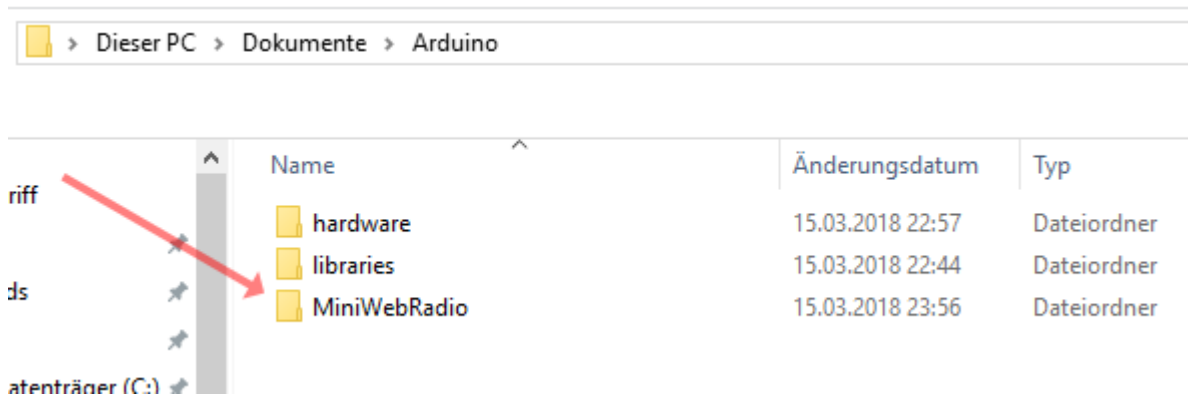


# Notes on programming with the Arduino IDE

---

**The Arduino IDE must be installed and the libraries for the ESP32 be included.**

Create a new sketch and save it as MiniWebRadio. The IDE creates a new folder named MiniWebRadio.



The easiest way to do this is to add all the libraries you need in this folder. The required files can be found in my repositories.

[https://github.com/schreibfaul1/ESP32-vs1053\\_ext](https://github.com/schreibfaul1/ESP32-vs1053_ext)














<https://github.com/schreibfaul1/ESP32-IR-Remote-Control> Optional, for a IR Remote Control)

In addition, the driver for an SPI display with Touchpad is required. For the Waveshare 2.8 inch display, the:



















<https://github.com/schreibfaul1/ESP32-TFT-Library-ILI9431-HX8347D>

For other displays an adjustment is necessary. The TFT libraries from Adafruit are well suited.

If everything is included, the contents of the folder will look like this:

Name	Änderungsdatum	Typ	Größe
 IR.cpp	31.10.2017 07:39	CPP-Datei	6 KB
 IR.h	31.10.2017 07:39	H-Datei	1 KB
 fonts.h	14.03.2018 09:30	H-Datei	1.424 KB
 tft.cpp	14.03.2018 09:30	CPP-Datei	38 KB
 tft.h	14.03.2018 09:30	H-Datei	10 KB
 vs1053_ext.cpp	15.03.2018 10:50	CPP-Datei	44 KB
 vs1053_ext.h	15.03.2018 10:50	H-Datei	9 KB
 html.cpp	15.03.2018 13:11	CPP-Datei	10 KB
 html.h	15.03.2018 13:11	H-Datei	2 KB
 rtime.cpp	15.03.2018 13:11	CPP-Datei	3 KB
 rtime.h	15.03.2018 13:11	H-Datei	1 KB
 web.h	15.03.2018 13:11	H-Datei	25 KB
 MiniWebRadio.ino	15.03.2018 23:17	INO-Datei	46 KB

The contents of the archive „Content\_on\_SD\_Card. zip " [https://github.com/schreibfaul1/ESP32-MiniWebRadio/blob/master/Content\\_on\\_SD\\_Card.zip](https://github.com/schreibfaul1/ESP32-MiniWebRadio/blob/master/Content_on_SD_Card.zip) will be unzipped to the SD card.

 voice_time	
 ring	
 pictures	
 mp3files	
 logo	
 digits	
 day	
 btn	
 Remote_Control.gif	36.122
 Presets.csv	16.876
 networks.csv	187
 MP3_Board.gif	125.352
 jessica.bmp	230.454
 favicon.ico	1.536
 ESP32_Radio_gr.bmp	460.854
 ESP32_Radio.bmp	230.454
 Dev_Board.gif	136.185
 Brightness.bmp	112.374

