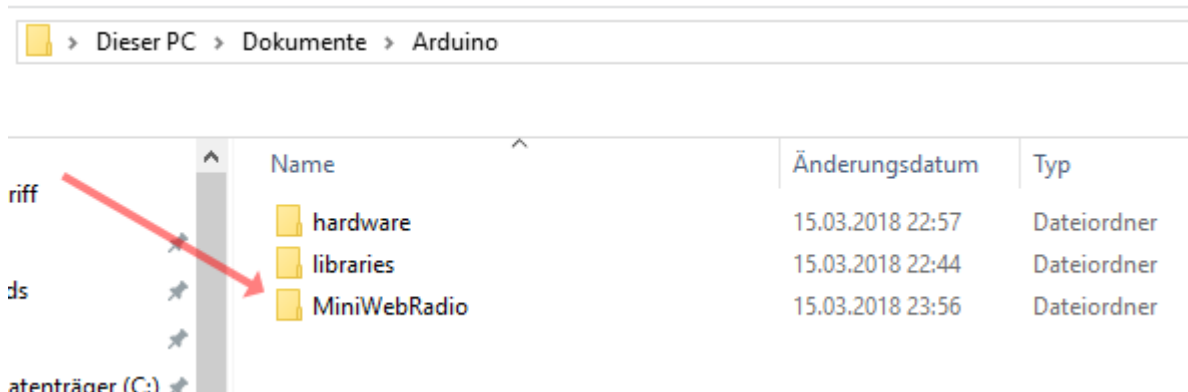


Hinweise zur Programmierung mit der Arduino IDE

Die Arduino IDE muss installiert und die Bibliotheken für den ESP32 eingebunden sein.

Legen Sie einen neuen Sketch an und speichern Sie diesen als MiniWebRadio. Die IDE legt einen neuen Ordner mit dem Namen MiniWebRadio an.



Am einfachsten ist es, in diesem Ordner alle benötigten Bibliotheken hinzuzufügen. Die erforderlichen Dateien finden Sie in meinen Repositories.

https://github.com/schreibfaul1/ESP32-vs1053_ext

<https://github.com/schreibfaul1/ESP32-IR-Remote-Control> (optional, für eine IR Fernbedienung)

Zusätzlich wird der Treiber für ein SPI-Display mit Touchpad benötigt. Für das Waveshare 2.8inch Display ist das

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

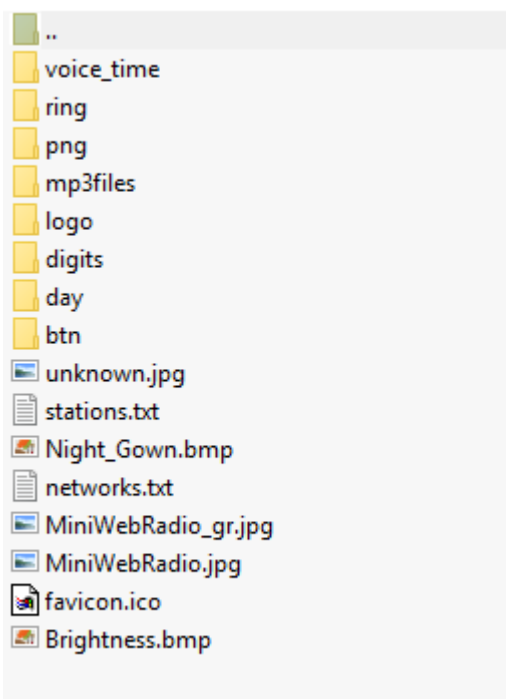
Für andere Displays ist eine Anpassung notwendig.

Ist alles eingebunden, sieht der Inhalt des Ordners so aus:

Name	Änderungsdatum	Typ	Größe
fonts	19.11.2018 18:38	Dateiordner	
html.cpp	20.10.2018 16:47	CPP-Datei	18 KB
html.h	15.10.2018 22:58	H-Datei	3 KB
IR.cpp	09.09.2018 20:56	CPP-Datei	7 KB
IR.h	22.08.2018 23:24	H-Datei	1 KB
MiniWebRadio.ino	19.11.2018 18:32	Arduino file	62 KB
rtime.cpp	04.09.2018 14:03	CPP-Datei	14 KB
rtime.h	04.09.2018 14:03	H-Datei	2 KB
tft.cpp	21.10.2018 20:08	CPP-Datei	128 KB
tft.h	22.10.2018 05:36	H-Datei	30 KB
vs1053_ext.cpp	17.11.2018 11:56	CPP-Datei	57 KB
vs1053_ext.h	07.10.2018 12:12	H-Datei	10 KB
web.h	20.10.2018 10:05	H-Datei	62 KB

In fonts befinden sich verschiedene Schriftarten, MiniWebRadio verwendet nur die Schrift Garamond

Der Inhalt des Archivs „Content_on_SD_Card.zip“ https://github.com/schreibfaul1/ESP32-MiniWebRadio/blob/master/Content_on_SD_Card.zip wird auf die SD Card entpackt.



