



► FAKULTA  
APLIKOVANÝCH VĚD  
ZÁPADOČESKÉ  
UNIVERZITY  
V PLZNI

Západočeská univerzita v Plzni

Fakulta aplikovaných věd

PROJEKT 4

## **IoT - ZÁKLADY SYSTÉMU CHYTRÉ DOMÁCNOSTI**

Patrik Nachtmann

2019

# ZADÁNÍ

## NÁZEV PRÁCE

IoT - základy chytré domácnosti

## POPIS PROJEKTU

Výstupem tohoto projektu bude fyzická realizace jednoúčelového snímacího zařízení (např. teploty, vlhkosti, tlaku, ...). Snímaná data budou pomocí NodeMCU (ESP8266) a protokolu MQTT posílána do centrálního brokeru, odkud je bude možné číst a vykreslit v rozhraní na webu.

## POSTUP PRÁCE

1. Seznámení se s mikročipem ESP8266 a MicroPythonem
2. Seznámení se se základními příkazy (rozsvícení led)
3. Zapojení a komunikace se základními senzory (teploměr, vlhkoměr, tlakoměr,...)
4. Využití protokolu MQTT pro komunikaci se zařízením
5. Zpracování a (např. webové) vykreslení sbíraných dat
6. Návrh fyzické realizace snímacího zařízení
  - 6.1. Sestrojení obvodů pro senzory a NodeMCU
  - 6.2. Vyřešení napájení
  - 6.3. Vytvoření krytu zařízení na 3D tiskárně
7. Stručná dokumentace a popis navrženého systému

Iniciativa studenta podílet se na návrhu zadání je vítána.

# ZPRACOVÁNÍ

## 1. Seznámení se s mikročipem ESP8266 a MicroPythonem

Prvotním úkolem tohoto projektu je stažení a instalace aktuálního firmwaru s MicroPythonem pro mikročip ESP8266 pomocí nástroje esptool. K samotnému flashování firmwaru do mikročipu potřebujeme nejprve zjistit usb port, pod kterým je čip připojen k počítači. V MacOS otevřeme terminál a příkazem:

```
$ ls /dev/cu.*
```

vypíšeme všechna usb zařízení připojená k počítači. Po zjištění označení mikročipu v počítači můžeme přejít k samotnému flashování. Nainstalujeme esptool a pro jistotu nejprve vymažeme flashovací paměť ESP8266 příkazem:

```
$ esptool.py --port /dev/cu.wchusbserial14110 erase_flash
```

Text za atributem --port je konkrétní cesta k mikročipu v počítači, kterou jsme zjistili z výpisu předchozího příkazu.

Po úspěšném naformátování flash paměti můžeme konečně přejít k nahrání firmwaru s MicroPythonem do mikročipu. Použijeme k tomu následující příkaz:

```
$ esptool.py --port /dev/cu.wchusbserial14110 --baud 115200  
write_flash --flash_size=detect -fm dio 0 esp8266-20190125-  
v1.10.bin
```

Na konci příkazu je konkrétní název souboru s firmwarem.

### TIP 1

Pro vymazání flash paměti lze alternativně při současném držení tlačítka FLASH na mikročipu použít příkaz:

```
$ esptool.py --chip esp8266 erase_flash
```

případně jen:

```
$ esptool.py erase_flash
```

esptool.py by měl cestu k čipu najít i bez ručně zadaného portu. Mně se nejvíce osvědčil nejjednodušší (poslední) příkaz, který zafungoval zpravidla vždy na poprvé.

## TIP 2

Doporučuji zkontoľovat USB kabel a ověřit si, že podporuje napájení a zároveň datový přenos. Ušetří to spoustu času při trápení s flashováním. Kabel podporuje datový přenos, pokud po připojení USB do mikročipu a do počítače krátce problikne interní LED dioda.

