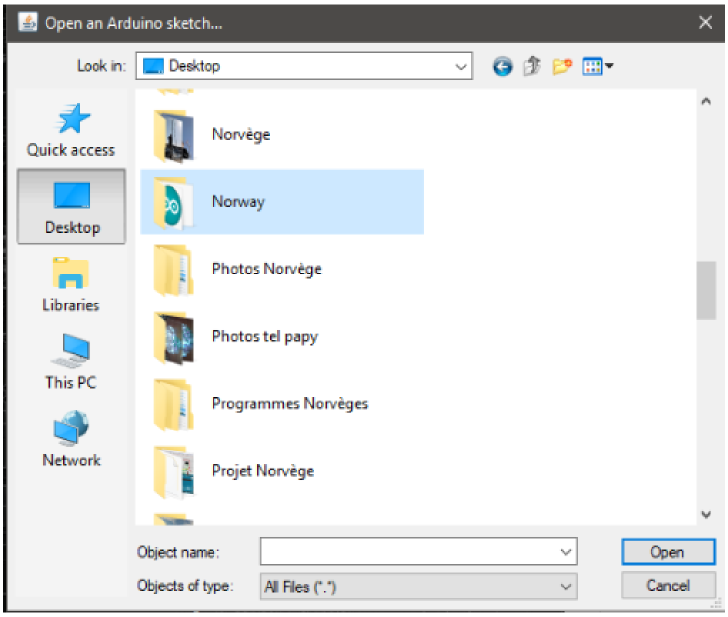
Once in Arduino IDE, go in File and select Open (shortcut: Ctrl + O).



```
 here, to run once:


here, to run repeatedly:
```

Arduino/Genuino Uno on COM5

Now select the file on the Desktop named *Norway*.

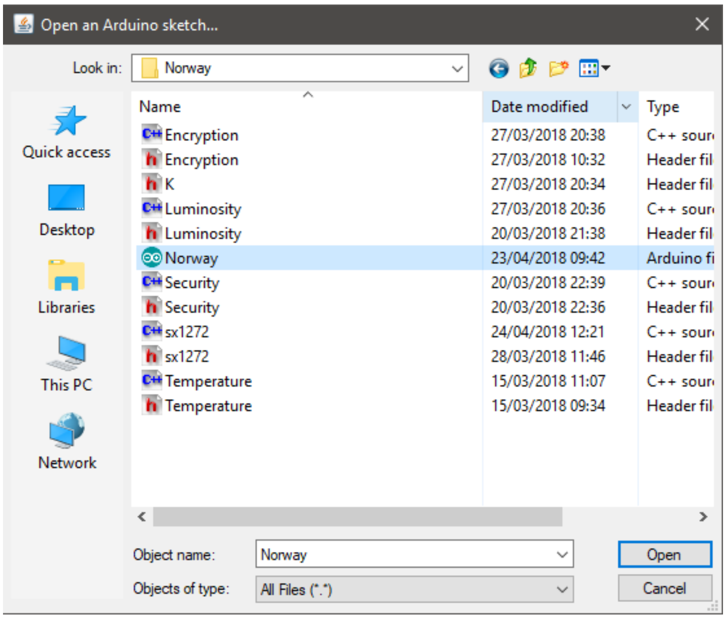


Look in: Desktop

Object name:

Objects of type: All Files (*.*)

Inside, choose the file *Norway.ino* and click on Open.



