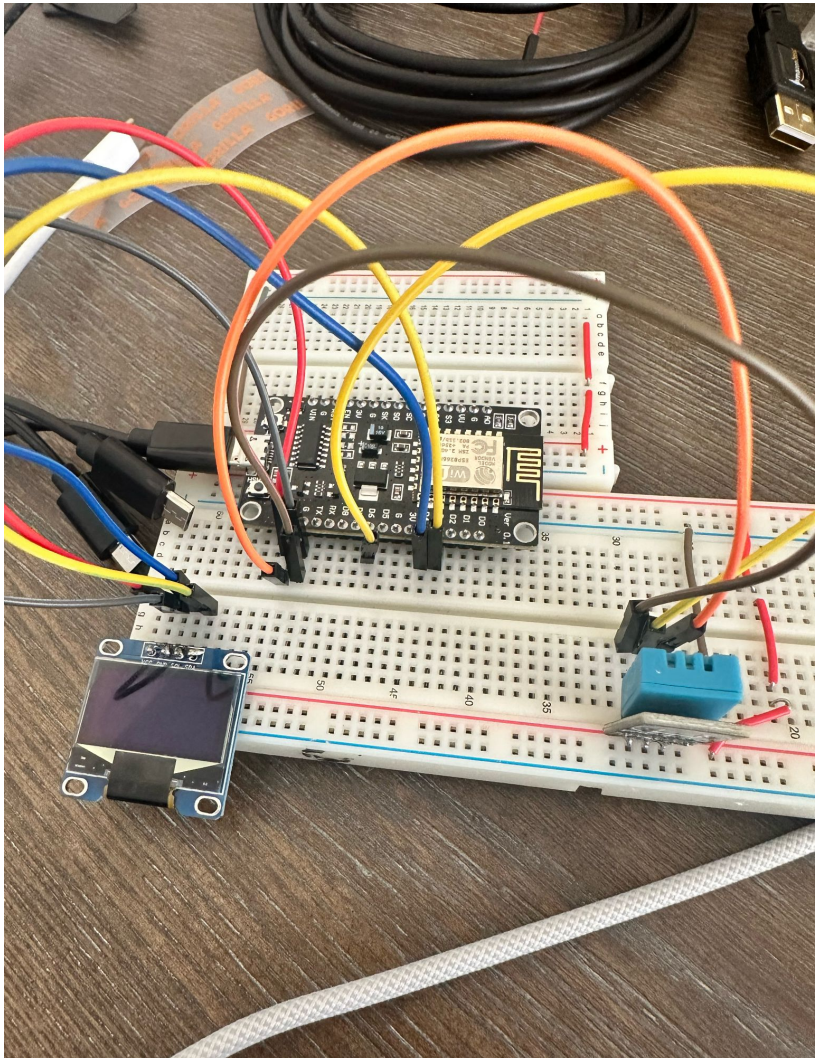
ESP8266 Wifi Temperature and Humidity

Julian Torres

Requirements

Get temperature and humidity readings using the ESP8266 using the weather station implementation

Circuit Design



Code

```
#include <Arduino.h>

#if defined(ESP8266)
#include <ESP8266WiFi.h>
#include <coredecls.h>                  // settimeofday_cb()
#else
#include <WiFi.h>
#endif
#include <ESPHTTPClient.h>
#include <JsonListener.h>

// time
#include <time.h>                       // time() ctime()
#include <sys/time.h>                   // struct timeval

#include "SSD1306Wire.h"
#include "OLEDDisplayUi.h"
#include "Wire.h"
#include "OpenWeatherMapCurrent.h"
#include "OpenWeatherMapForecast.h"
#include "WeatherStationFonts.h"
#include "WeatherStationImages.h"

// humidity sensor
#include "DHTesp.h"

/***************************
 * Begin Settings
 **************************/

// WIFI
const char* WIFI_SSID = "Verizon_GSD6BH";
const char* WIFI_PWD = "say-rower6-fife";

#define TZ              2       // (utc+) TZ in hours
#define DST_MN          60      // use 60mn for summer time in some countries

// Setup
const int UPDATE_INTERVAL_SECS = 20 * 60; // Update every 20 minutes