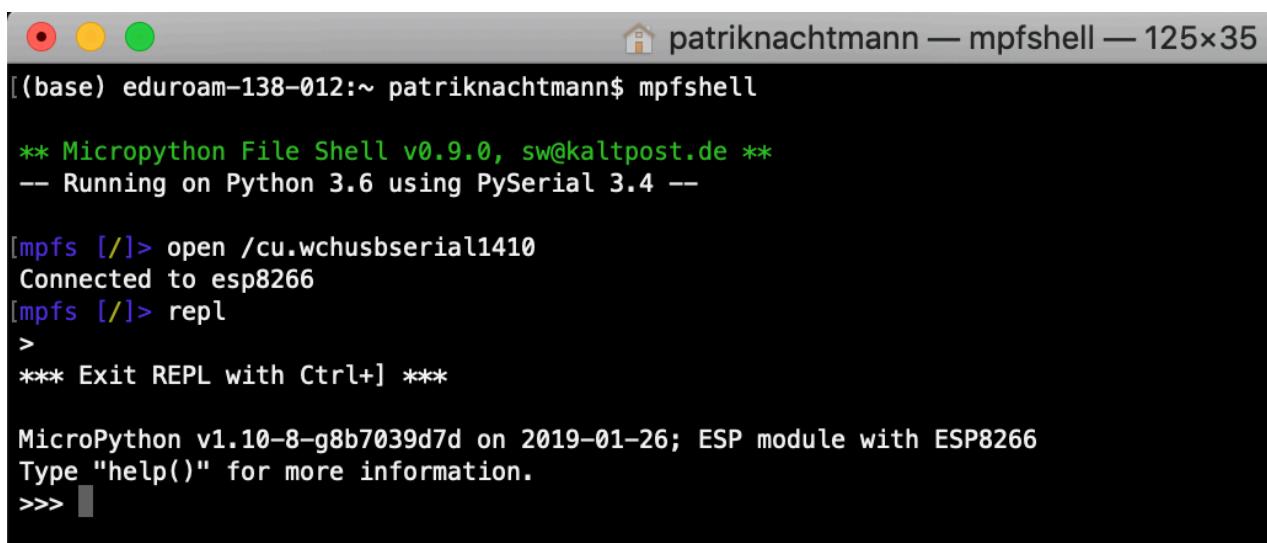
## 2. Seznámení se se základními příkazy

Po úspěšném nahrání firmwaru s MicroPythonem se začneme seznamovat s vlastnostmi ESP8266. Komunikace s mikročipem v počáteční fázi probíhá pouze pomocí usb kabelu, který slouží současně pro napájení a pro datový přenos. Po instalaci firmwaru je možné pomocí nástroje `mpfshell` spravovat úložiště mikročipu na úrovni procházení složek, mazání souborů a kopírování skriptů z počítače na mikročip. Prvním krokem v testování komunikace s mikročipem je rozsvícení interní LED diody. Po úspěšném zvládnutí tohoto kroku jsem pomocí schématu popisu GPIO pinů rozsvítil externí LED diody. Zapojení barevných LED diod je pro zjednodušení realizováno pomocí breadboardu a příslušného odporu. Tyto jednobarevné diody jsou ovládány pouze dvěma výstupy - jedním kabelem s uzemněním GND a jedním kabelem připojeným k příslušnému GPIO pinu. Ovládání jednotlivých pinů přes terminál je zcela intuitivní a k rozsvícení diod stačí pouze několik základních příkazů.

Připojení k ESP8266 pomocí nástroje `mpfshell` provedeme následovně:

```
$ mpfshell
```

```
$ open /cu.wchusbserial14110
```