Look in: Norway

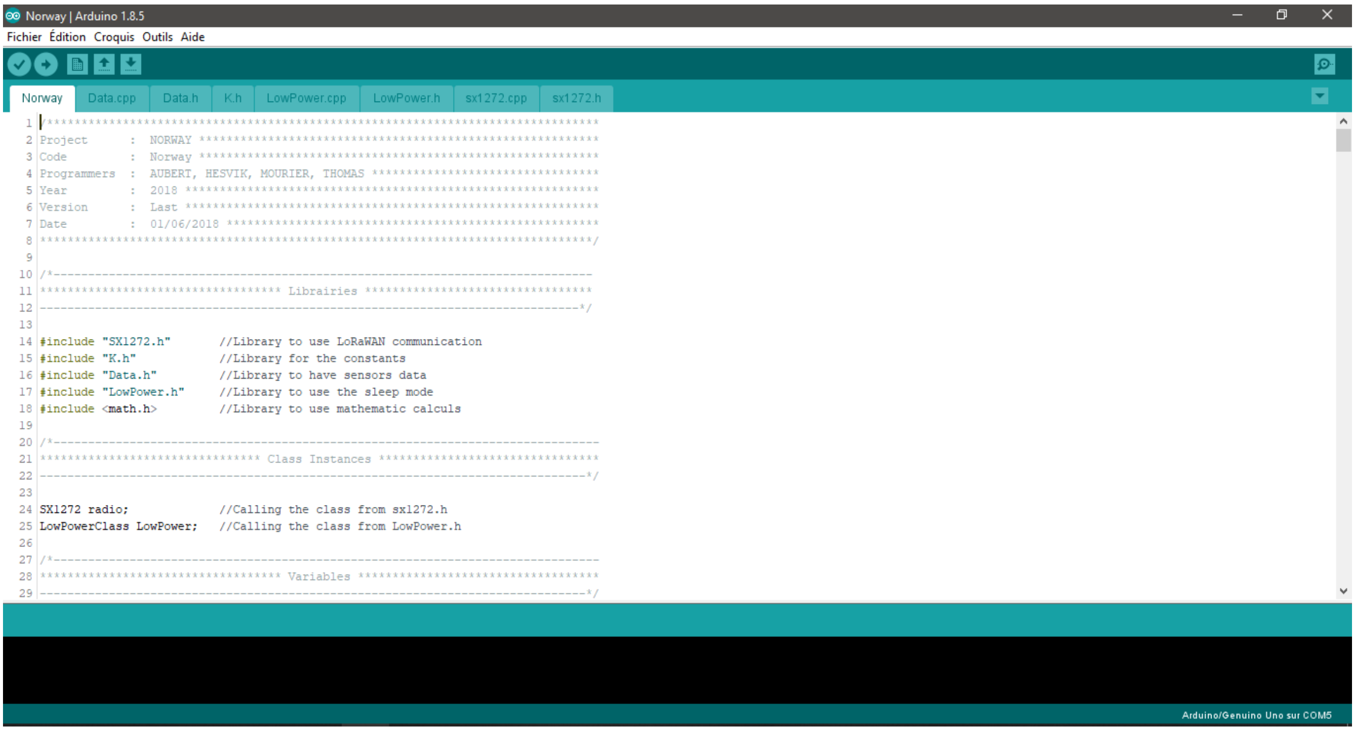
Name	Date modified	Type
Encryption	27/03/2018 20:38	C++ sour
Encryption	27/03/2018 10:32	Header fil
K	27/03/2018 20:34	Header fil
Luminosity	27/03/2018 20:36	C++ sour
Luminosity	20/03/2018 21:38	Header fil
Norway	23/04/2018 09:42	Arduino fi
Security	20/03/2018 22:39	C++ sour
Security	20/03/2018 22:36	Header fil
sx1272	24/04/2018 12:21	C++ sour
sx1272	28/03/2018 11:46	Header fil
Temperature	15/03/2018 11:07	C++ sour
Temperature	15/03/2018 09:34	Header fil

Object name: Norway

Objects of type: All Files (*.*)