```

Code

```
// Display Settings
const int I2C_DISPLAY_ADDRESS = 0x3c;
#if defined(ESP8266)
const int SDA_PIN = D3;
const int SDC_PIN = D4;
#else
const int SDA_PIN = 5; //D3;
const int SDC_PIN = 4; //D4;
#endif

// OpenWeatherMap Settings
String OPEN_WEATHER_MAP_APP_ID = "80f686f1bf564a0dcf570d707a0b4920";
float OPEN_WEATHER_MAP_LOCATION_LAT = 39.2904;
float OPEN_WEATHER_MAP_LOCATION_LON = 76.6122;

String OPEN_WEATHER_MAP_LANGUAGE = "en";
const uint8_t MAX_FORECASTS = 4;

const boolean IS_METRIC = true;

const String WDAY_NAMES[] = {"SUN", "MON", "TUE", "WED", "THU", "FRI", "SAT"};
const String MONTH_NAMES[] = {"JAN", "FEB", "MAR", "APR", "MAY", "JUN", "JUL", "AUG", "SEP", "OCT", "NOV", "DEC"};

/***************************
 * End Settings
 **************************/

SSD1306Wire     display(I2C_DISPLAY_ADDRESS, SDA_PIN, SDC_PIN);
OLEDDisplayUi   ui( &display );

OpenWeatherMapCurrentData currentWeather;
OpenWeatherMapCurrent currentWeatherClient;

OpenWeatherMapForecastData forecasts[MAX_FORECASTS];
OpenWeatherMapForecast forecastClient;

#define TZ_MN           ((TZ)*60)
#define TZ_SEC          ((TZ)*3600)
#define DST_SEC         ((DST_MN)*60)
time_t now;

bool readyForWeatherUpdate = false;
String lastUpdate = "--";
long timeSinceLastWUpdate = 0;

DHTesp dht; // Global DHT sensor object
```

Code

```
// declaring prototypes
void drawProgress(OLEDDisplay *display, int percentage, String label);
void updateData(OLEDDisplay *display);
void drawDateTime(OLEDDisplay *display, OLEDDisplayUiState* state, int16_t x, int16_t y);
void drawCurrentWeather(OLEDDisplay *display, OLEDDisplayUiState* state, int16_t x, int16_t y);
void drawForecast(OLEDDisplay *display, OLEDDisplayUiState* state, int16_t x, int16_t y);
void drawForecastDetails(OLEDDisplay *display, int x, int y, int dayIndex);
void drawHeaderOverlay(OLEDDisplay *display, OLEDDisplayUiState* state);
void setReadyForWeatherUpdate();

// Add frames
FrameCallback frames[] = { drawDateTime, drawCurrentWeather, drawForecast };
int numberOfFrames = 3;

OverlayCallback overlays[] = { drawHeaderOverlay };
int numberOfOverlays = 1;

void setup() {
  Serial.begin(115200);
  Serial.println();
  Serial.println();

  // Initialize humidity sensor
  dht.setup(D6, DHTesp::DHT11);

  // initialize display
  display.init();
  display.clear();
  display.display();

  display.setFont(ArialMT_Plain_10);
  display.setTextAlignment(TEXT_ALIGN_CENTER);
  display.setContrast(255);

  WiFi.begin(WIFI_SSID, WIFI_PWD);

  int counter = 0;
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
    display.clear();
    display.drawString(64, 10, "Connecting to WiFi");
    display.drawXbm(46, 30, 8, 8, counter % 3 == 0 ? activeSymbole : inactiveSymbole);
    display.drawXbm(60, 30, 8, 8, counter % 3 == 1 ? activeSymbole : inactiveSymbole);
    display.drawXbm(74, 30, 8, 8, counter % 3 == 2 ? activeSymbole : inactiveSymbole);
    display.display();

    counter++;
  }
```