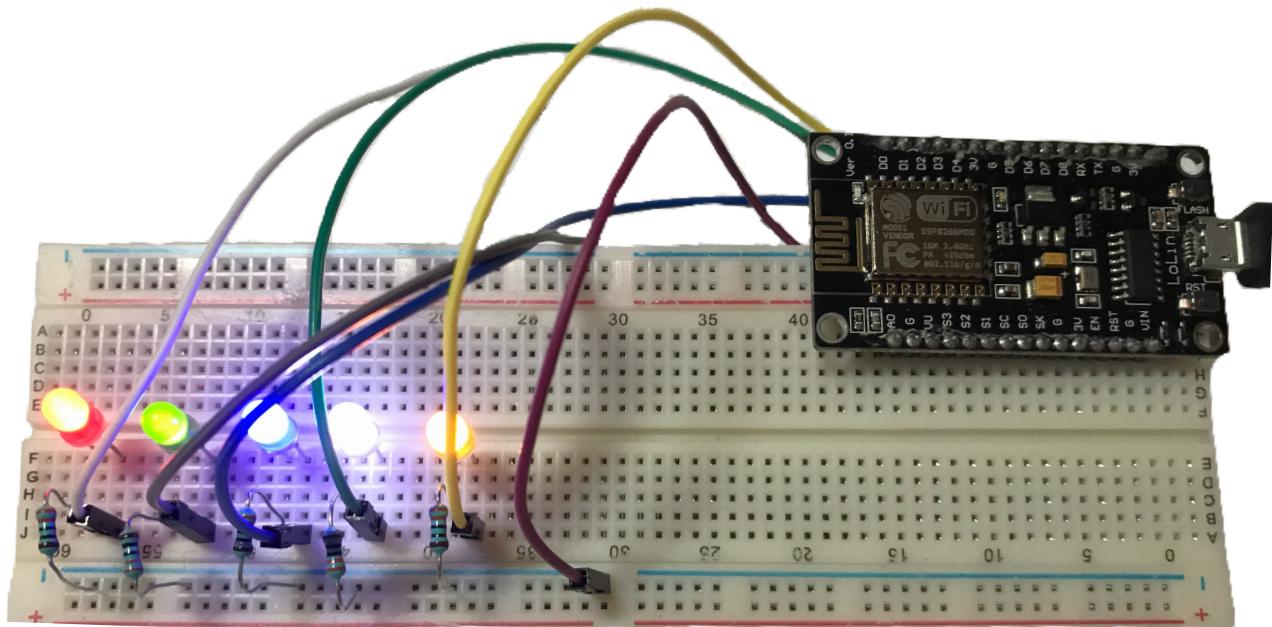


```
patriknachtmann — mpfshell — 125x35
[(base) eduroam-138-012:~ patriknachtmann$ mpfshell
** Micropython File Shell v0.9.0, sw@kaltpost.de **
-- Running on Python 3.6 using PySerial 3.4 --
[mpfs [/]> open /cu.wchusbserial1410
Connected to esp8266
[mpfs [/]> repl
>
*** Exit REPL with Ctrl+]
*** Exit REPL with Ctrl+]

MicroPython v1.10-8-g8b7039d7d on 2019-01-26; ESP module with ESP8266
Type "help()" for more information.
>>>
```

Krátký skript pro rozsvícení interní LED diody:

```
import machine
pin = machine.Pin(2, machine.Pin.OUT)
pin.off()
pin.on()
```



Obrázek 1: Zapojení barevných led diod pomocí breadboardu k mikročipu.

### **3. Zapojení a komunikace se základními senzory**

Stejným principem jsem zapojil teploměr Dallas DS18B20 a napsal skript pro automatické měření teploty každých několik vteřin. Následně jsem otestoval teploměr DHT11, který měří současně s teplotou vlhkost. Data z obou teploměrů jsem porovnal a došel k závěru, že měří téměř shodně a velmi přesně.

```
>>> import Temperature.py
Loading script ...
DS18B20, DHT11 Temperature Measuring Comparison
-----
DHT11 - Temperature: 23 °C
DS18B20 - Temperature: 23.125 °C
DHT11 - Temperature: 23 °C
DS18B20 - Temperature: 23.125 °C
DHT11 - Temperature: 23 °C
DS18B20 - Temperature: 23.125 °C
DHT11 - Temperature: 23 °C
DS18B20 - Temperature: 23.125 °C
DHT11 - Temperature: 23 °C
DS18B20 - Temperature: 23.1875 °C
DHT11 - Temperature: 23 °C
DS18B20 - Temperature: 23.1875 °C
DHT11 - Temperature: 23 °C
DS18B20 - Temperature: 23.1875 °C
DHT11 - Temperature: 23 °C
DS18B20 - Temperature: 23.1875 °C
DHT11 - Temperature: 23 °C
DS18B20 - Temperature: 23.1875 °C
Ending script ...
```

```
[>>> import DHT11.py
Loading script DHT11 ...
-----DHT11-----
Temperaure: 23 °C
Humidity: 34 %
Temperaure: 23 °C
Humidity: 36 %
Temperaure: 23 °C
Humidity: 34 %
Temperaure: 23 °C
Humidity: 34 %
Temperaure: 23 °C
Humidity: 34 %
Temperaure: 24 °C
Humidity: 34 %
Temperaure: 23 °C
Humidity: 34 %
Temperaure: 24 °C
Humidity: 34 %
Temperaure: 24 °C
Humidity: 34 %
Temperaure: 24 °C
Humidity: 34 %
```

