

By Bread Boards & Bill

Project # 1 Scanning Radio AKA Ghost Box

January 16 2022

About this Project

- ❖ This Project is presented by Bill Chappell Build at your own risk
- ❖ Author is not responsible for errors or omissions in this document.
- ❖ This project uses open source software “[arduino.org](https://www.arduino.org)”
- ❖ An Arduino UNO cost \$9-14
- ❖ Radio Module cost \$4-5
- ❖ Hook up wire cost \$3-5
- ❖ The total cost to get this project running should be @20.00
- ❖ Purchase links can be found at www.digitaldowsing.com/diy/

About this Project

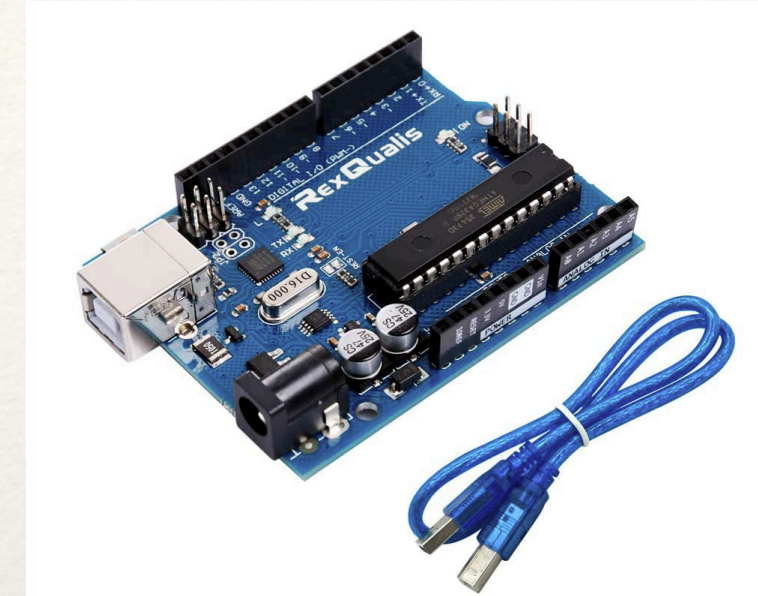
- ❖ The project will start with a simple build not even a bread board
- ❖ Over several weeks many features will be added
- ❖ Software will be kept very simple to allow everyone to understand it
- ❖ Several posts will be about the program, the software and how to use it
- ❖ example and complete code will be available at
- ❖ www.digitaldowsing.com/diy

About this Project

- ❖ Features to be included in this project
- ❖ Full scanning control, up / down , energy driven scan. Scan Increments
- ❖ Speak when spoken to feature “radio responds after you ask something”
- ❖ Serial display as well as a full display
- ❖ When complete the “radio” can run without a computer or additional device

Materials Needed for this Project

Arduino Uno [Arduino Uno](#)



Arduino Radio Module [Radio Module](#)



Hook Up Wire [Hook Up Wire](#)



Software

Download Arduino Software , from www.Arduino.org
Choose the IDE based on the type of computer it will run on
Download and start Arduino IDE

[HARDWARE](#) [SOFTWARE](#) [CLOUD](#) [DOCUMENTATION](#) [COMMUNITY](#) [BLOG](#) [ABOUT](#)

Downloads



Arduino IDE 1.8.19

The open-source Arduino Software (IDE) makes it easy to write code and upload it to the board. This software can be used with any Arduino board.

Refer to the [Getting Started](#) page for Installation instructions.

SOURCE CODE

Active development of the Arduino software is [hosted by GitHub](#). See the instructions for [building the code](#). Latest release source code archives are available [here](#). The archives are PGP-signed so they can be verified using [this](#) gpg key.

DOWNLOAD OPTIONS

Windows Win 7 and newer
Windows ZIP file

Windows app Win 8.1 or 10

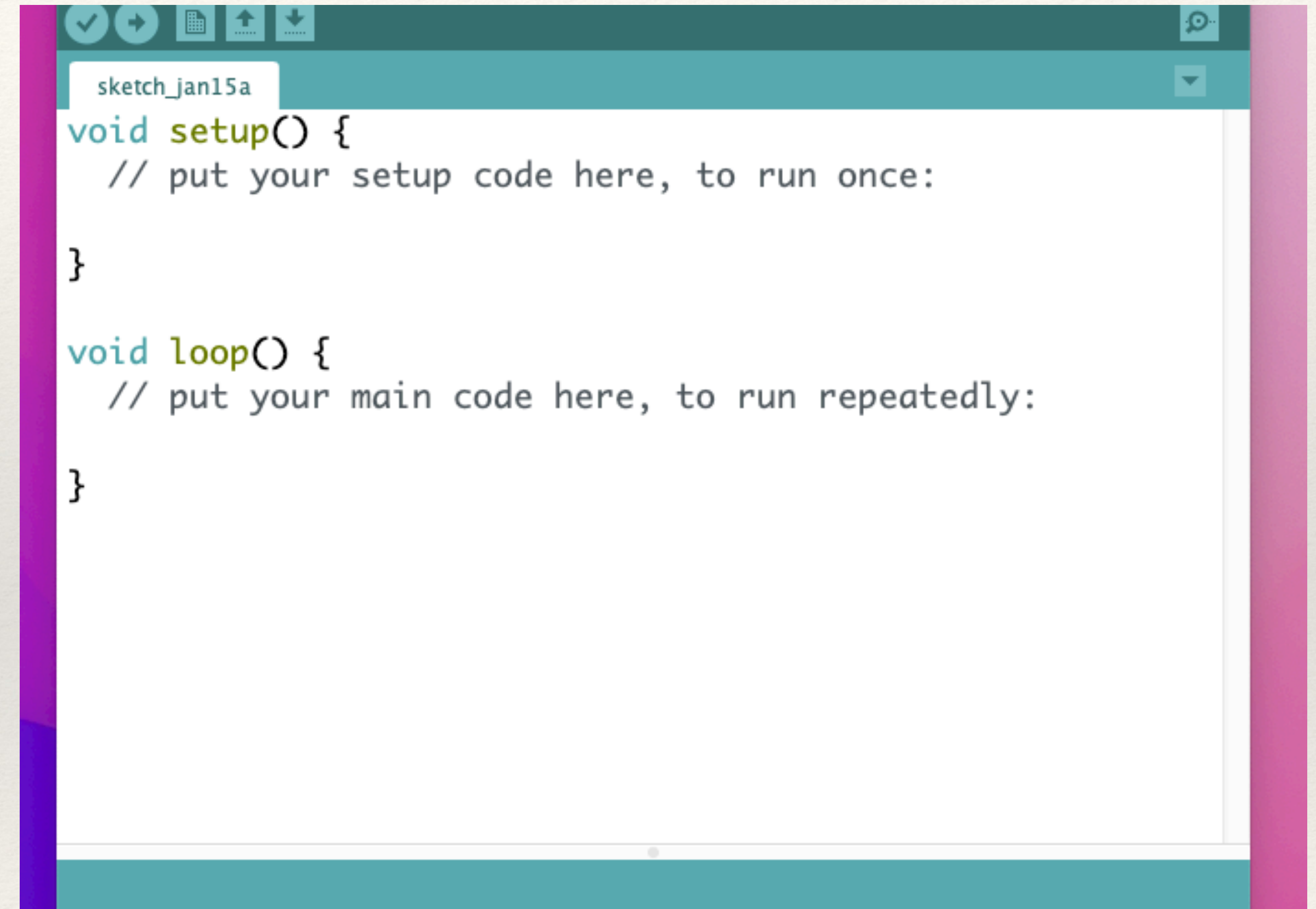
Linux 32 bits
Linux 64 bits
Linux ARM 32 bits
Linux ARM 64 bits

Mac OS X 10.10 or newer

[Release Notes](#) [Checksums \(sha512\)](#)

Software

Run the Arduino Software.
Your Screen should look something like this.



```
sketch_jan15a
void setup() {
  // put your setup code here, to run once:
}

void loop() {
  // put your main code here, to run repeatedly:
}
```


Software

If you're having trouble, Try searching on
Installing Arduino Software.
There are 1000's of online resources to help.

[HARDWARE](#) [SOFTWARE](#) [CLOUD](#) [DOCUMENTATION](#) [COMMUNITY](#) [BLOG](#) [ABOUT](#)

Downloads



Arduino IDE 1.8.19

The open-source Arduino Software (IDE) makes it easy to write code and upload it to the board. This software can be used with any Arduino board.

Refer to the [Getting Started](#) page for Installation instructions.

SOURCE CODE

Active development of the Arduino software is [hosted by GitHub](#). See the instructions for [building the code](#). Latest release source code archives are available [here](#). The archives are PGP-signed so they can be verified using [this](#) gpg key.

