|  |
| --- |
| Heltec Automation |
| https://i0.wp.com/heltec.org/wp-content/uploads/2019/09/cubecell_100x258.png?resize=1000%2C258&is-pending-load=1 |
| Getting Started Guide  V1.0 © by WASN.eu |

Summary

[Setup of development Environment 3](#_Toc24538926)

[Download and Install Arduino IDE 3](#_Toc24538927)

[Configure the Arduino IDE 3](#_Toc24538928)

[Loading Firmware 6](#_Toc24538929)

[Arduino DIE compile settings 7](#_Toc24538930)

[The Things Network decoder 9](#_Toc24538931)

# [Setup](https://docs.heltec.cn/#/en/user_manual/how_to_install_ASR650x_Arduino?id=setting-environment) of development Environment

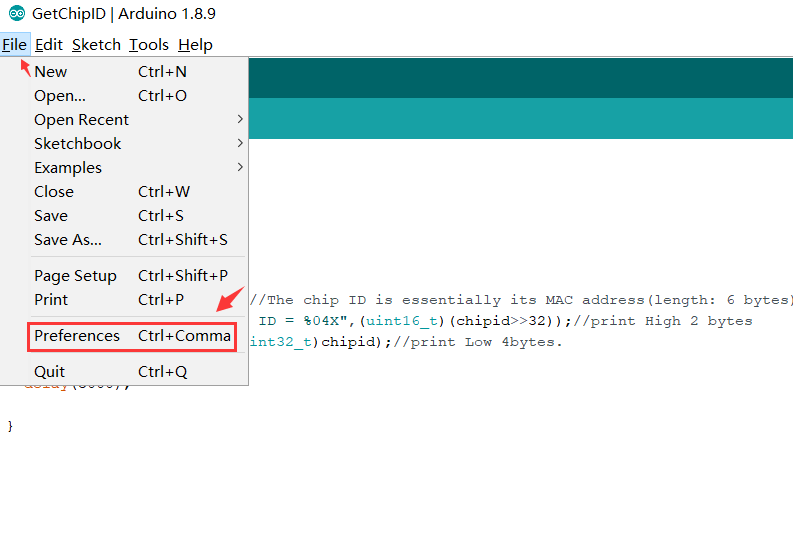
## Download and Install Arduino IDE

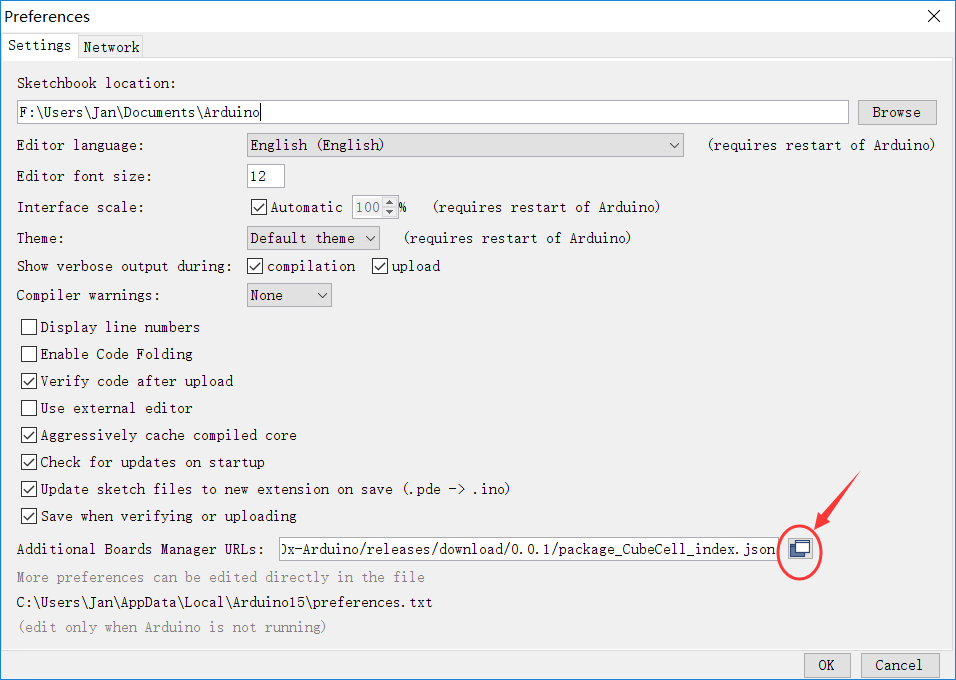
Download and install the Arduino IDE from:

https://www.arduino.cc/en/Main/Software

## [Configure](https://docs.heltec.cn/#/en/user_manual/how_to_install_ASR650x_Arduino?id=step1-download-arduino-asr650x-cubecell-support) the Arduino IDE