Pro čtení teploty z teploměru DS18B20 jsem napsal následující skript:

```
import time, machine, onewire, ds18x20

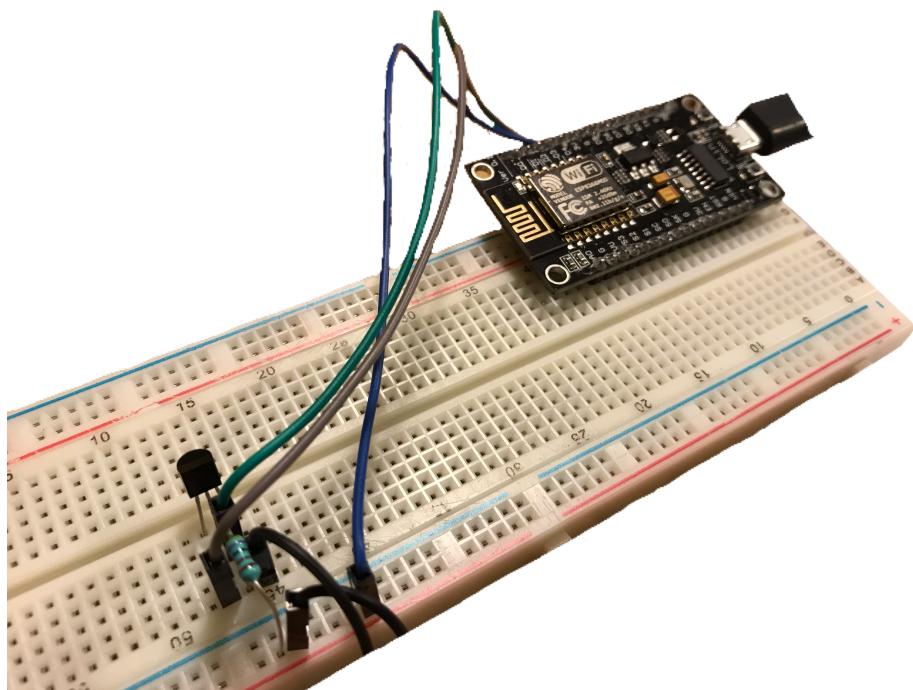
print('Loading script DS18B20 ...')

# device connected to GPIO12
GPIO = 12
data = machine.Pin(GPIO, machine.Pin.IN)

# create the onewire object
ds = ds18x20.DS18X20(onewire.OneWire(data))

# scan for devices on the bus
devices = ds.scan()
if devices:
    print('Found devices:', devices)
else:
    print('No device found.')

print('-----DB18B20-----')
for i in range(10):
    ds.convert_temp()
    time.sleep_ms(750)
    for device in devices:
        temperature = ds.read_temp(device)
        print('Temperature: ', temperature, '° C')
        time.sleep(1)
```



Obrázek 2: Zapojení teploměru DS18B20 pomocí breadboardu k mikročipu.

