



Uniwersalna platforma chmurowa w
zastosowaniach Internet Of Things

Praca z fizycznym urządzeniem

Warsztaty techniczne
Skrypt do ćwiczeń

2019-04-28, wersja druga

Autor: Maciej Szulc, IBM, maciej.szulc@pl.ibm.com

ACHITEKTURA TWORZONEGO ROZWIAZANIA

Niniejsze drugie wspólne ćwiczenie będzie dotyczyło budowy własnego, sprzętowego sensora IoT współpracującego z platformą IBM Watson IOT Platform.

Do budowy rozwiązania użyjemy:

- Płytki prototypowej NodeMcu zawierającej procesor ESP8266 i interfejs USB
- Scalonego termometru Maxim DS18B20

ESP8266 to chętnie używana (zarówno w projektach komercyjnych, jak i hobbystycznych) rozwiązanie integrujące wiele przydatnych w środowiskach IoT technologii:

- 32-bitowy procesor RISC taktowany zegarem 80MHz
- 64 KiB RAM na program,
- 96 KiB RAM na dane,
- 512 KiB do 4 MiB FLASH na firmware,
- do 16 linii cyfrowych we/wy,
- jedno 10-bitowe wejście ADC,
- obsługa SPI, I2C, I2S, UART,
- WiFi 802.11 b/g/n,
- wspiera zabezpieczenia WPA / WPA2
- może pracować w trybie AP (Access Point), STA (standalone) oraz AP+STA



Programowanie procesora może odbywać się w wielu językach i przy wykorzystaniu wielu środowisk IDE. W naszym ćwiczeniu skorzystamy z popularnego IDE Arduino i języka C++, choć dostępne są środowiska IDE o większej użyteczności praktycznej (np. Atom, VisualStudio Code).

Nasza płytka podłączy się do platformy MQTT i wysyłała będzie dane pomiarowe z termometru w postaci komunikatu o nazwie „status” i formacie:

```
"d": {  
    "temp": <float>,  
    "stamp": <int>  
}
```

KROKI ĆWICZENIA

Pierwszy krok to instalacja środowiska programistycznego Arduino, w którym będziemy tworzyć nasz projekt.

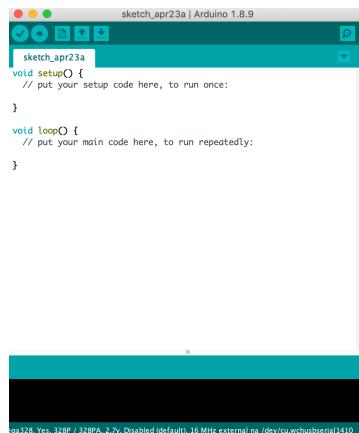
Musimy również zdefiniować w platformie IoT nowy typ urządzenia i uzyskać dane niezbędne do

1. INSTALACJA ARDUINO IDE

1. Wejdź na stronę <https://www.arduino.cc/en/Main/Software>
2. Pobierz i zainstaluj Arduino IDE na swoją platformę systemową



3. Uruchom środowisko Arduino

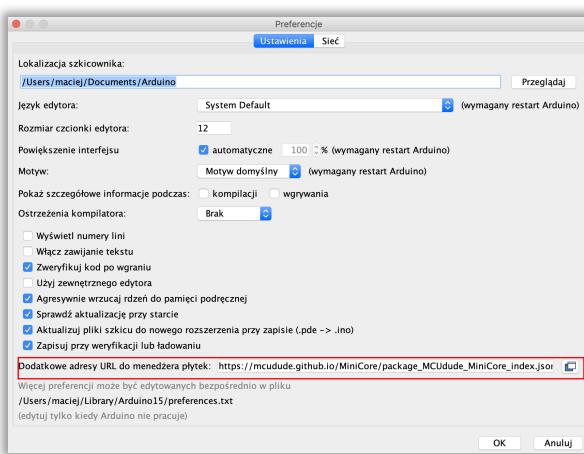


2.DODANIE WSPARCIA DLA PROCESORA ESP8266

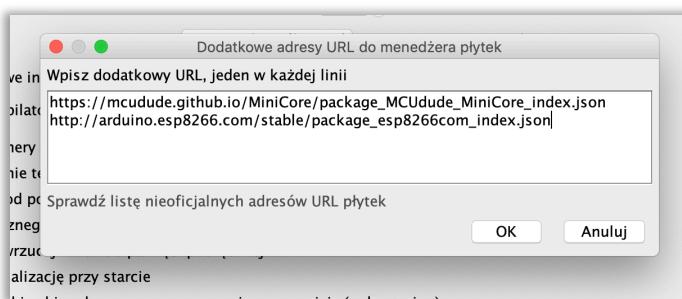
Środowisko Arduino zostało inicjalnie przygotowane na potrzeby płyt prototypowych opartych na procesorach AtMega. Ze względu na popularność wkrótce zostało dostosowane do wymogów innych platform sprzętowych.

Standardowo Arduino IDE nie zawiera wsparcia dla procesora ESP, jednak bardzo łatwo można to zmienić.