* Open Arduino IDE
* click File->Peferences->Settings



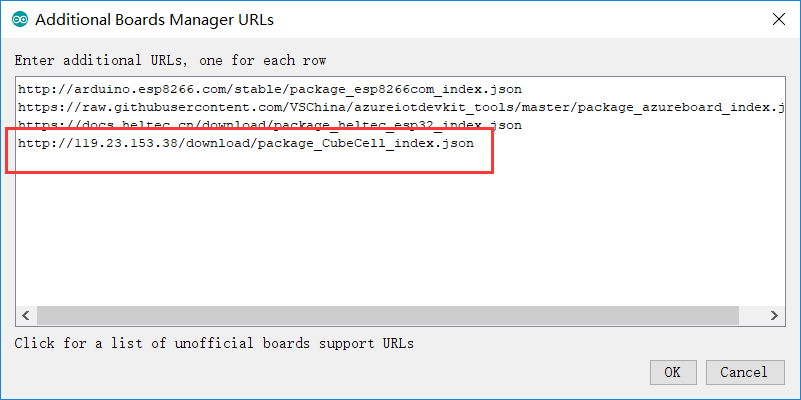


Input last ASR650x board manager URL:

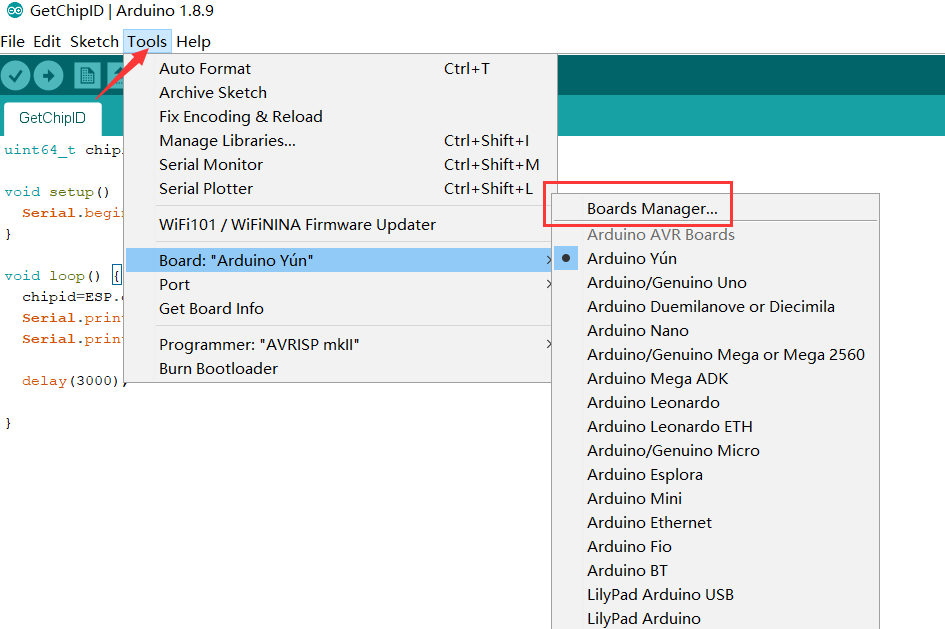
<https://docs.heltec.cn/download/package_CubeCell_index.json>