### **TIP 3**

V úložišti mikročipu je po naflashování defaultně jeden soubor `boot.py`. Tento skript se spustí jako první při kažném zapnutí mikročipu. Je možné vytvořit soubor `main.py`, který se spustí vždy hned po `boot.py`. V tomto skriptu je vhodné volat jednotlivé funkce, které se mají vykonat při každém spuštění ESP8266, například připojení k wifi a změření teploty. Po optimalizaci kódu je tedy možné zcela automatizovat chování mikročipu i v případě restartu. Hodí se to zejména při automatizaci domácnosti a umístění jednotlivých čipů na ne příliš dostupná místa.

### **TIP 4**

Doporučuji dávat pozor a dobře rozmýšlet, jakou sekvenci příkazů píšeme do `boot.py`. Snadno se může stát, že se skript zacyklí například v podmínce `while:True`, nevykoná se celý kód a není možné se dostat do `repl`. Pak pomůže už jen přeflashování mikročipu.

### **TIP 5**

Po připojení k mikročipu pomocí `mpfshell` se často hodí ho softwarově restartovat proto, aby se znova spustil skript `boot.py` a `main.py` - tentokrát s výpisem do okna terminálu. V `mpfshell` přejdeme do `repl` a provedeme soft restart stisknutím kláves `CONTROL + D`.