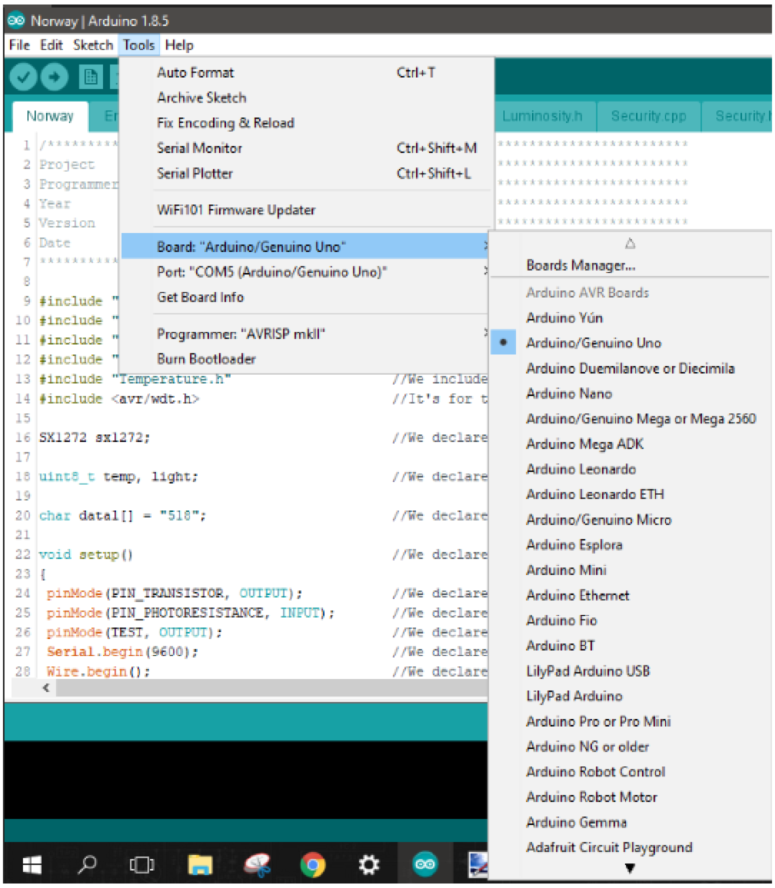
Step 7: Configurations

You should get this page:



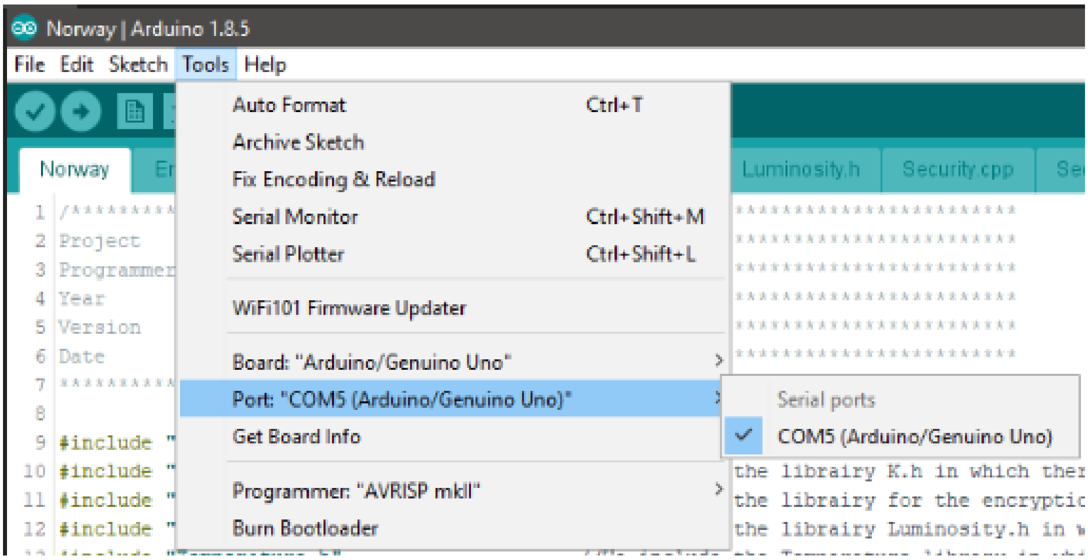
```
/***********************************************************************
Project      :  NORWAY ***************************************************
Code         :  Norway ***************************************************
Programmers  :  AUBERT, HESVIK, MOURIER, THOMAS *************************
Year         :  2018 *****************************************************
Version      :  Last *****************************************************
Date         :  01/06/2018 ***********************************************
**********************************************************************/

/*---------------------------------------------------------------------
******************************* Librairies *****************************
---------------------------------------------------------------------*/

#include "SX1272.h"       //Library to use LoRaWAN communication
#include "K.h"            //Library for the constants
#include "Data.h"         //Library to have sensors data
#include "LowPower.h"     //Library to use the sleep mode
#include <math.h>         //Library to use mathematic calculs

/*----------------------------------------------------------------------
****************************** Class Instances ***************************
----------------------------------------------------------------------*/

SX1272 radio;             //Calling the class from sx1272.h
LowPowerClass LowPower;   //Calling the class from LowPower.h

/*----------------------------------------------------------------------
******************************* Variables *******************************
----------------------------------------------------------------------*/
```