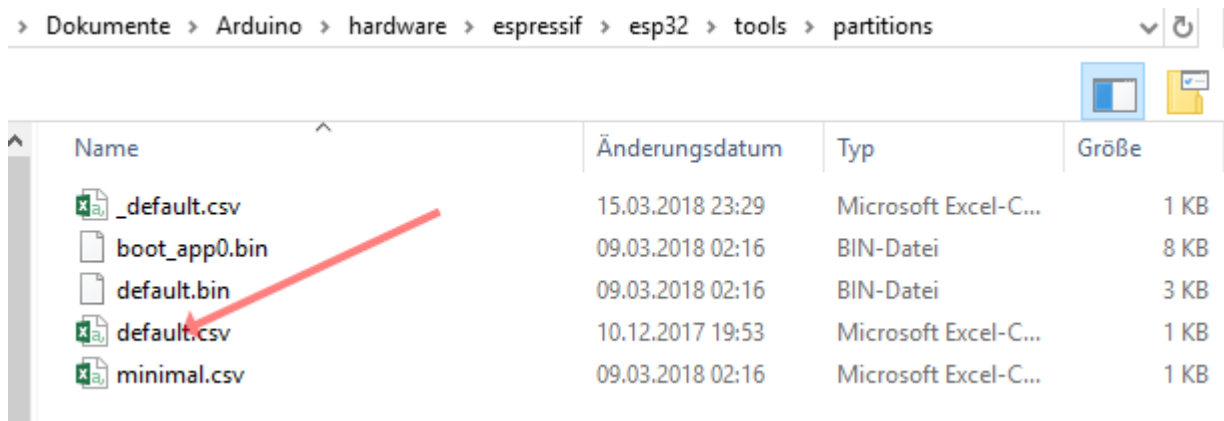
voice_time	Sprachdateien für die Uhrzeit (kann zu jeder vollen Stunde gespielt werden)
ring	mp3 Datei für den Weckton
png	Symbole für die Webseite
mp3files	Musikdateien etc. für den mp3 Player
logo	Senderlogos als Jpeg (96x96 Pixel groß)
digits	Bitmaps für Wecker und Uhrzeit
day	Bitmaps für den Tag (Wecker ein/aus)
btn	Bitmaps für die Buttons
stations.txt	die Senderliste, kann bis auf 999 Stationen erweitert werden, ein * am Zeilenanfang ignoriert den Eintrag, STsubstitute wird angezeigt wenn

networks.txt	die Station keine Streamtitel (Titel, Interpret etc) übermittelt optional, bei mehr als einem verfügbaren WLAN können hier die Zugangsdaten (Netzwerkname, Passwort, Bemerkung) eingetragen werden
favicon.ico	wird vom Browser auf dem Webportal angezeigt. Die Standard URL ist: http://esp32radio/index.html
ESP32_Radio.jpg	der Startbildschirm
Brightness.bmp	Grafik des Menüs Display Helligkeit

Weil mehr nvs Speicher für die Senderliste erforderlich ist sollte die Partitionstabelle geändert werden. Die default.csv befindet sich bei einer Standardinstallation in
C:\Users\...\AppData\Local\Arduino15\packages\esp32\hardware\esp32\1.0.0\tools\partitions

#	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	

Das kann mit einem Texteditor gemacht werden.



Oder alternativ wird die default.csv mit der Datei aus meinem Reposotory überschrieben.

Die Arduino IDE bezieht das Upload-Maximum aus der boards.txt.
Dort steht für das eingestellte Bord z.B. [BoardName].upload.maximum_size=1310720
Miniwebradio ist größer als dieser Wert (ca. 1.5Mbytes) und wird daher nicht hochgeladen.
Daher muss dieser Wert vergrößert werden z.B. [BoardName].upload.maximum_size=**3145728**

Danach kann der Sketch kompiliert und hochgeladen werden.

The screenshot shows the Arduino IDE interface. 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 'MiniWebRadio' with files: 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 code for 'web.h', which includes 'rtime.h' and 'web.h'. It defines various pins for the VS1053 module and the TFT display. Global variables are declared, including a buffer 'sbuf' and IP address 'myIP'. The code also defines strings for station information and button labels. The bottom status bar indicates the board is 'ESP32 Dev Module, QIO, 80MHz, 4MB (32Mb), 921600, None auf COM3'. The bottom console shows the upload progress, indicating that the upload is complete and the device is resetting.

```
#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];

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...
```

Mit freundlichen Grüßen,
Wolle