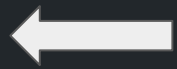
Code

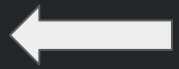
```
163   configTime(TZ_SEC, DST_SEC, "pool.ntp.org");
164
165   ui.setTargetFPS(30);
166   ui.setActiveSymbol(activeSymbole);
167   ui.setInactiveSymbol(inactiveSymbole);
168   ui.setIndicatorPosition(BOTTOM);
169   ui.setIndicatorDirection(LEFT_RIGHT);
170   ui.setFrameAnimation(SLIDE_LEFT);
171   ui.setFrames(frames, numberOfFrames);
172   ui.setOverlays(overlays, numberOfOverlays);
173   ui.init();
174
175   Serial.println("");
176
177   updateData(&display);
178 }
179
180 void loop() {
181   if (millis() - timeSinceLastWUpdate > (1000L*UPDATE_INTERVAL_SECS)) {
182     setReadyForWeatherUpdate();
183     timeSinceLastWUpdate = millis();
184   }
185
186   if (readyForWeatherUpdate && ui.getUiState()->frameState == FIXED) {
187     updateData(&display);
188   }
189
190   int remainingTimeBudget = ui.update();
191
192   if (remainingTimeBudget > 0) {
193     delay(remainingTimeBudget);
194   }
195 }
196
197 void drawProgress(OLEDDisplay *display, int percentage, String label) {
198   display->clear();
199   display->setTextAlignment(TEXT_ALIGN_CENTER);
200   display->setFont(ArialMT_Plain_10);
201   display->drawString(64, 10, label);
202   display->drawProgressBar(2, 28, 124, 10, percentage);
203   display->display();
204 }
205
206 void updateData(OLEDDisplay *display) {
207   drawProgress(display, 10, "Updating time...");
208   drawProgress(display, 30, "Updating weather...");
209   currentWeatherClient.setMetric(IS_METRIC);
210   currentWeatherClient.setLanguage(OPEN_WEATHER_MAP_LANGUAGE);
211   currentWeatherClient.updateCurrent(&currentWeather, OPEN_WEATHER_MAP_APP_ID, OPEN_WEATHER_MAP_LOCATION_LAT, OPEN_WEATHER_MAP_LOCATION_LON);
212   drawProgress(display, 50, "Updating forecasts...");
213   forecastClient.setMetric(IS_METRIC);
214   forecastClient.setLanguage(OPEN_WEATHER_MAP_LANGUAGE);
215   uint8_t allowedHours[] = {12};
```

humidity

Code

```
214   forecastClient.setLanguage(OPEN_WEATHER_MAP_LANGUAGE);
215   uint8_t allowedHours[] = {12};
216   forecastClient.setAllowedHours(allowedHours, sizeof(allowedHours));
217   forecastClient.updateForecasts(forecasts, OPEN_WEATHER_MAP_APP_ID, OPEN_WEATHER_MAP_LOCATION_LAT, OPEN_WEATHER_MAP_LOCATION_LON, MAX_FORECASTS);
218
219   readyForWeatherUpdate = false;
220   drawProgress(display, 100, "Done...");
221   delay(1000);
222 }
223
224 void drawDateTime(OLEDDisplay *display, OLEDDisplayUiState* state, int16_t x, int16_t y) {
225   now = time(nullptr);
226   struct tm* timeInfo;
227   timeInfo = localtime(&now);
228   char buff[16];
229
230   display->setTextAlignment(TEXT_ALIGN_CENTER);
231   display->setFont(ArialMT_Plain_10);
232   String date = WDAY_NAMES[timeInfo->tm_wday];
233
234   sprintf_P(buff, PSTR("%s, %02d/%02d/%04d"), WDAY_NAMES[timeInfo->tm_wday].c_str(), timeInfo->tm_mday, timeInfo->tm_mon+1, timeInfo->tm_year + 1900);
235   display->drawString(64 + x, 5 + y, String(buff));
236   display->setFont(ArialMT_Plain_24);
237
238   sprintf_P(buff, PSTR("%02d:%02d:%02d"), timeInfo->tm_hour, timeInfo->tm_min, timeInfo->tm_sec);
239   display->drawString(64 + x, 15 + y, String(buff));
240   display->setTextAlignment(TEXT_ALIGN_LEFT);
241 }
242
243 void drawCurrentWeather(OLEDDisplay *display, OLEDDisplayUiState* state, int16_t x, int16_t y) {
244   display->setFont(ArialMT_Plain_10);
245   display->setTextAlignment(TEXT_ALIGN_CENTER);
246   display->drawString(64 + x, 38 + y, currentWeather.description);
247
248   display->setFont(ArialMT_Plain_24);
249   display->setTextAlignment(TEXT_ALIGN_LEFT);
250   String temp = String(currentWeather.temp, 1) + (IS_METRIC ? "°C" : "°F");
251   display->drawString(60 + x, 5 + y, temp);
252
253   // Display humidity
254   String humidity = String(dht.getHumidity(), 1) ;//+ String(char(37));
255   display->setFont(ArialMT_Plain_10);
256   display->drawString(64 + x, 28 + y, "Humidity: " + humidity);
257
258   display->setFont(Meteocons_Plain_36);
259   display->setTextAlignment(TEXT_ALIGN_CENTER);
260   display->drawString(32 + x, 0 + y, currentWeather.iconMeteoCon);
261 }
```