Now, go in Tool, select Board, and chose the type of card: it is an *Arduino/Genuino Uno*.



```
/*********
Project
Programmer
Year
Version
Date
*********

#include "
#include "
#include "
#include "
#include "Temperature.h"                      //We include
#include <avr/wdt.h>                          //It's for t

SX1272 sx1272;                                //We declare

uint8_t temp, light;                          //We declare

char data1[] = "518";                         //We declare

void setup()                                  //We declare
{
 pinMode(PIN_TRANSISTOR, OUTPUT);             //We declare
 pinMode(PIN_PHOTORESISTANCE, INPUT);         //We declare
 pinMode(TEST, OUTPUT);                       //We declare
 Serial.begin(9600);                          //We declare
 Wire.begin();                                //We declare
```

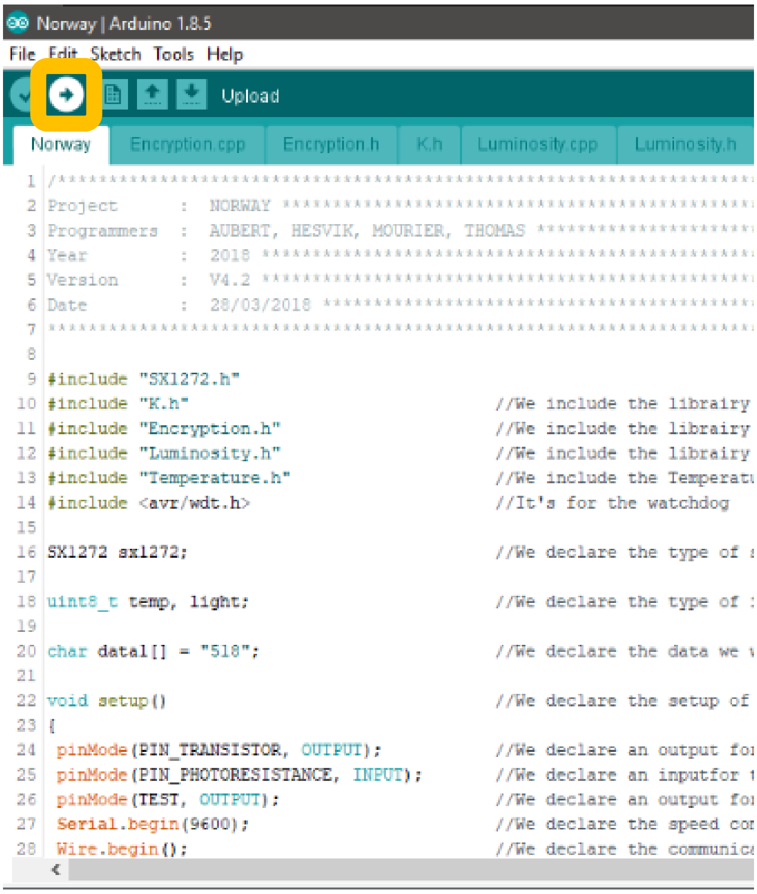
Then, indicate the USB port used by the Arduino: Tool -> Port and select the adequate COM. Most often, it is the first one on the list.


Norway | Arduino 1.8.5
File Edit Sketch Tools Help
Auto Format Ctrl+T
Archive Sketch
Fix Encoding & Reload
Serial Monitor Ctrl+Shift+M
Serial Plotter Ctrl+Shift+L
WiFi101 Firmware Updater
Board: "Arduino/Genuino Uno"
Port: "COM5 (Arduino/Genuino Uno)"
Get Board Info
Programmer: "AVRISP mkII"
Burn Bootloader
Serial ports
COM5 (Arduino/Genuino Uno)
Norway
Luminosity.h
Security.cpp

```
/*********
Project
Programmer
Year
Version
Date
*********

#include "
#include "
#include "
#include "
```