DOWNLOAD OPTIONS

Windows Win 7 and newer
Windows ZIP file

Windows app Win 8.1 or 10

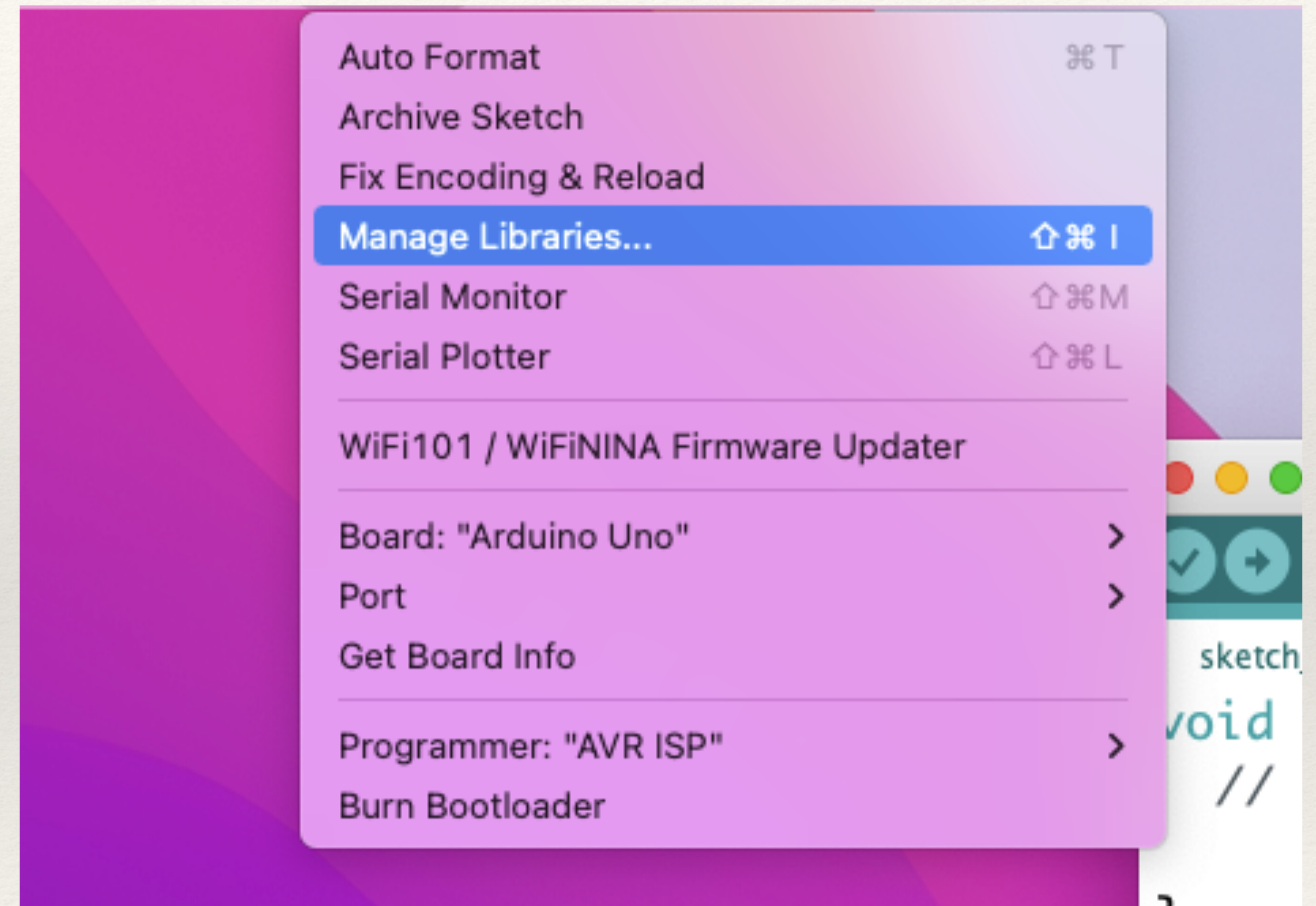
Linux 32 bits
Linux 64 bits
Linux ARM 32 bits
Linux ARM 64 bits

Mac OS X 10.10 or newer

[Release Notes](#) [Checksums \(sha512\)](#)

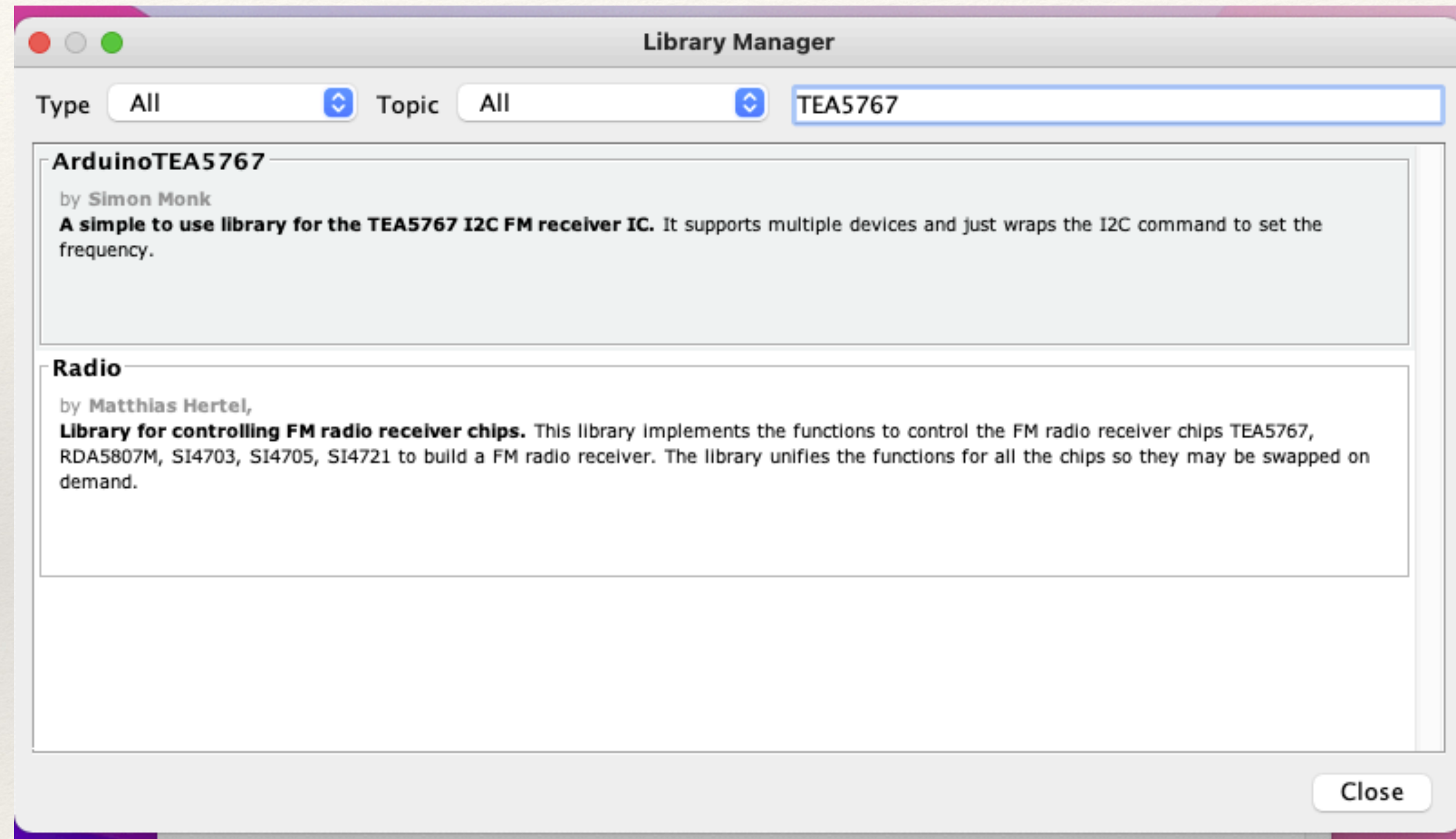
Software

Before we can Write code for the project
A couple libraries need to be installed .
From the Tools Menu Select Manage Libraries

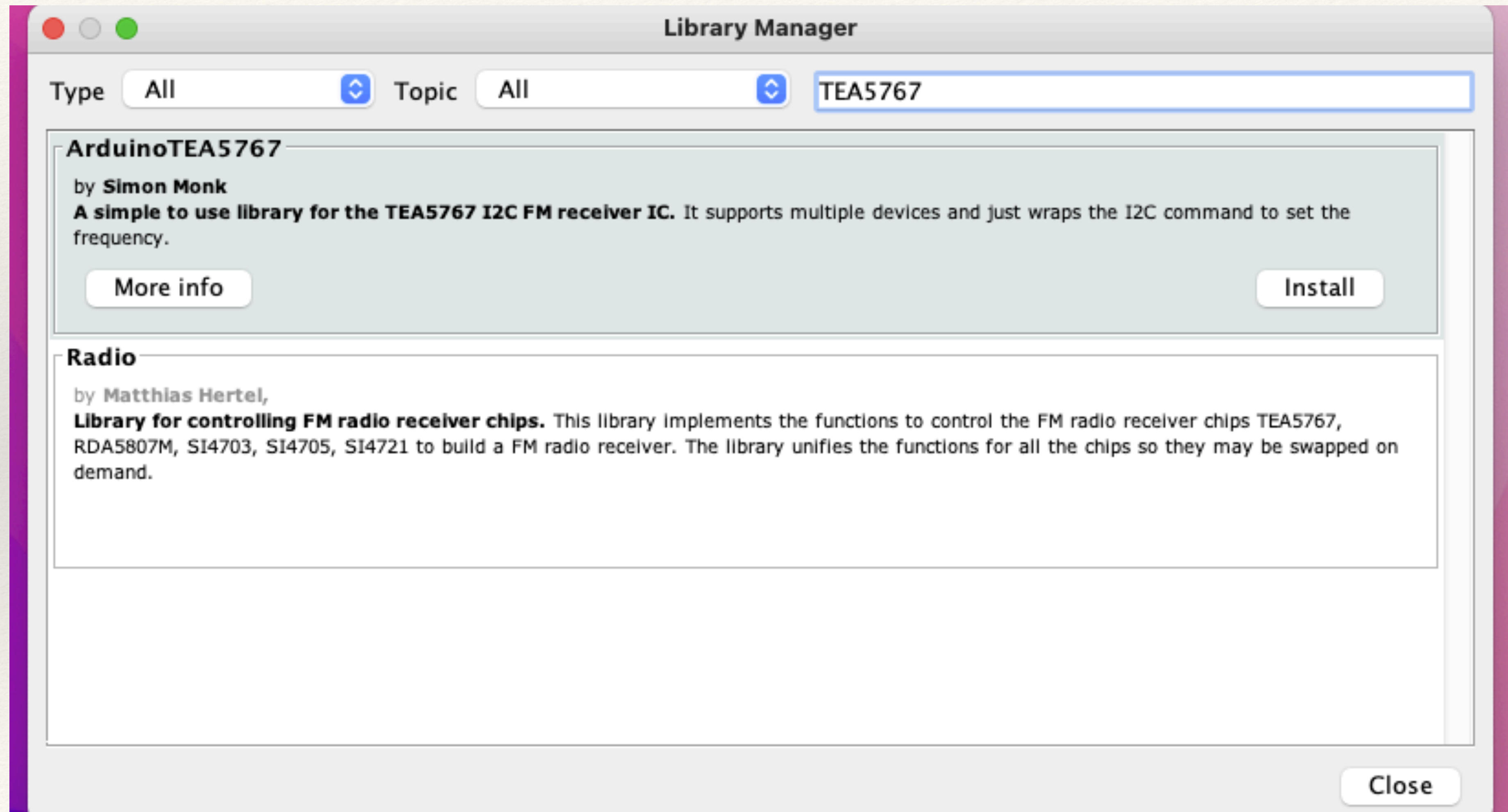


Software

When the Library Manager window appears type TEA5767 in the search bar. The Window will show a couple libraries Click on ArduinoTEA5767