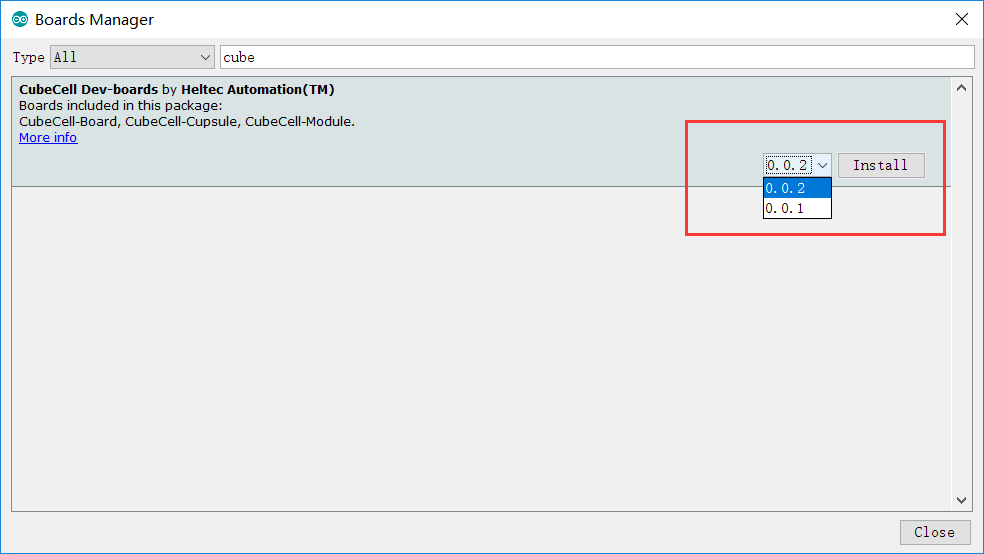
or

<http://119.23.153.38/download/package_CubeCell_index.json>



* Click Tools->Board:->Boards Manager...
* search Heltec cubecell in the new pop-up dialog
* select the latest releases and click install





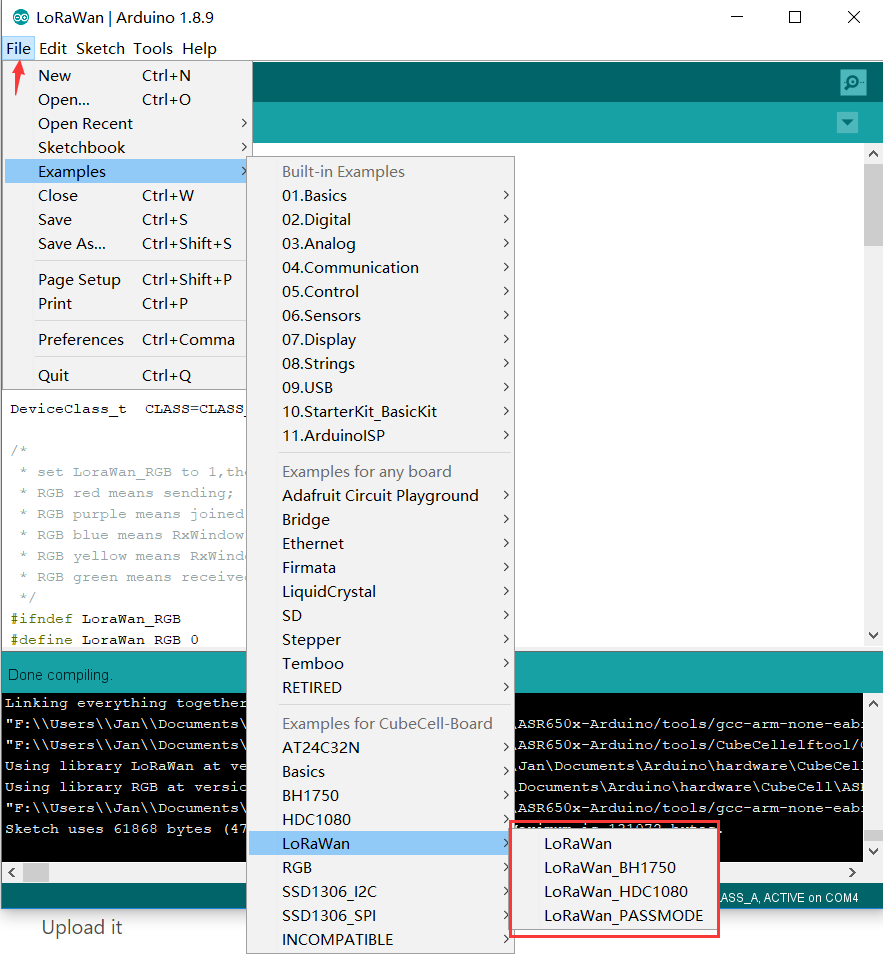
The source code of Heltec ASR650x series (ASR6501&ASR6502) framework is also available here:

<https://github.com/HelTecAutomation/ASR650x-Arduino>

# [Loading](https://docs.heltec.cn/#/en/user_manual/how_to_install_ASR650x_Arduino?id=example) Firmware

Select the example:

File -> examples -> LoRa -> LoRaWan\_Sensors -> LoRaWan\_Multisensor



This sketch allows you to use a number of sensors. All you have to do is to set the number to a 1 beside the sensor you would like to use   
(**only 1 selection is possible, selecting more will result in errors**).