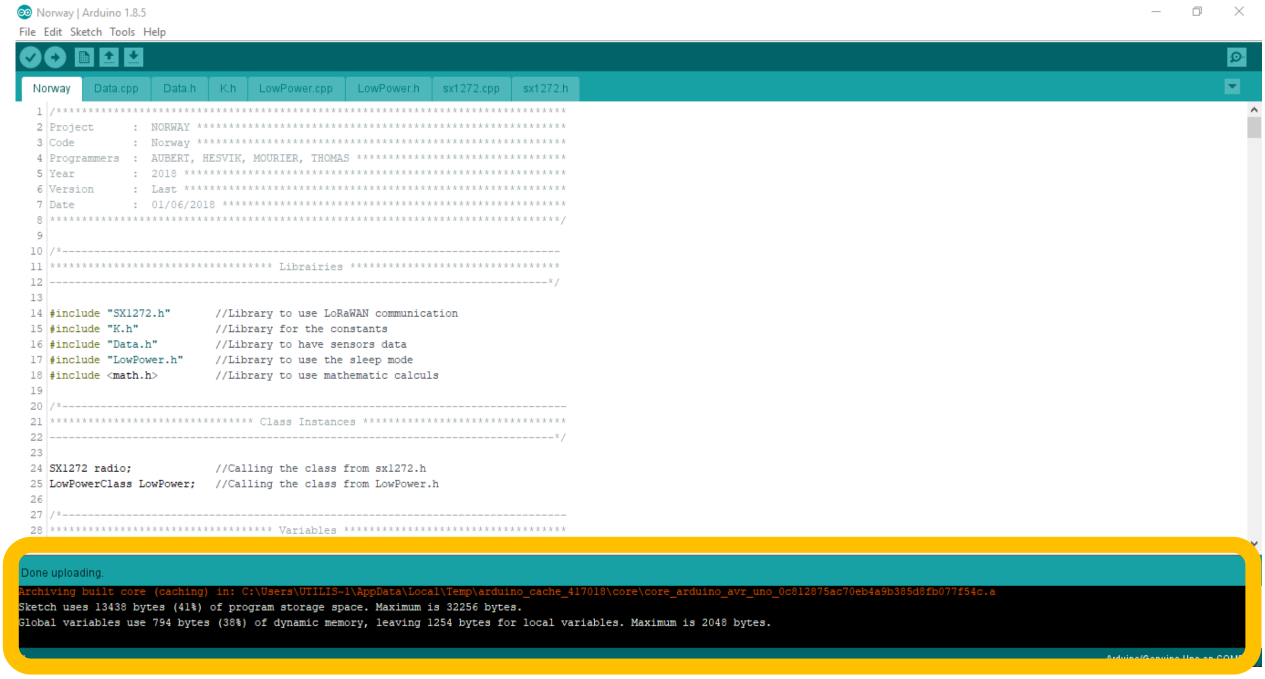
Step 8: Uploading

Now, just upload the code to the Arduino: select Sketch, then Upload, or use the icon surrounded in the picture below.



```
/*************************************************************
Project       :  NORWAY ********************************************
Programmers   :  AUBERT, HESVIK, MOURIER, THOMAS *********************
Year          :  2018 **********************************************
Version       :  V4.2 **********************************************
Date          :  28/03/2018 ****************************************
*************************************************************

#include "SX1272.h"
#include "K.h"                               //We include the librairy
#include "Encryption.h"                      //We include the librairy
#include "Luminosity.h"                      //We include the librairy
#include "Temperature.h"                     //We include the Temperatu
#include <avr/wdt.h>                         //It's for the watchdog

SX1272 sx1272;                               //We declare the type of 

uint8_t temp, light;                         //We declare the type of 

char data1[] = "518";                        //We declare the data we 

void setup()                                 //We declare the setup of
{
 pinMode(PIN_TRANSISTOR, OUTPUT);            //We declare an output fo
 pinMode(PIN_PHOTORESISTANCE, INPUT);        //We declare an inputfor 
 pinMode(TEST, OUTPUT);                      //We declare an output fo
 Serial.begin(9600);                         //We declare the speed co
 Wire.begin();                               //We declare the communic
```