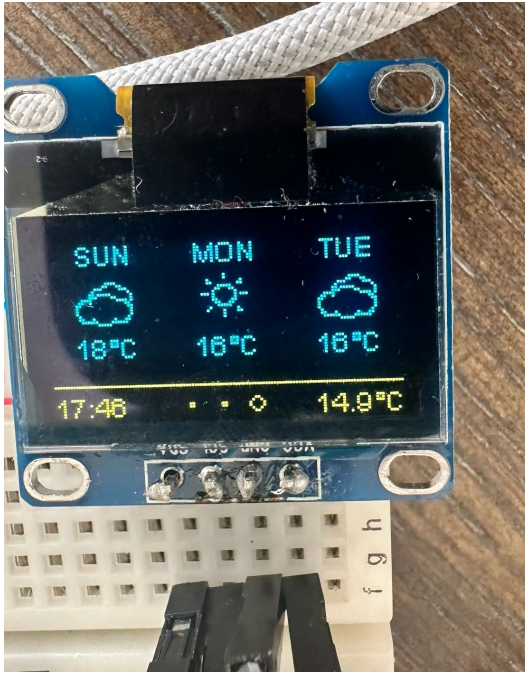




Code

```
260   display->drawString(32 + x, 0 + y, currentWeather.iconMeteoCon);
261 }
262
263 void drawForecast(OLEDDisplay *display, OLEDDisplayUiState* state, int16_t x, int16_t y) {
264   drawForecastDetails(display, x, y, 0);
265   drawForecastDetails(display, x + 44, y, 1);
266   drawForecastDetails(display, x + 88, y, 2);
267 }
268
269 void drawForecastDetails(OLEDDisplay *display, int x, int y, int dayIndex) {
270   time_t observationTimestamp = forecasts[dayIndex].observationTime;
271   struct tm* timeInfo;
272   timeInfo = localtime(&observationTimestamp);
273   display->setTextAlignment(TEXT_ALIGN_CENTER);
274   display->setFont(ArialMT_Plain_10);
275   display->drawString(x + 20, y, WDAY_NAMES[timeInfo->tm_wday]);
276
277   display->setFont(Meteocons_Plain_21);
278   display->drawString(x + 20, y + 12, forecasts[dayIndex].iconMeteoCon);
279   String temp = String(forecasts[dayIndex].temp, 0) + (IS_METRIC ? "°C" : "°F");
280   display->setFont(ArialMT_Plain_10);
281   display->drawString(x + 20, y + 34, temp);
282   display->setTextAlignment(TEXT_ALIGN_LEFT);
283 }
284
285 void drawHeaderOverlay(OLEDDisplay *display, OLEDDisplayUiState* state) {
286   now = time(nullptr);
287   struct tm* timeInfo;
288   timeInfo = localtime(&now);
289   char buff[14];
290   sprintf_P(buff, PSTR("%02d:%02d"), timeInfo->tm_hour, timeInfo->tm_min);
291
292   display->setColor(WHITE);
293   display->setFont(ArialMT_Plain_10);
294   display->setTextAlignment(TEXT_ALIGN_LEFT);
295   display->drawString(0, 54, String(buff));
296   display->setTextAlignment(TEXT_ALIGN_RIGHT);
297   String temp = String(currentWeather.temp, 1) + (IS_METRIC ? "°C" : "°F");
298   display->drawString(128, 54, temp);
299   display->drawHorizontalLine(0, 52, 128);
300 }
301
302 void setReadyForWeatherUpdate() {
303   Serial.println("Setting readyForUpdate to true");
304   readyForWeatherUpdate = true;
305 }
```

Results



Video: https://youtu.be/OUlfw0rF7PQ