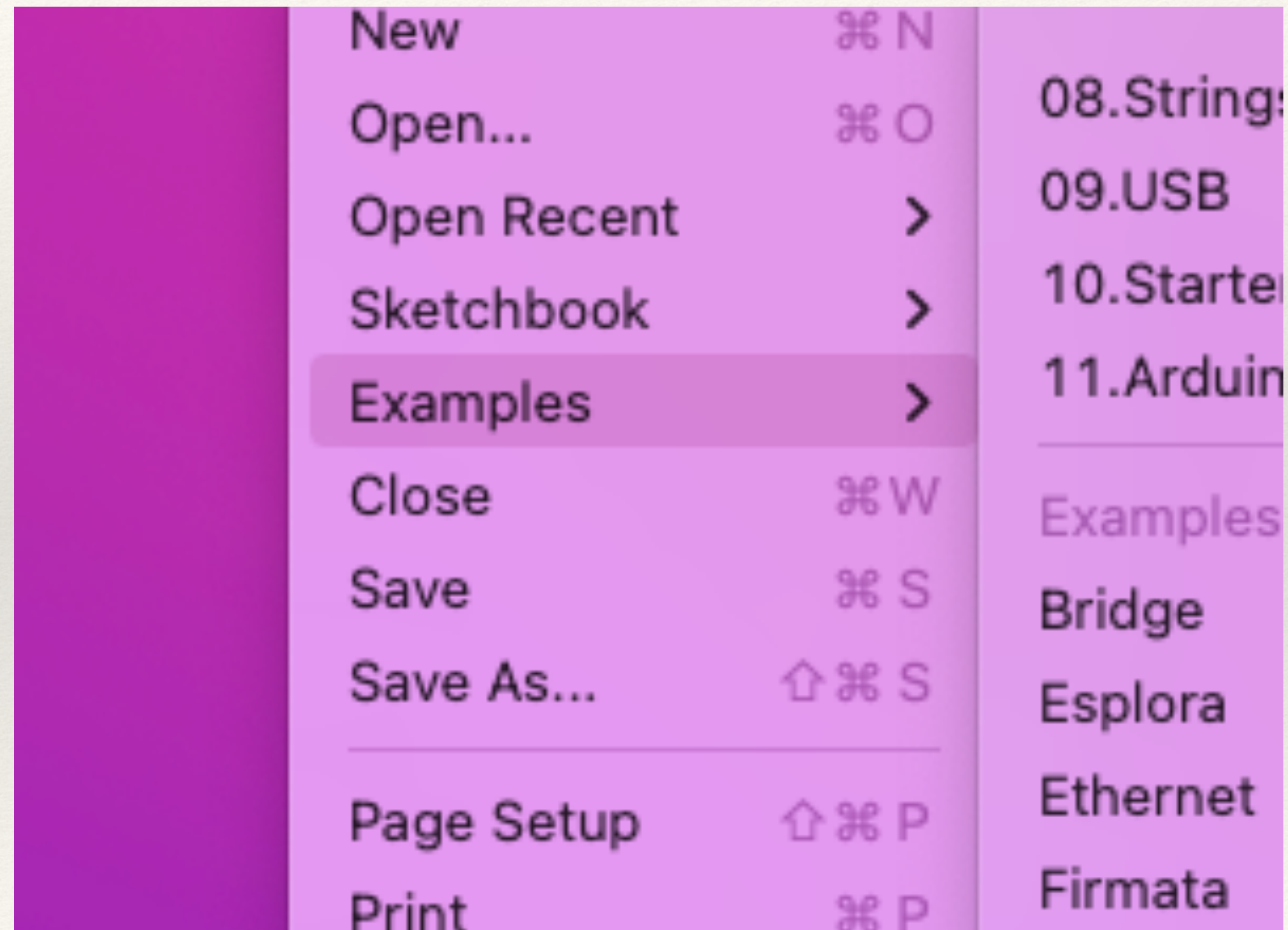
Software



Click on ArduinoTEA5767 then Click on Install when done Click close.