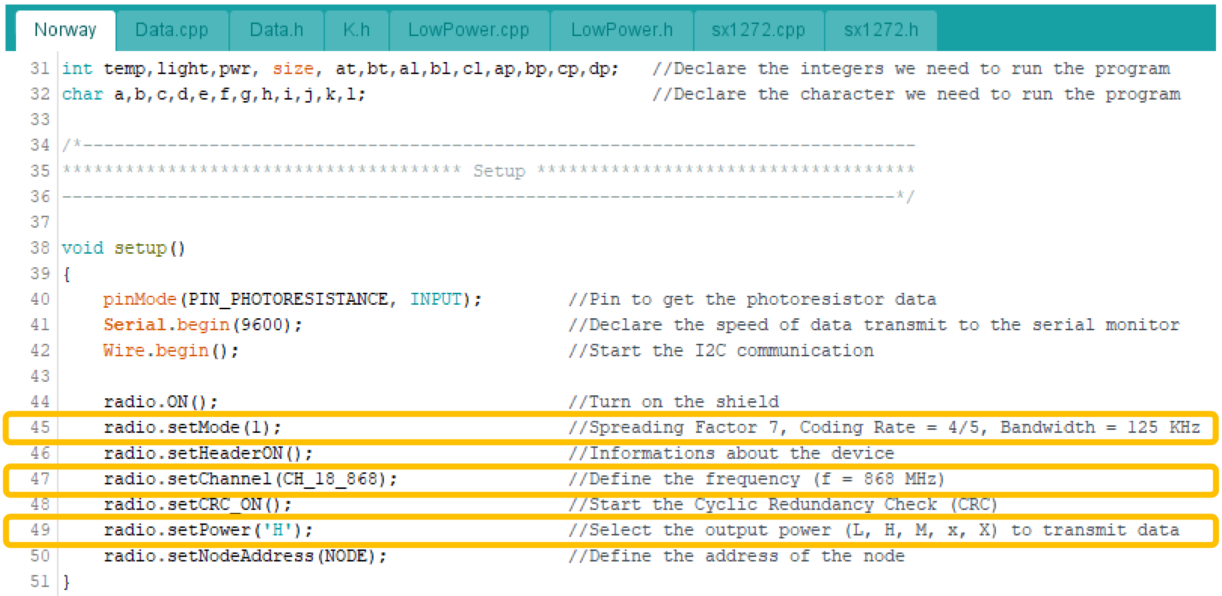
When the upload is done you should have this message “Done uploading”:



```
/*************************************************************************
Project       :  NORWAY ****************************************************
Code          :  Norway ****************************************************
Programmers   :  AUBERT, HESVIK, MOURIER, THOMAS ***************************
Year          :  2018 ******************************************************
Version       :  Last ******************************************************
Date          :  01/06/2018 ************************************************
*************************************************************************/

/*------------------------------------------------------------------------
********************************** Librairies ****************************
------------------------------------------------------------------------*/

#include "SX1272.h"       //Library to use LoRaWAN communication
#include "K.h"            //Library for the constants
#include "Data.h"         //Library to have sensors data
#include "LowPower.h"     //Library to use the sleep mode
#include <math.h>         //Library to use mathematic calculs

/*------------------------------------------------------------------------
******************************* Class Instances **************************
------------------------------------------------------------------------*/

SX1272 radio;             //Calling the class from sx1272.h
LowPowerClass LowPower;   //Calling the class from LowPower.h

/*------------------------------------------------------------------------
********************************** Variables *****************************
```