Selecting AUTO\_SCAN the Firmware will try to identify the connected I2C Sensor by its address. AUTO\_SCAN takes its time, it is not very battery friendly.   
Please select the sensor for saving battery power,   
for maximum flexibility choose AUTO\_SCAN.

#define AUTO\_SCAN 1

#define MJMCU\_8128 0

#define BME\_680 0 // wrong values

#define BME\_280 0

#define CCS\_811 0

#define BMP\_180 0 // not tested

#define HDC\_1080 0

#define BH\_1750 0

#define One\_Wire 0 // sensors not found

Login to TTN and create a new device. Get your keys from TTN and set them in the Sketch:

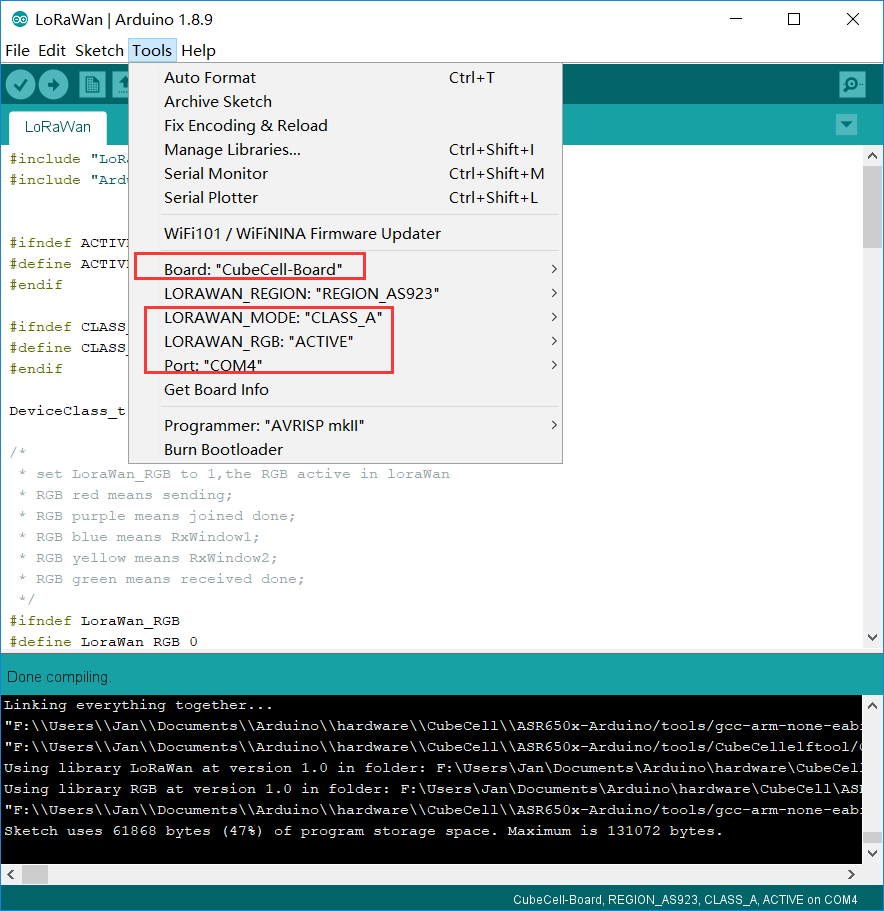
const char myDevEui[] = { 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00 };

const char myAppEui[] = { 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00 };

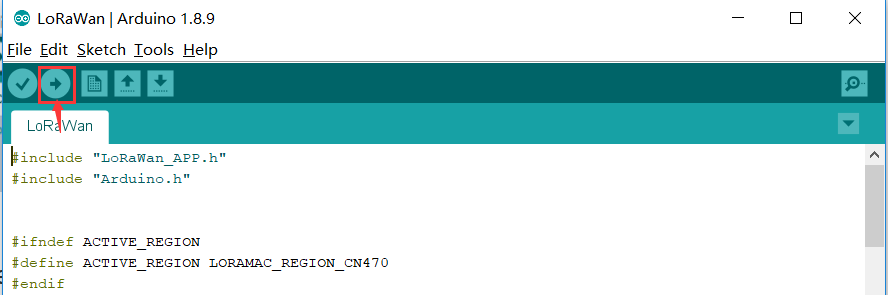
const char myAppKey[] = { 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x0

## [Arduino](https://docs.heltec.cn/#/en/user_manual/how_to_install_ASR650x_Arduino?id=_1-execute-a-example-likes-lorawanino) DIE compile settings

Correct select the board you had connected to computer and the specified serial port: e.g. CubeCell-Board, CubeCell-Module, CubeCell-Cupsule, COM-Port



Compile and upload it



# The Things Network decoder

Login to TTN and define the decoder.