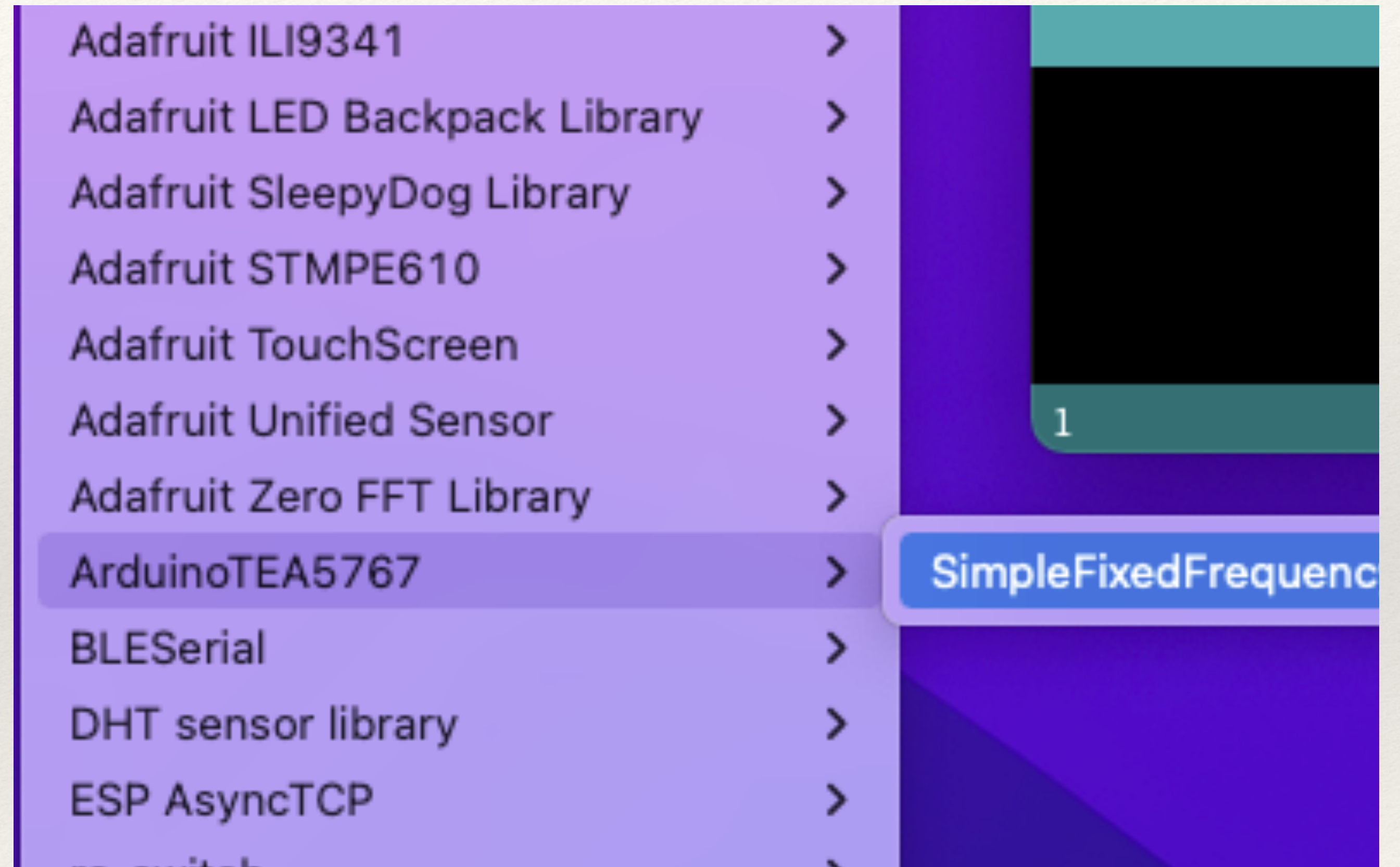
Software

From the File Menu Click on Examples



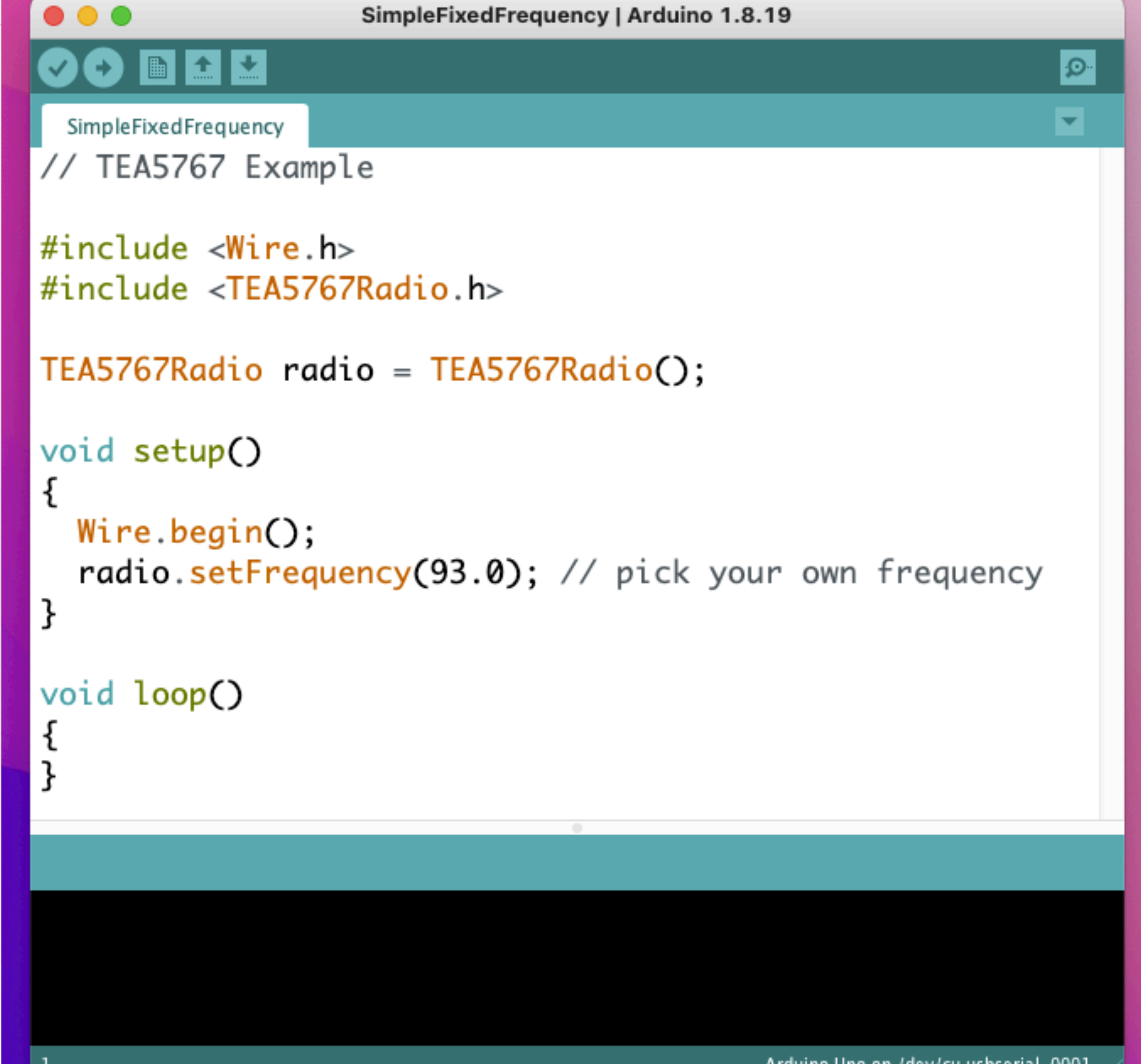
Software

Scroll down to ArduinoTea5767 ,Select
SimpleFixedFrequency



Software

The Arduino IDE Editor should open this file.
You have the basic software installed.
We will return to this after the build



```
SimpleFixedFrequency | Arduino 1.8.19

// TEA5767 Example

#include <Wire.h>
#include <TEA5767Radio.h>

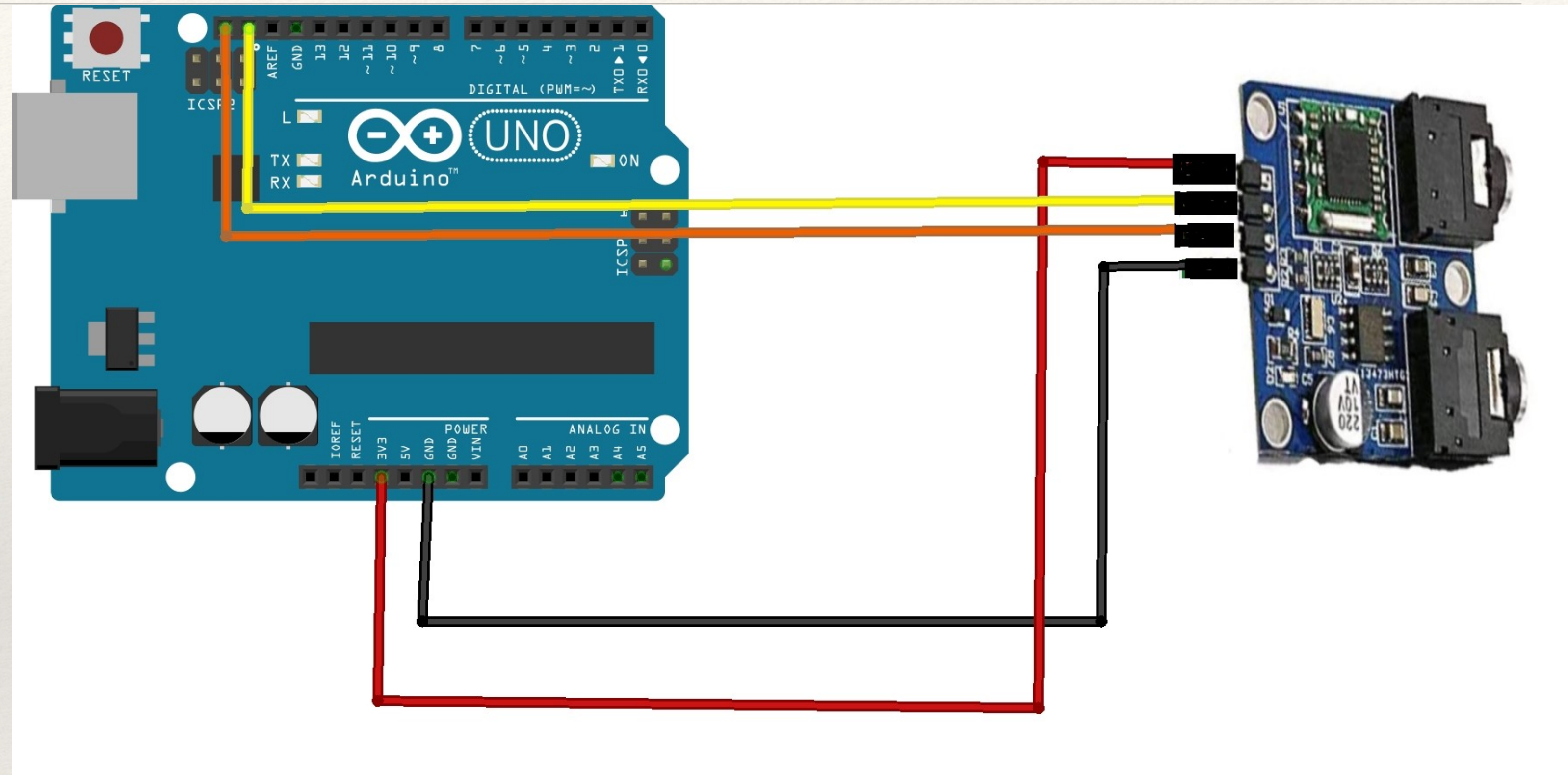
TEA5767Radio radio = TEA5767Radio();

void setup()
{
  Wire.begin();
  radio.setFrequency(93.0); // pick your own frequency
}

void loop()
{
}
```

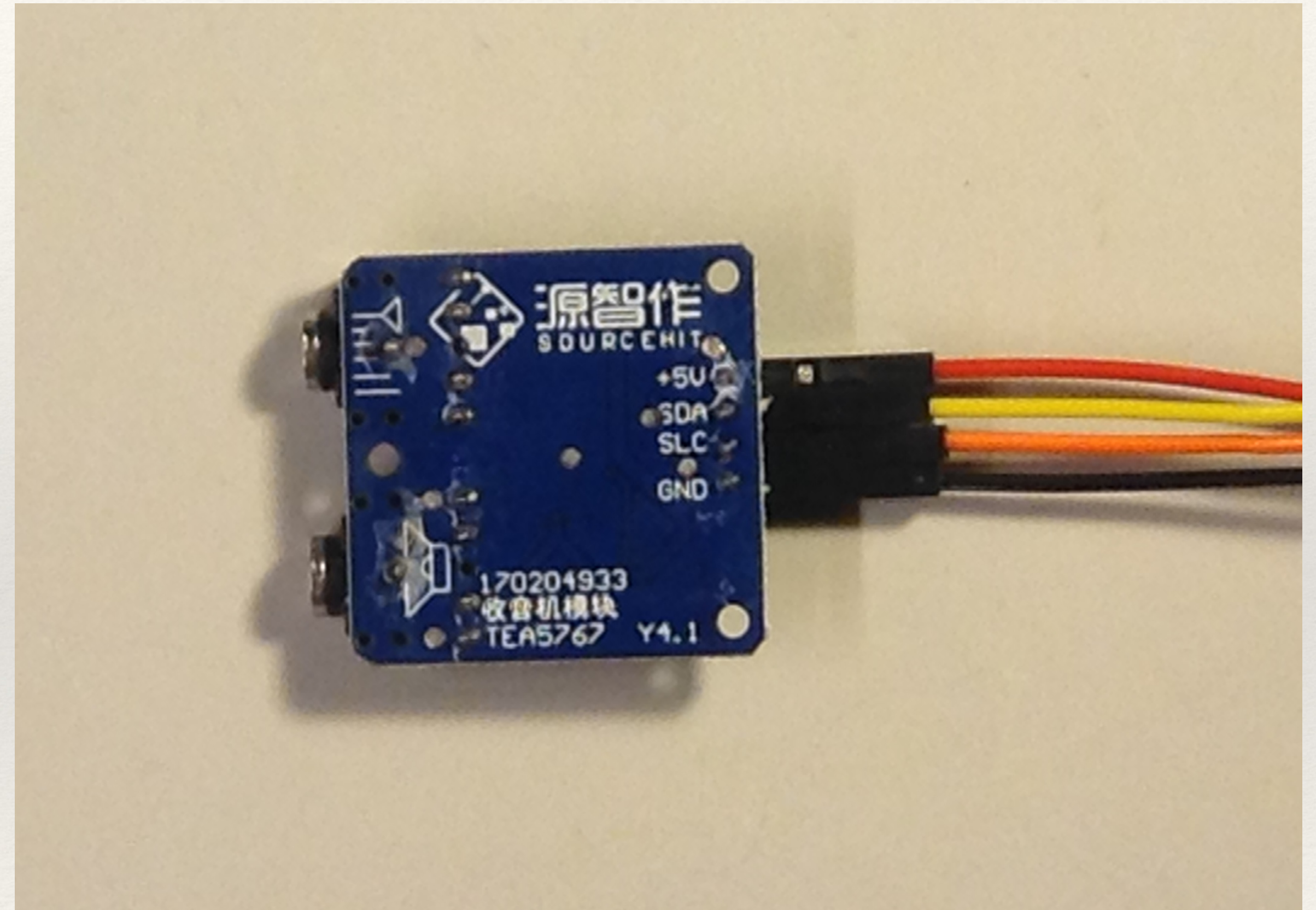

Hardware

- ❖ This Build is very easy!
- ❖ Using the Plug & Socket hook up wires
- ❖ Arduino GND to Radio GND Black Wire
- ❖ Arduino 3.3v to Radio +5v Red Wire
- ❖ Arduino SLC to Radio SLC Orange Wire
- ❖ Arduino SDA to Radio SDA Yellow Wire