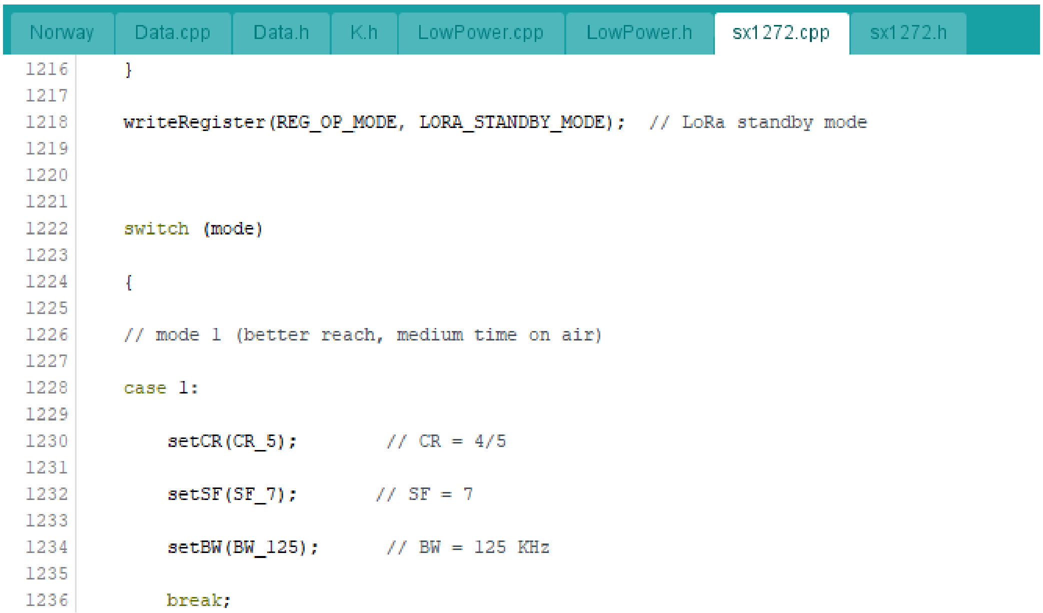
```
Done uploading.
Archiving built core (caching) in: C:\Users\UTILIS~1\AppData\Local\Temp\arduino_cache_417018\core\core_arduino_avr_uno_0c812875ac70eb4a9b385d8fb077f54c.a
Sketch uses 13438 bytes (41%) of program storage space. Maximum is 32256 bytes.
Global variables use 794 bytes (38%) of dynamic memory, leaving 1254 bytes for local variables. Maximum is 2048 bytes.
```

If you want to change things about the LoRaWan transmission, you can modify few things in these yellow rectangles:



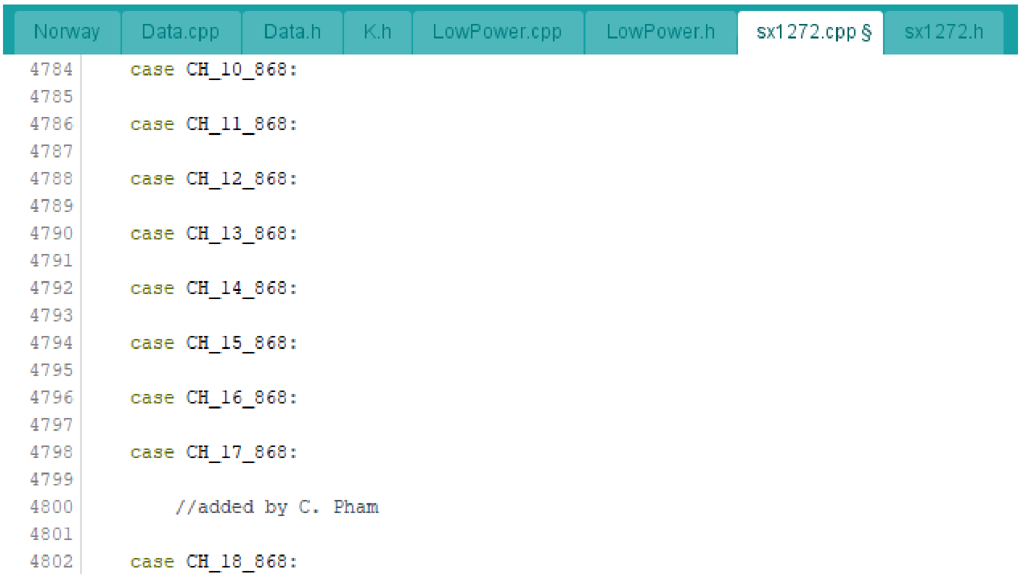
```
int temp,light,pwr, size, at,bt,al,bl,cl,ap,bp,cp,dp;   //Declare the integers we need to run the program
char a,b,c,d,e,f,g,h,i,j,k,l;                           //Declare the character we need to run the program

/*-----------------------------------------------------------------------------
************************************** Setup ***********************************
-----------------------------------------------------------------------------*/

void setup()
{
    pinMode(PIN_PHOTORESISTANCE, INPUT);        //Pin to get the photoresistor data
    Serial.begin(9600);                         //Declare the speed of data transmit to the serial monitor
    Wire.begin();                               //Start the I2C communication

    radio.ON();                                 //Turn on the shield
    radio.setMode(1);                           //Spreading Factor 7, Coding Rate = 4/5, Bandwidth = 125 KHz
    radio.setHeaderON();                        //Informations about the device
    radio.setChannel(CH_18_868);                //Define the frequency (f = 868 MHz)
    radio.setCRC_ON();                          //Start the Cyclic Redundancy Check (CRC)
    radio.setPower('H');                        //Select the output power (L, H, M, x, X) to transmit data
    radio.setNodeAddress(NODE);                 //Define the address of the node
}
```