The decoder is found in the example directory.

ttn-decoder.js:

|  |
| --- |
| function bytesToFloat(by) {  var bits = by[3]<<24 | by[2]<<16 | by[1]<<8 | by[0];  var sign = (bits>>>31 === 0) ? 1.0 : -1.0;  var e = bits>>>23 & 0xff;  var m = (e === 0) ? (bits & 0x7fffff)<<1 : (bits & 0x7fffff) | 0x800000;  var f = sign \* m \* Math.pow(2, e - 150);  return f;  }  function Decoder(bytes, port) {  // Decode an uplink message from a buffer  // (array) of bytes to an object of fields.  var decoded = {};  if (port === 2) {  var i = 1;  sensor = bytes[0].toFixed(0);  decoded.sensortype = sensor;  if (sensor === "0") { // MJMCU-8128  decoded.temperature = ((((bytes[i++] << 8) | bytes[i++]).toFixed(0) / 10) - 100).toFixed(1);  decoded.humidity = (((bytes[i++] << 8) | bytes[i++]).toFixed(0) / 10).toFixed(1);  decoded.pressure = (((bytes[i++] << 8) | bytes[i++]).toFixed(0) / 10).toFixed(1);  decoded.lux = (((bytes[i++] << 8) | bytes[i++]).toFixed(0) / 10).toFixed(1);  decoded.co2 = ((bytes[i++] << 8) | bytes[i++]).toFixed(0);  decoded.tvoc = ((bytes[i++] << 8) | bytes[i++]).toFixed(0);  decoded.battery = ((bytes[i++] << 8) | bytes[i++]).toFixed(0);  }  if (sensor === "1") { // BME680  decoded.temperature = ((((bytes[i++] << 8) | bytes[i++]).toFixed(0) / 10) - 100).toFixed(1);  decoded.humidity = (((bytes[i++] << 8) | bytes[i++]).toFixed(0) / 10).toFixed(1);  decoded.pressure = (((bytes[i++] << 8) | bytes[i++]).toFixed(0) / 10).toFixed(1);  decoded.gas = (((bytes[i++] << 8) | bytes[i++]).toFixed(0) / 10).toFixed(1);  decoded.battery = ((bytes[i++] << 8) | bytes[i++]).toFixed(0);  }  if (sensor === "2") { // BME280  decoded.temperature = ((((bytes[i++] << 8) | bytes[i++]).toFixed(0) / 10) - 100).toFixed(1);  decoded.humidity = (((bytes[i++] << 8) | bytes[i++]).toFixed(0) / 10).toFixed(1);  decoded.pressure = (((bytes[i++] << 8) | bytes[i++]).toFixed(0) / 10).toFixed(1);  decoded.battery = ((bytes[i++] << 8) | bytes[i++]).toFixed(0);  }  if (sensor === "3") { // CCS811  decoded.co2 = ((bytes[i++] << 8) | bytes[i++]).toFixed(0);  decoded.tvoc = ((bytes[i++] << 8) | bytes[i++]).toFixed(0);  decoded.battery = ((bytes[i++] << 8) | bytes[i++]).toFixed(0);  }  if (sensor === "4") { // HDC1080  decoded.temperature = ((((bytes[i++] << 8) | bytes[i++]).toFixed(0) / 10) - 100).toFixed(1);  decoded.humidity = (((bytes[i++] << 8) | bytes[i++]).toFixed(0) / 10).toFixed(1);  decoded.battery = ((bytes[i++] << 8) | bytes[i++]).toFixed(0);  }  if (sensor === "5") { // BMP180  decoded.temperature = ((((bytes[i++] << 8) | bytes[i++]).toFixed(0) / 10) - 100).toFixed(1);  decoded.pressure = (((bytes[i++] << 8) | bytes[i++]).toFixed(0) / 10).toFixed(1);  decoded.battery = ((bytes[i++] << 8) | bytes[i++]).toFixed(0);  }  if (sensor === "6") { // BH1750  decoded.lux = (((bytes[i++] << 8) | bytes[i++]).toFixed(0) / 10).toFixed(1);  decoded.battery = ((bytes[i++] << 8) | bytes[i++]).toFixed(0);  }  }  return decoded;  } |