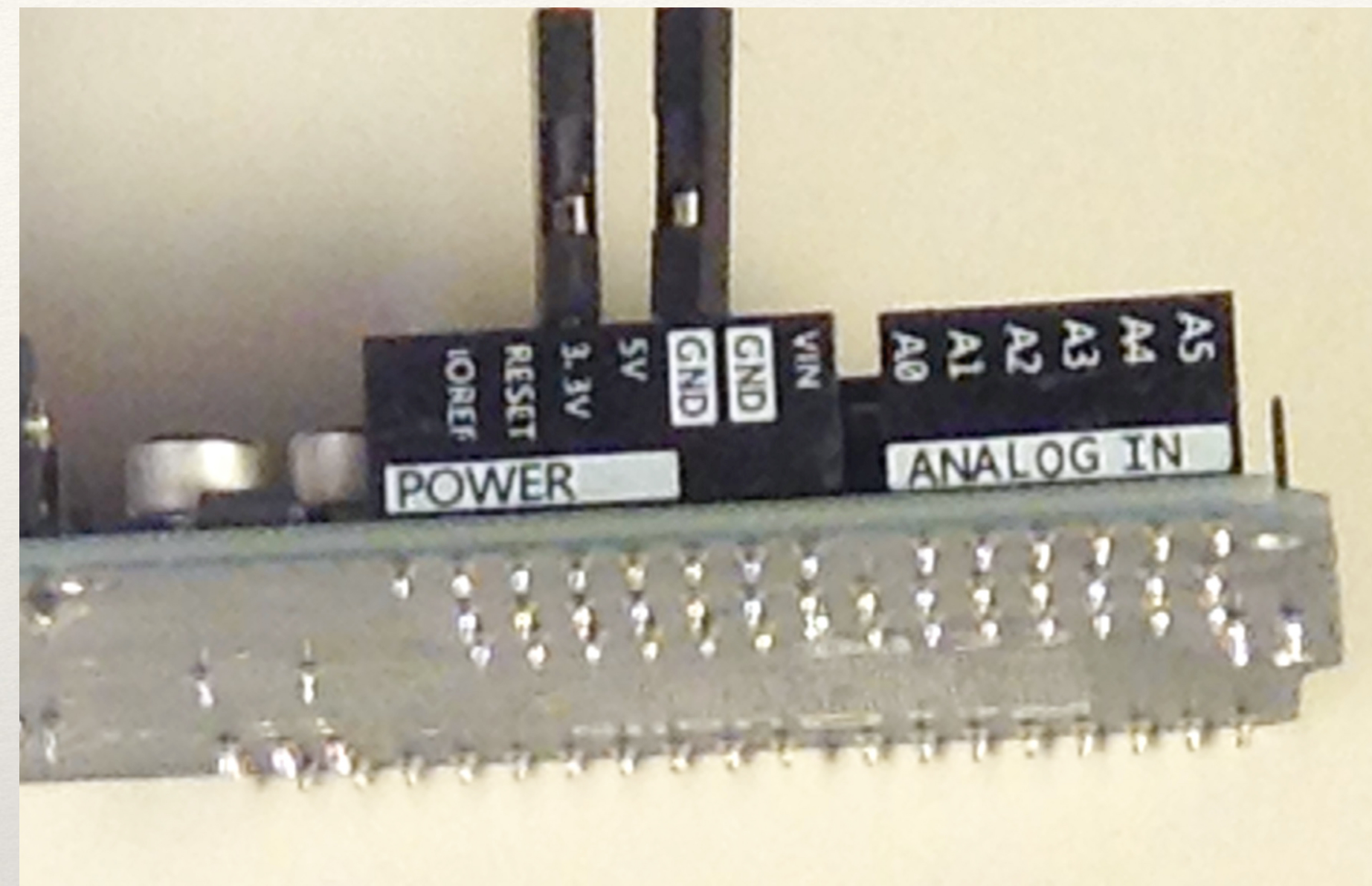
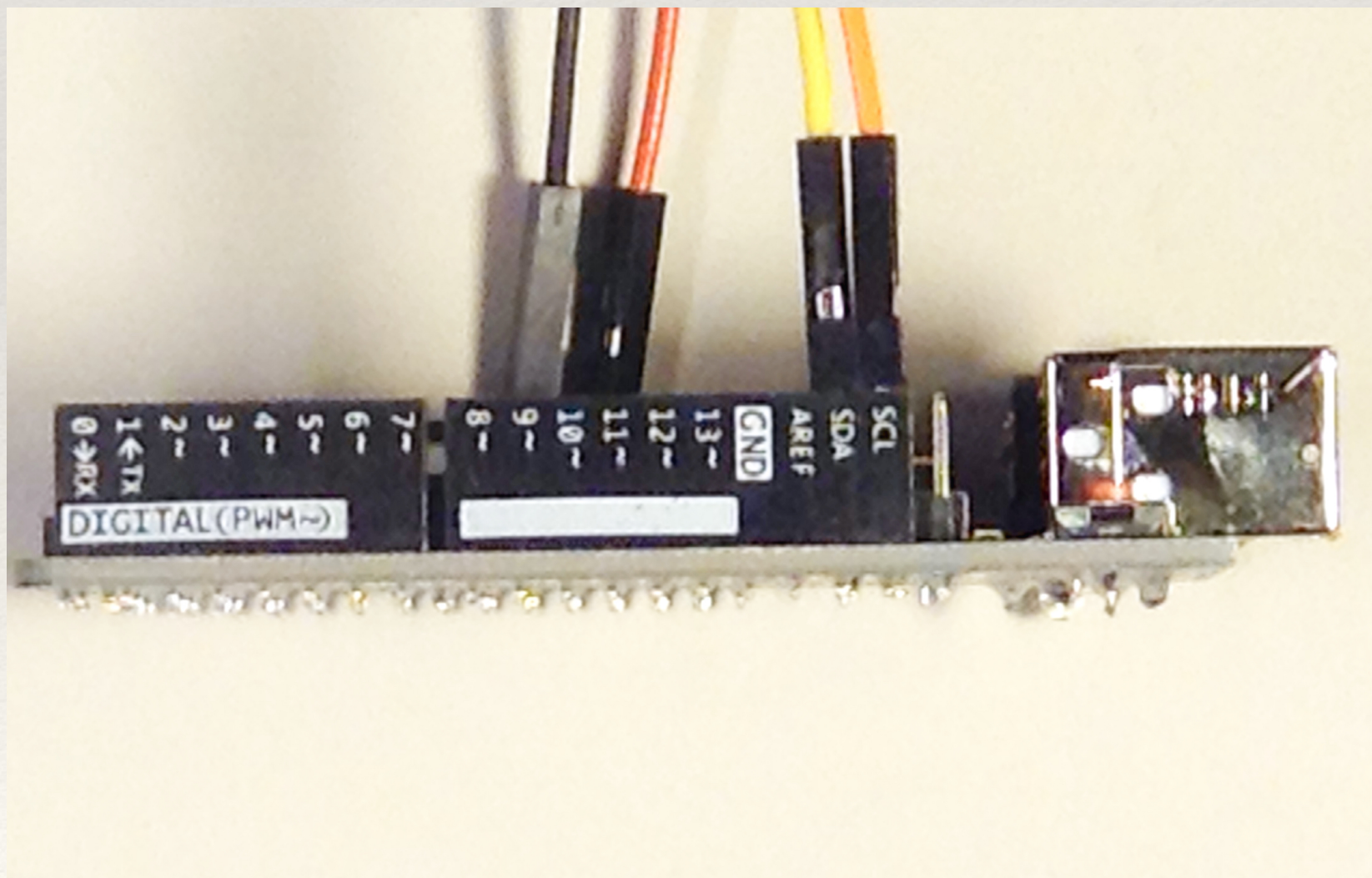
Hardware