If, you want more precisions, find these elements in the code:



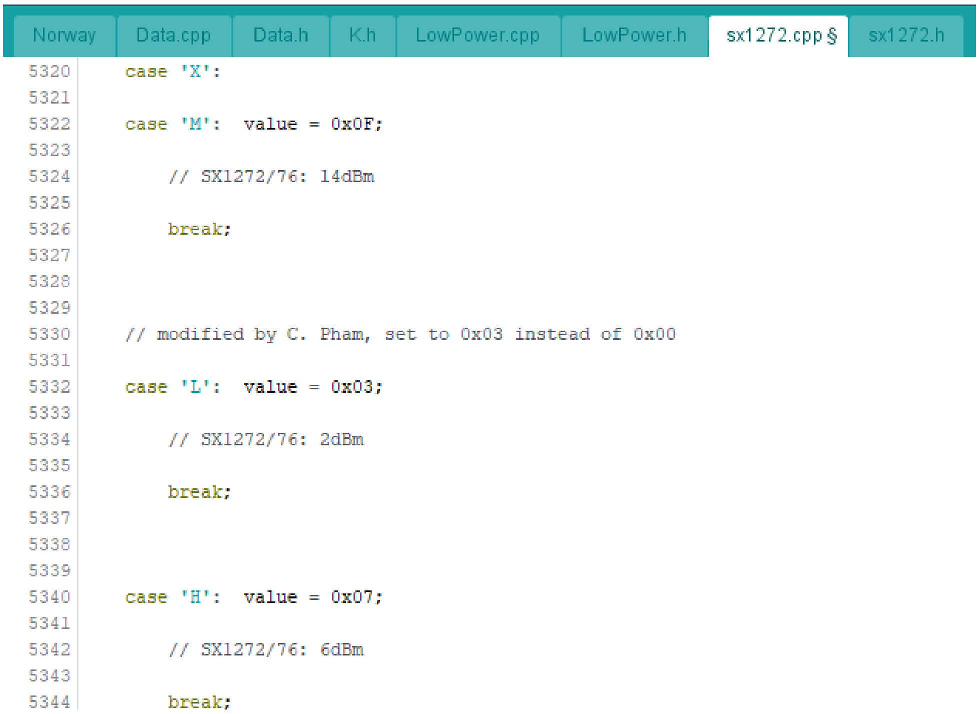
```
}

writeRegister(REG_OP_MODE, LORA_STANDBY_MODE);  // LoRa standby mode



switch (mode)

{

// mode 1 (better reach, medium time on air)

case 1:

    setCR(CR_5);        // CR = 4/5

    setSF(SF_7);       // SF = 7

    setBW(BW_125);      // BW = 125 KHz

    break;
```

The picture above is for the Coding rate, the Spreading Factor and also the Bandwidth.



```
case CH_10_868:

case CH_11_868:

case CH_12_868:

case CH_13_868:

case CH_14_868:

case CH_15_868:

case CH_16_868:

case CH_17_868:

    //added by C. Pham

case CH_18_868:
```

The picture above is for selecting the Frequency.



```
case 'X':

case 'M':  value = 0x0F;

    // SX1272/76: 14dBm

    break;



// modified by C. Pham, set to 0x03 instead of 0x00

case 'L':  value = 0x03;

    // SX1272/76: 2dBm

    break;



case 'H':  value = 0x07;

    // SX1272/76: 6dBm

    break;
```

The picture above is for selecting the power of the signal.

Now that the program is loaded, as it is automated you just need to plug the device (Arduino Uno + the SX1272 Embedded shield + sensor circuit board) to any USB port to power it in order to gather and transmit data.

This program will send data to the Lora gateway.

This tutorial is finished. You should now go read the tutorial about How to install the Raspberry Pi Gateway, in order to receive the LoRa transmitted data.

Thank you for reading.

Aurora project members