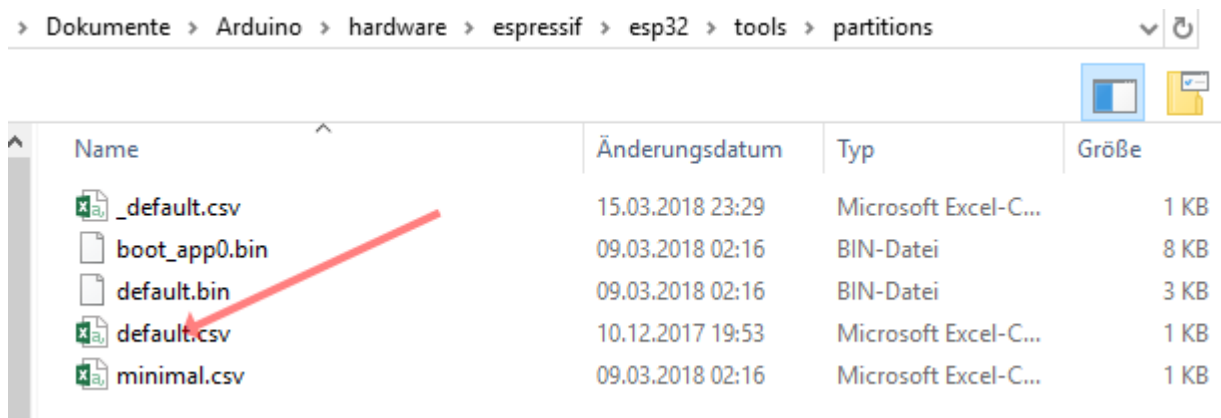
voice_time	Language files for the time (can be played at any hour)
ring	MP3 file for the alarm tone
pictures	Bitmaps to test the display (not strictly required)
mp3files	Music files etc. for the MP3 player
logo	Sender logos as bitmap (96x96 pixels in size)
digits	Alarm clock and time bitmaps
day	Bitmaps for the day (alarm on/off)
btn	Bitmaps for the buttons

preset.csv	The channel list can be edited, the first 256 entries are displayed in the internal nvs stored
networks.csv	If more than one WiFi network exists, the access data can be entered here
favicon.ico	is displayed by the browser on the Web portal. The default URL is: <a href="http://esp32radio/index.html">http://esp32radio/index.html</a>
ESP32_Radio.bmp	The Home screen
Brightness.bmp	Display Brightness menu graphic

Because more NVS memory is required for the channel list, the partition table must be changed.

#	Name	Type	SubType	Offset	Size	Flags
	phy_init	data	phy	0x9000	0x7000	
	factory	app	factory	0x10000	0x300000	
	nvs	data	nvs	0x310000	0x32000	
	spiffs	data	spiffs	0x342000	0xB0000	
	eeeprom	data	0x99	0x3F2000	0xD000	

This can be done with a text editor.



Or alternatively, the default. csv will overwrite the file from my repository.

In boards.txt xxx.esp32.upload.maximum\_size=1310720 xxx stands for Your ESP32board.  
You have to increase that value because MiniWebRadio is bigger than 1.3MBytes.  
Set the value to **3145728** (3Mbytes)

After that, the sketch can be compiled and uploaded.

The screenshot shows the Arduino IDE interface for the 'MiniWebRadio' project. The top menu bar includes 'Datei', 'Bearbeiten', 'Sketch', 'Werkzeuge', and 'Hilfe'. The toolbar contains icons for opening, saving, and uploading files. The file explorer shows the project structure with files like 'IR.cpp', 'IR.h', 'fonts.h', 'html.cpp', 'html.h', 'rtime.cpp', 'rtime.h', 'tft.cpp', 'tft.h', 'vs1053\_ext.cpp', 'vs1053\_ext.h', and 'web.h'. The main editor displays the 'IR.cpp' file with the following code:

```
#include "rtime.h"
#include "web.h"

// Digital I/O used
#define VS1053_CS      2
#define VS1053_DCS     4
#define VS1053_DREQ    36
#define TFT_CS         22
#define TFT_DC         21
#define TFT_BL         17
#define TP_IRQ         39
#define TP_CS          16
#define SD_CS           5
#define IR_PIN         34

//global variables

char sbuf[256], myIP[100];
String _station="", _title="", _info="", _myIP="", _stationname="", _alarmtime="", _time_s="", _hour="", _bitrate="";
String _mp3Name[10], _pressBtn[5], _releaseBtn[5];
```

The bottom status bar shows the upload progress and completion message:

```
Hochladen abgeschlossen.
writing at 0x0005c000... (76 %)
Writing at 0x00060000... (80 %)
Writing at 0x00064000... (84 %)
Writing at 0x00068000... (88 %)
Writing at 0x0006c000... (92 %)
Writing at 0x00070000... (96 %)
Writing at 0x00074000... (100 %)
Wrote 1003696 bytes (424212 compressed) at 0x00010000 in 10.4 seconds (effective 770.0 kbit/s)...
Hash of data verified.
Compressed 3072 bytes to 119...

Writing at 0x00008000... (100 %)
Wrote 3072 bytes (119 compressed) at 0x00008000 in 0.0 seconds (effective 4915.1 kbit/s)...
Hash of data verified.
Leaving...
Hard resetting...
```

The bottom status bar also displays the hardware information: 'ESP32 Dev Module, Q10, 80MHz, 4MB (32Mb), 921600, None auf COM3'.

Sincerely,

Wolle