- ❖ Your radio should look like this



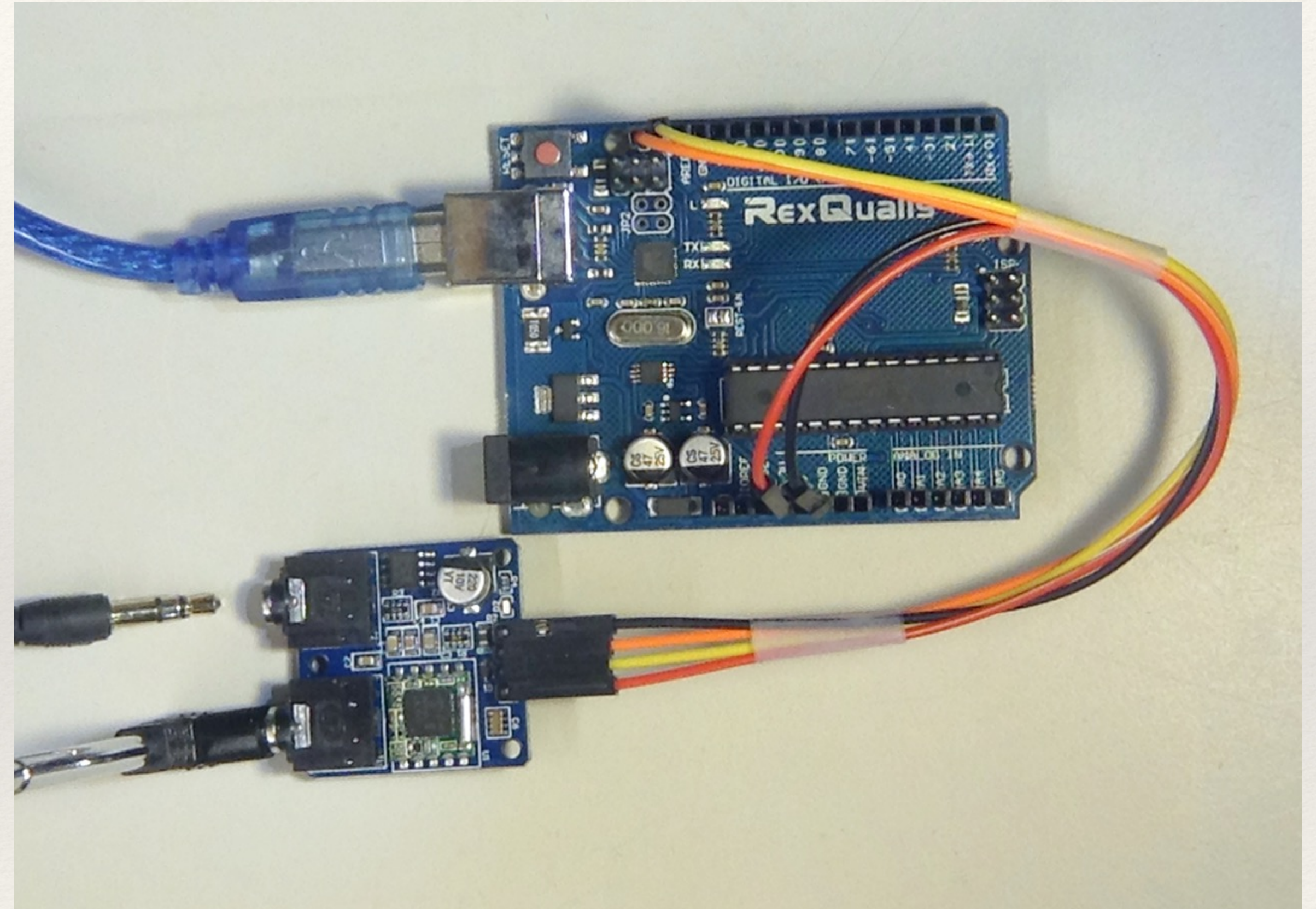
Hardware

- ❖ Your Arduino Uno should look like this



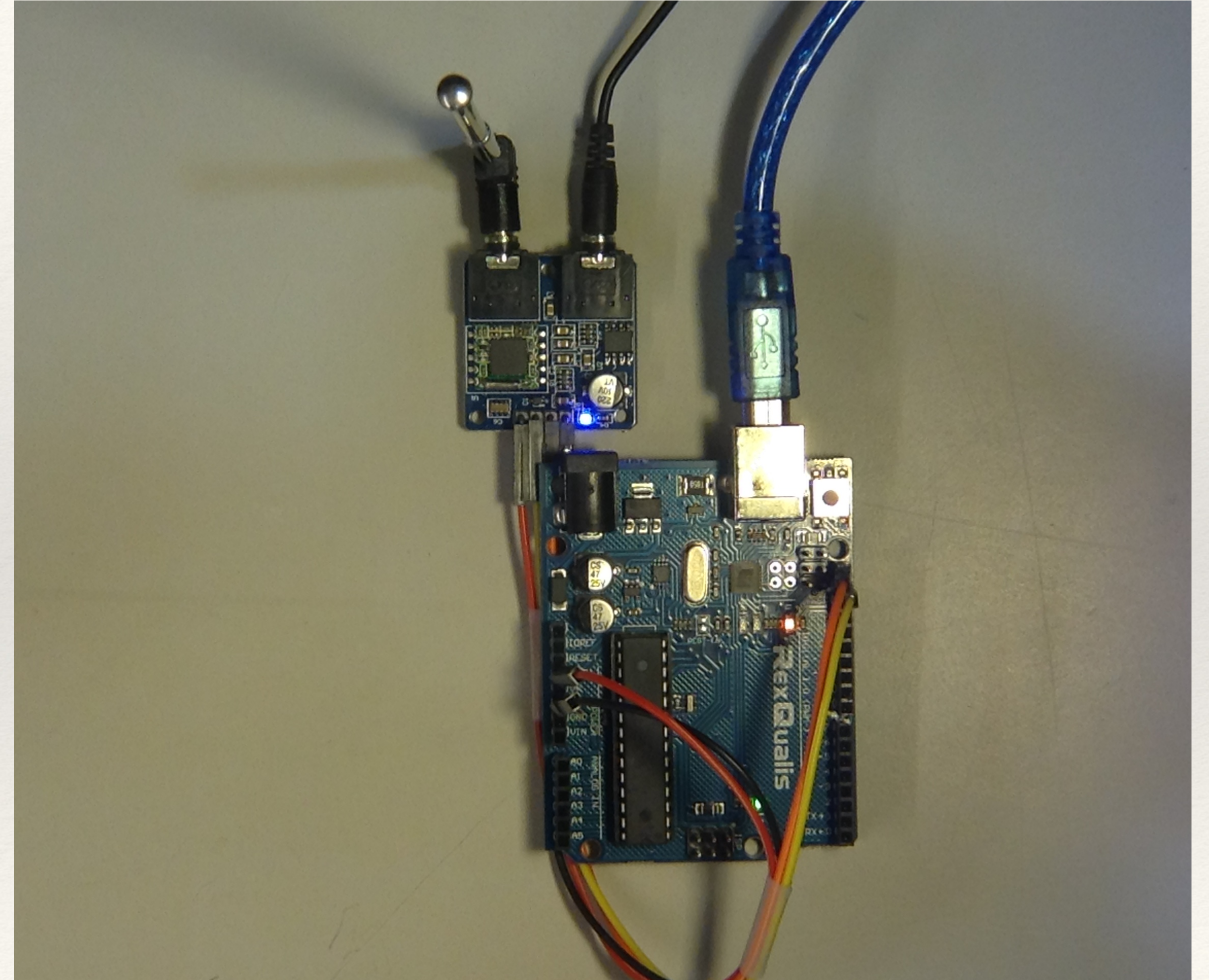
Hardware

- ❖ Next plug in the Antenna to the Radio
- ❖ Plug in Speakers or Head Phones to the Radio
- ❖ Plug in the USB to the Arduino UNO



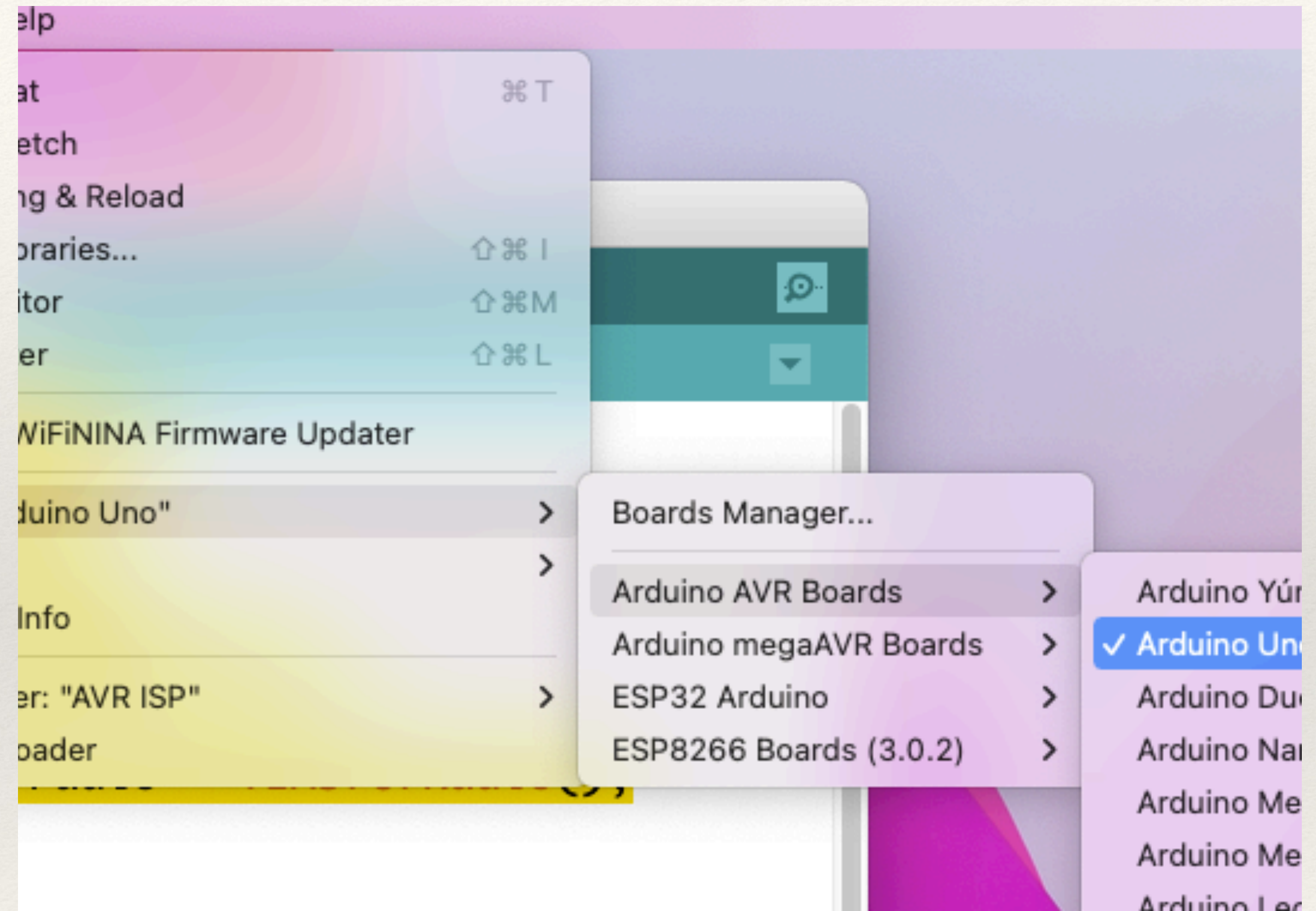
Hardware

- ❖ Double Check the wires
- ❖ Plug in the other end of the USB cable to your computer
- ❖ The Arduino Uno and Radio should have lights on



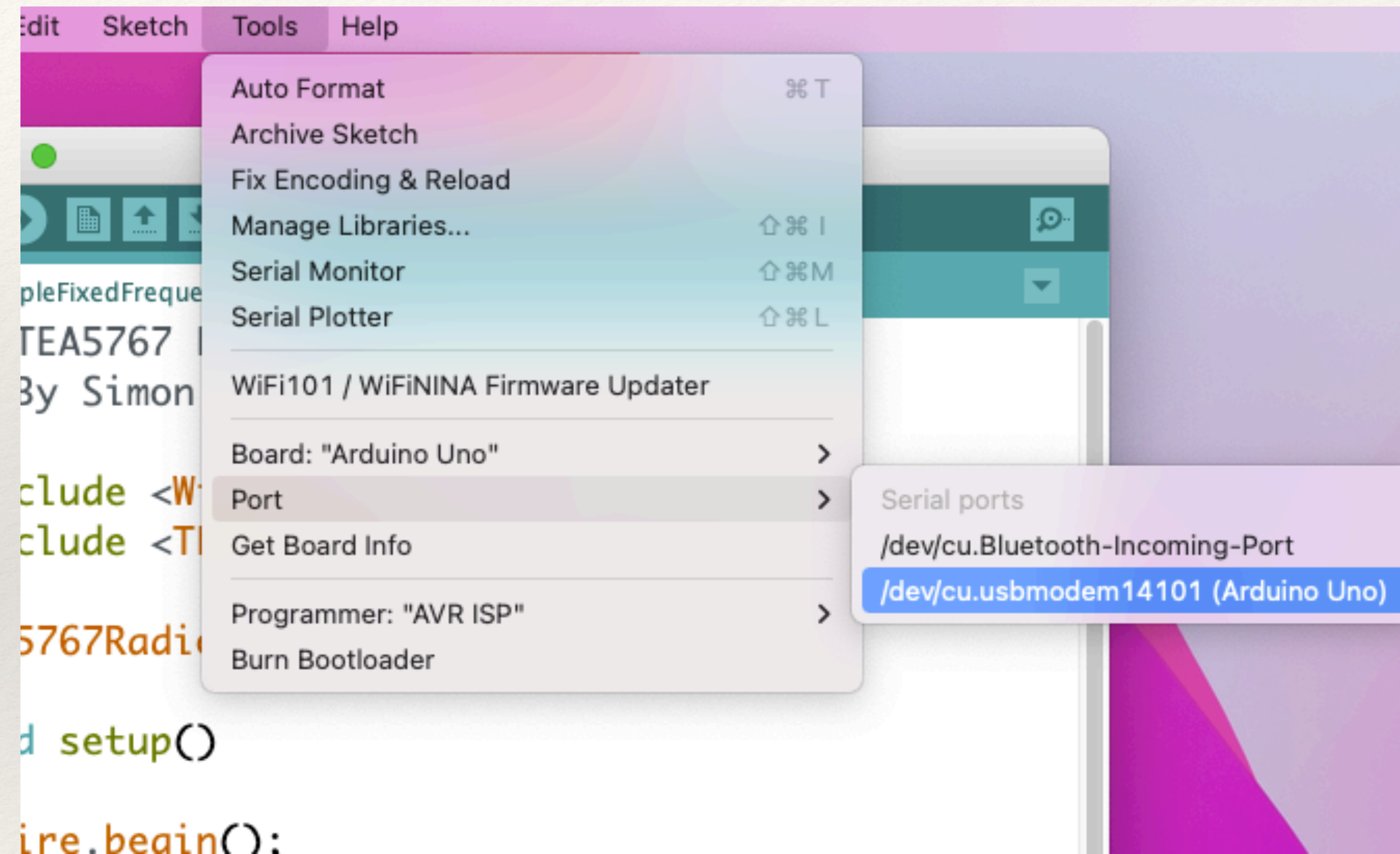
Programming the Arduino Uno

- ❖ We need to tell the Arduino IDE what we are programming
- ❖ From the Tools Menu Select Boards:Arduino AVR:Arduino UNO
- ❖