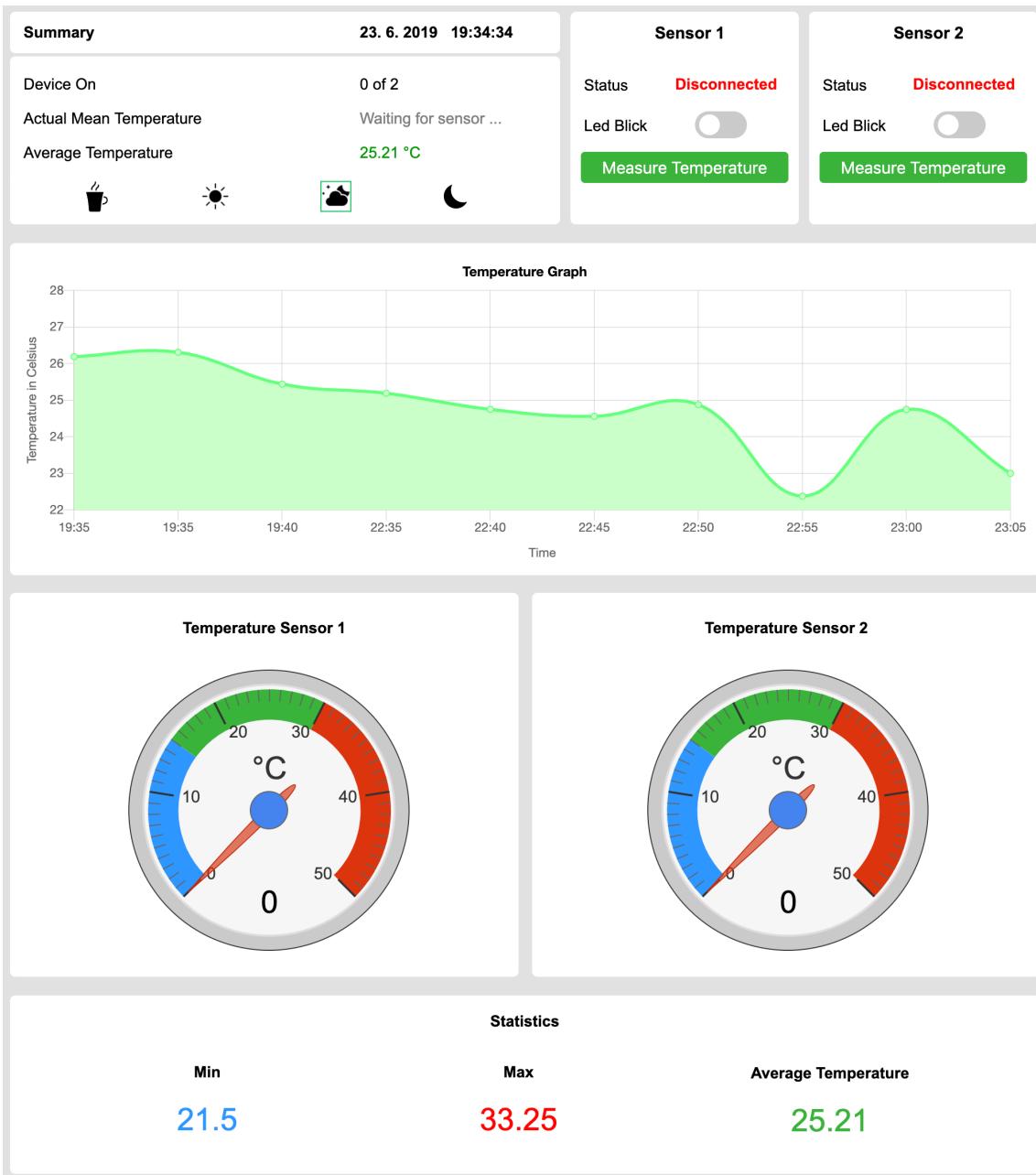
### **TIP 6**

Často se hodí přepínat mezi `repl` oknem a `mpfshell` oknem. Ukončení `replu` a přechod zpět do `mpfshell` je možné dosáhnout stisknutím kláves `CONTROL + C`. Není tedy nutné pokaždé mikročip restartovat.

## **4. Využití protokolu MQTT pro komunikace se zařízením**

Protokol MQTT je jeden z možných způsobů komunikace s ESP8266 bez nutnosti použití USB kabelu. Tato komunikace probíhá čistě prostřednictvím sítě a je založena na principu `publish` a `subscribe`. V síti se musí nacházet broker, kterého můžeme vnímat jako server, který přijímá data od jednotlivých mikročipů (`publisherů`). Mikročip tedy neuchovává žádná data, pouze získává hodnoty z čidel, které následně po síti pošle brokerovi. Mikročip publikuje data pod daným tématem (`topicem`), což umožňuje udržet přehledný systém dat v síti. Díky javascriptu je možné na broker napojit webovou stránku a vytvořit přehledné webové vykreslení.

## 5. Zpracování a webové vykreslení dat



Získaná data se ukládají na server a pomocí webové aplikace vykreslují v prohlížeči. Webová stránka je navržena s ohledem na jednoduchost a přehlednost. V levém horním rohu je stručný přehled poskytující informace o aktuální teplotě (průměr aktuálních teplot z obou čidel), dlouhodobé průměrné teplotě, počtu aktivních čidel a grafické znázornění aktuální části dne. V pravém horním rohu jsou ovládací panely obou čidel. Status teploměru nabývá hodnot "OK", "Sleeping", "Disconnected" a "Subscribing". Pokud je čidlo v "Subscribing" módu, je možné tlačítkem Led Blick rozsvítit interní led diodu a otestovat tak obousměrnou komunikaci mezi teplotním senzorem a webovou stránkou. Stejně tak je možné stisknutím tlačítka změřit aktuální teplotu. Všechny naměřené teploty jsou na serveru ukládány do jednoho .json souboru, odkud je načítá webová stránka.

## **6. Fyzická realizace teploměru**

Prvotním úkolem v realizaci snímacího zařízení je navržení a sestrojení obvodů pro teploměr a senzor. Rozhodl jsem se použít teploměr DS18B20, hlavě z důvodu malých rozměrů a přesného měření. Tento senzor je k NodeMCU připojen třemi kably - napájení 3V, uzemění GND a datový přenos. Všechny kably jsou k jednotlivým výstupům připojeny a odizolovány, aby se zamezilo nechtěnému odpojení kabelu. Jelikož výstupy senzoru nejsou nijak chráněné, je nutné je odizolovat, aby nedocházelo ke zkratu.

## **Závěr**

Během projektu se podařilo sestrojit 2 chytré teplotní čidla, které v pravidelných časových intervalech odesílají teploty na server. Na serveru se z teplot počítají statistiky, které jsou pak pro přehlednost vykresleny ve webové vizualizaci. Celkové se tento projekt povedl a je možné na něj plynule navázat v dalších semestrech.