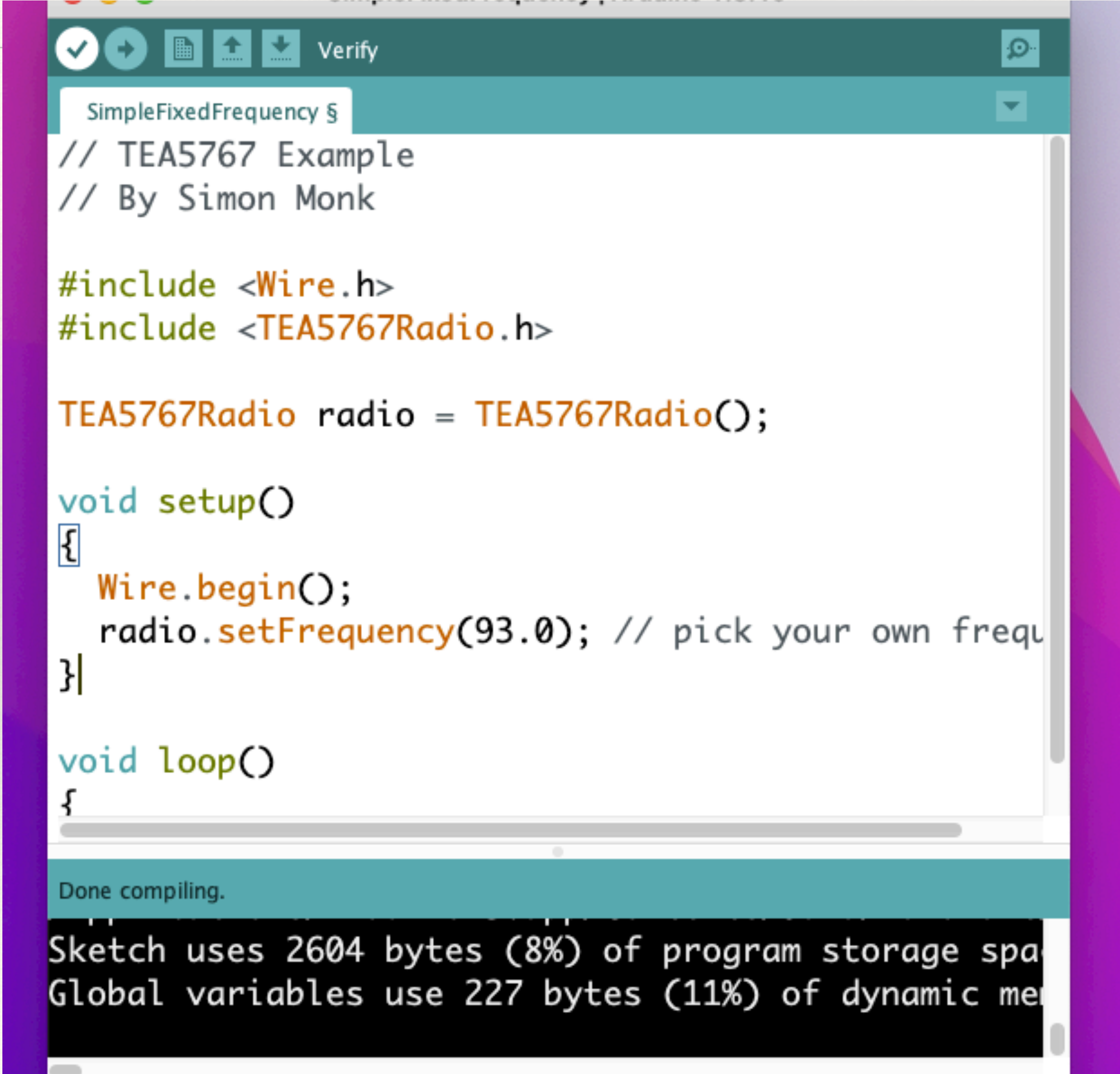
Programming the Arduino Uno

- ❖ Last step before we program , Tell the Arduino IDE
- ❖ Where the UNO is .. From Tools Select Port
- ❖ The Select the Port the Arduino Uno is on.
- ❖ This will look different on a PC it will say COMX (Arduino Uno)
- ❖ We are ready to load a program !



Programming the Arduino Uno

- ❖ From the Arduino IDE Screen Press the Check Mark
- ❖ This will compile the code for the Arduino
- ❖ If successful your screen will look like this.
- ❖



The screenshot shows the Arduino IDE interface. At the top, there is a toolbar with icons for checking, uploading, and verifying code. Below the toolbar, the file name 'SimpleFixedFrequency \$' is displayed. The main text area contains the following C++ code:

```
// TEA5767 Example
// By Simon Monk

#include <Wire.h>
#include <TEA5767Radio.h>

TEA5767Radio radio = TEA5767Radio();

void setup()
{
  Wire.begin();
  radio.setFrequency(93.0); // pick your own frequency
}

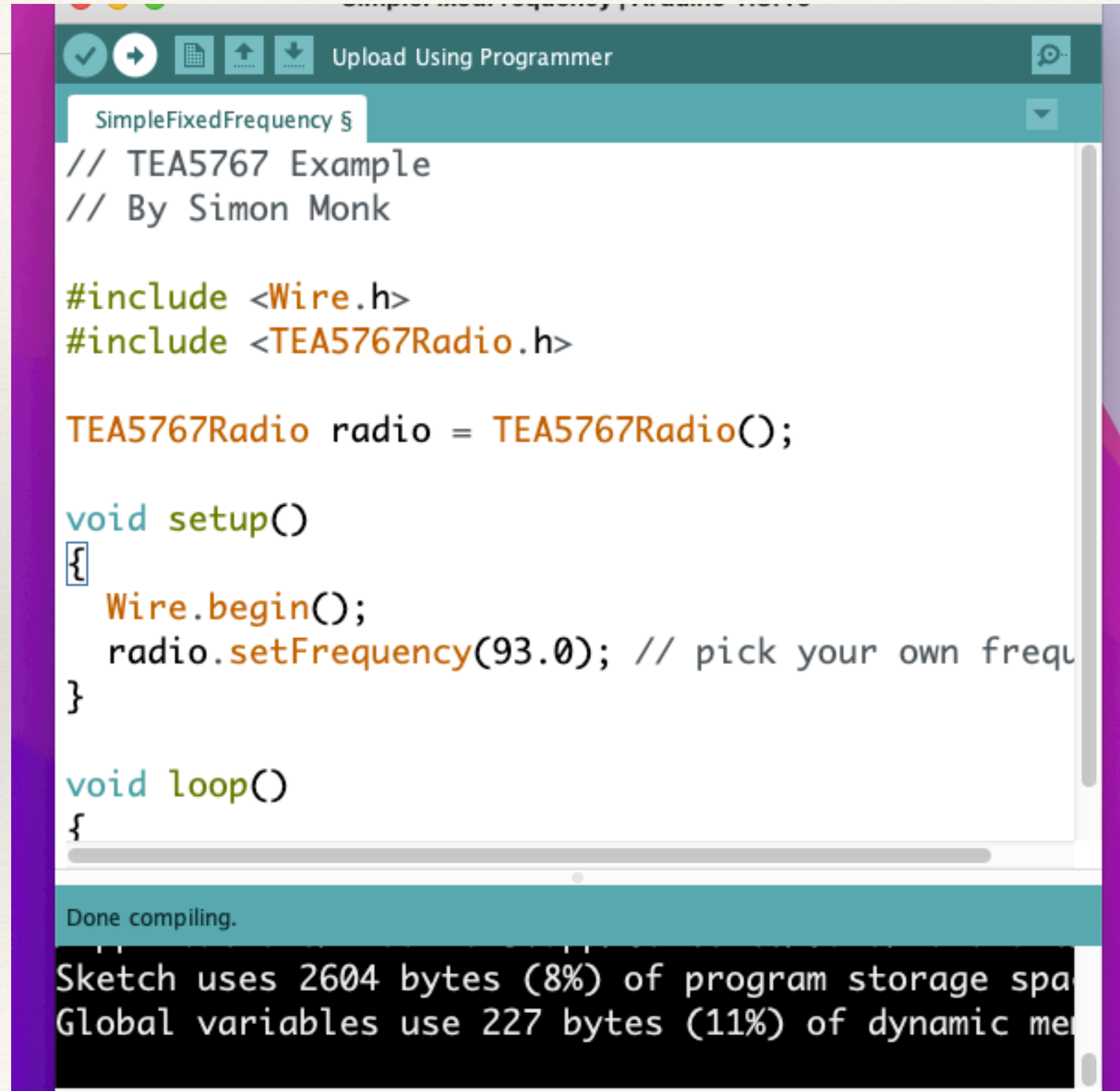
void loop()
{
}
```

Below the code editor, a status bar indicates 'Done compiling.' At the bottom, a black console window displays the following information:

```
Sketch uses 2604 bytes (8%) of program storage space.
Global variables use 227 bytes (11%) of dynamic memory.
```


Programming the Arduino Uno

- ❖ From the Arduino IDE Screen Press the Right Arrow
- ❖ This will send the Program to the Arduino
- ❖ When done Your Screen should look like this.



```
SimpleFixedFrequency §
// TEA5767 Example
// By Simon Monk

#include <Wire.h>
#include <TEA5767Radio.h>

TEA5767Radio radio = TEA5767Radio();

void setup()
{
  Wire.begin();
  radio.setFrequency(93.0); // pick your own frequency
}

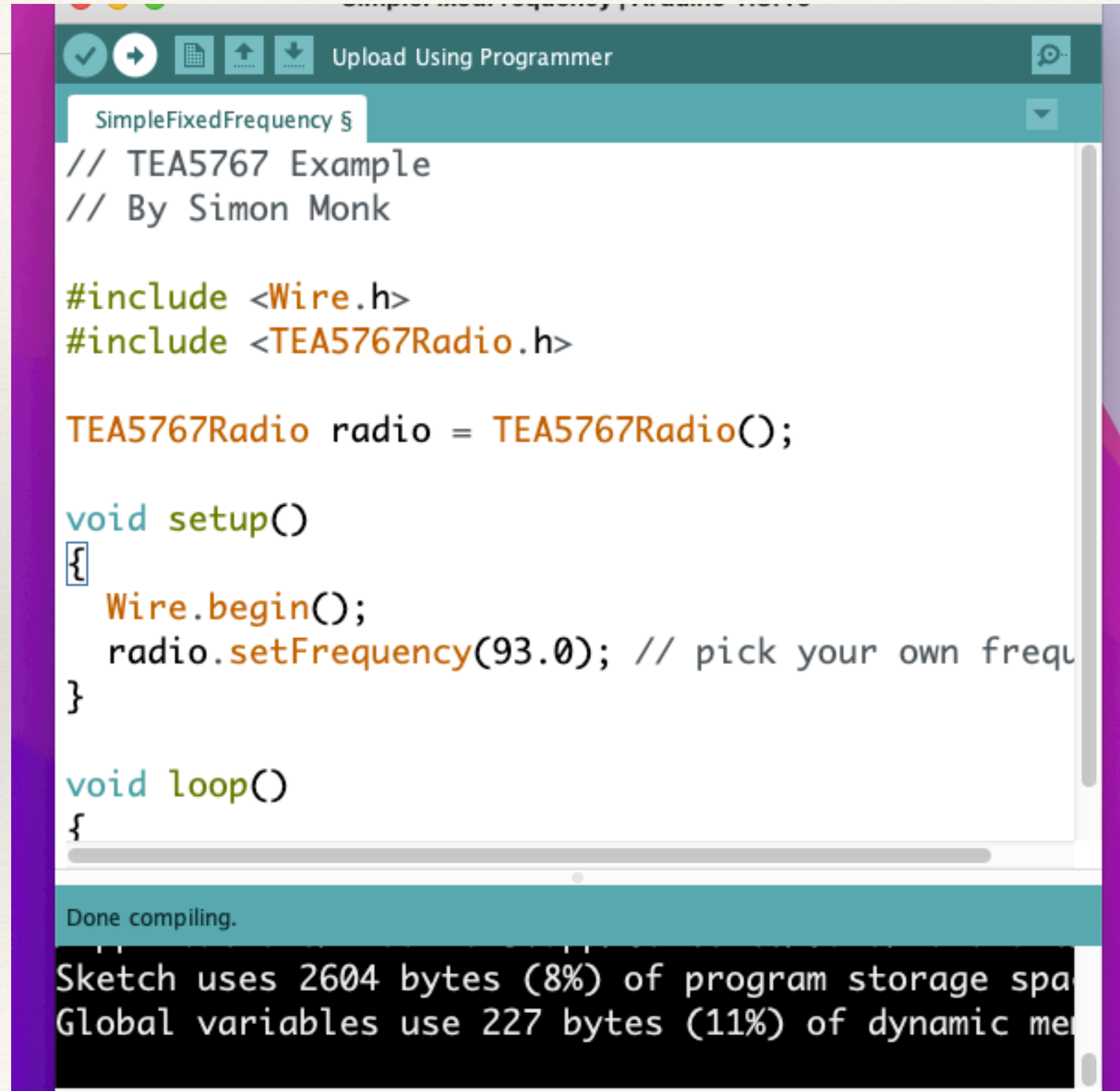
void loop()
{
}
```

Done compiling.

Sketch uses 2604 bytes (8%) of program storage space
Global variables use 227 bytes (11%) of dynamic memory

Programming the Arduino Uno

- ❖ From the Arduino IDE Screen Press the Right Arrow
- ❖ This will send the Program to the Arduino
- ❖ When done Your Screen should look like this.



```
SimpleFixedFrequency §
// TEA5767 Example
// By Simon Monk

#include <Wire.h>
#include <TEA5767Radio.h>

TEA5767Radio radio = TEA5767Radio();

void setup()
{
  Wire.begin();
  radio.setFrequency(93.0); // pick your own frequency
}

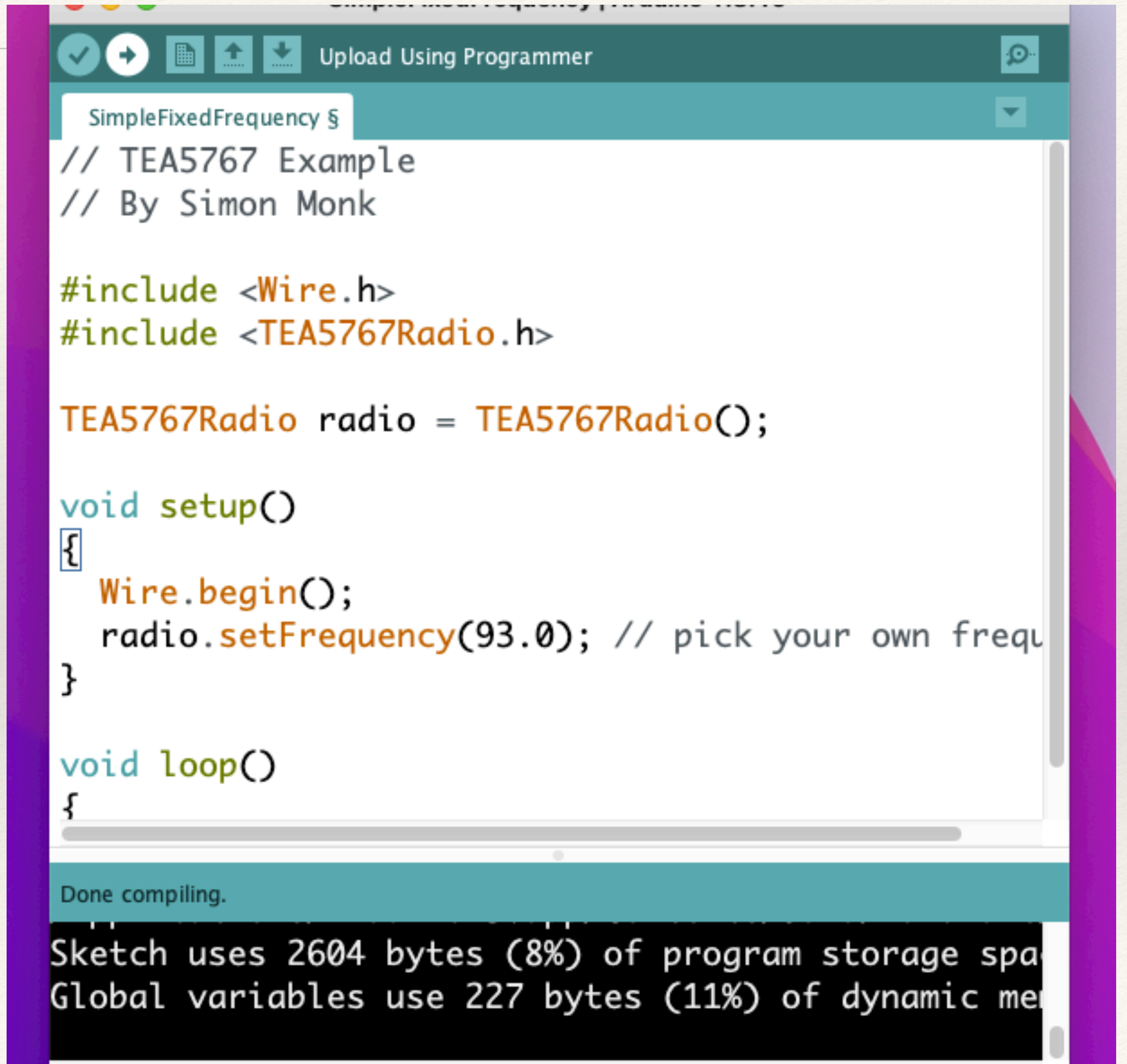
void loop()
{
}
```

Done compiling.

Sketch uses 2604 bytes (8%) of program storage space.
Global variables use 227 bytes (11%) of dynamic memory.

Programming the Arduino Uno

- ❖ This is the end of Part 1 installment 1
- ❖ Later in the week I'll post code to make the radio scan
- ❖ It will be a simple copy and paste
- ❖ To tune your radio in this example
- ❖ change this line `radio.setFrequency(93.0); //`
- ❖ For Example to tune to 103.5 FM you make the change to
- ❖ `radio.setFrequency(103.5); //`
- ❖ See you in a couple days !!!!

A screenshot of the Arduino IDE interface. The top toolbar shows icons for checking, uploading, and saving, with the text "Upload Using Programmer". Below the toolbar, a dropdown menu shows "SimpleFixedFrequency \$". The main text area contains C++ code for a TEA5767 radio example. The code includes comments, header files, a radio object declaration, and setup/loop functions. The status bar at the bottom indicates "Done compiling." and shows memory usage: "Sketch uses 2604 bytes (8%) of program storage space" and "Global variables use 227 bytes (11%) of dynamic memory".

```
SimpleFixedFrequency $
// TEA5767 Example
// By Simon Monk

#include <Wire.h>
#include <TEA5767Radio.h>

TEA5767Radio radio = TEA5767Radio();

void setup()
{
  Wire.begin();
  radio.setFrequency(93.0); // pick your own frequency
}

void loop()
{
}
```

Done compiling.

Sketch uses 2604 bytes (8%) of program storage space
Global variables use 227 bytes (11%) of dynamic memory

Programming the Arduino Uno

```
// TEA5767 Example
// By Simon Monk
// Scanning for open Ghost Box

#include <Wire.h>           // I2C SDA communications for radio module
#include <TEA5767Radio.h>   // Radio library tells the Arduino how to interface the radio module
TEA5767Radio radio = TEA5767Radio(); // Define radio model

void setup()               // setup files
{
  Wire.begin();            // start I2C communications
}

void loop()
{
  for ( float r = 88.00; r <= 108.00;r = r+.1) // loop and scan from 88 to 108 stepping .1 MHz
  {
    radio.setFrequency(r); // Set radio to new frequency
    delay(200);            // wait here for .2 seconds
  }
}
```

Copy and paste the above text into your Arduino IDE editor this will allow the GhostBox to scan from 88.00 to 108.00 FM. The R+.1 sets the scan distance .1 = .1 MHz. The time between scans is is controlled by the delay (200); IE: delay 200 mills second or .2 seconds