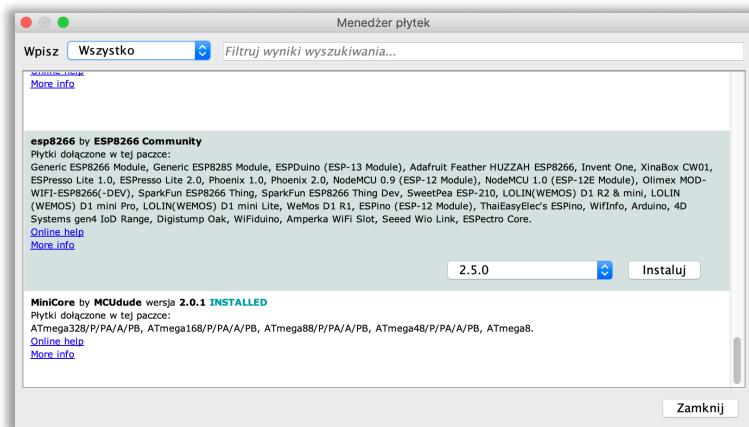
4. Wybierz z menu Arduino->Preferences (na Windows Plik->Preferencje)
5. Otwórz pole „Dodatkowe adresy URL dla menedżera płytEK”



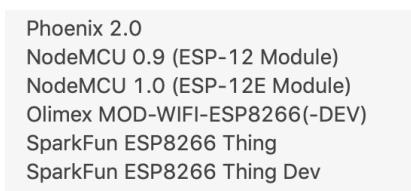
6. Wpisz dodatkowy adres (istniejące pozostaw bez zmian), kliknij OK:
http://arduino.esp8266.com/stable/package_esp8266com_index.json



7. Wybierz Narzędzia->Płytki (...) -> Menedżer płytEK
8. Znajdź pozycję „esp8266” i ją zainstaluj przyciskiem „Instaluj”, poczekaj na zakończenie instalacji pakietu i zamknij okno menedżera.



9. Wybierz Narzędzia->Płytki (...) i kliknij na „NodeMCU 0.9”

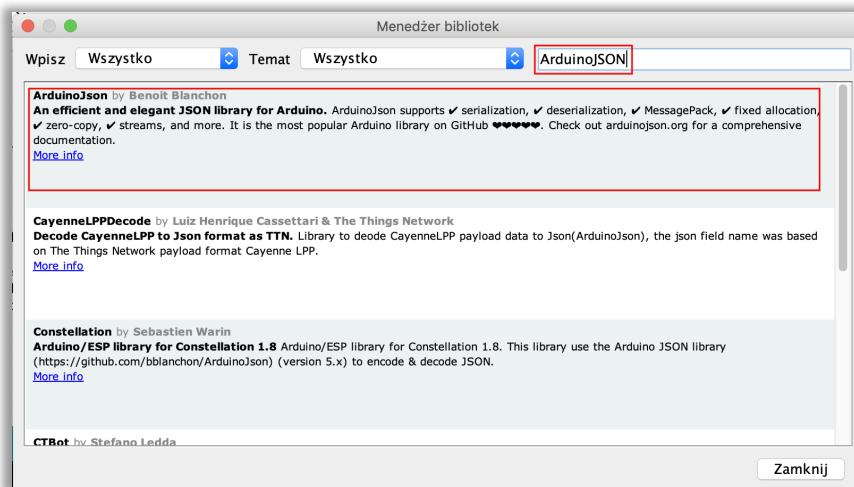


3.NASZ PIERWSZY PROGRAM DLA ESP8266

Dla środowiska Arduino powstało wiele gotowych bibliotek ułatwiających tworzenie kodu. My dodamy biblioteki związane z protokołem MQTT oraz obsługą scalonych czujników temperatury Maxim (kiedyś: Dallas)

10. Teraz dodamy do środowiska niezbędne biblioteki. W tym celu wybierz „Narzędzia”-> „Zarządzaj bibliotekami”, następnie wyszukaj i zainstaluj:

- ArduinoJSON
- PubSubClient
- OneWire
- Dallas Temperatrure



Sprawdźmy, czy środowisko IDE działa prawidłowo poprzez skompilowanie przykładowego programu na platformę esp8266.

W tym celu skopiuj w okno edytora kod programu pobrany z:

<https://github.com/maciejs20/IoTWorkshop-part2/tree/master/01.blink>

```
#define LED 2

void setup() {
  Serial.begin(115200);
  Serial.println ("");
  pinMode(LED, OUTPUT);      // LED pin as output.
  Serial.println ("Start");
}

void loop() {
  digitalWrite(LED, HIGH);
  delay(1000);               // wait for 1 second.
```

```

    digitalWrite(LED, LOW);      // turn the LED on.
    delay(1000);                // wait for 1 second.
    Serial.println ("Loop");
}
}

```

11. Następnie skompiluj go przyciskiem

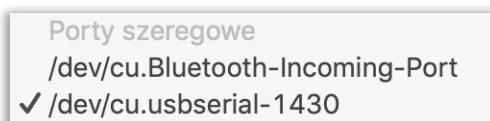


12. Podłącz do portu USB płytę NodeMCU którą otrzymałeś do wykonania ćwiczenia.

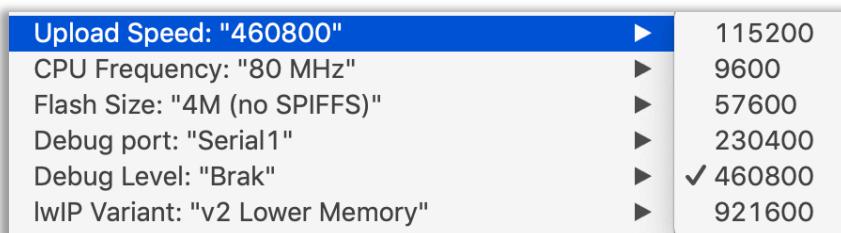
W zależności od systemu operacyjnego może być niezbędna instalacja drivera do konwertera usb-serial umieszczonego na naszej płytce – CH340. Dalszą pomoc możesz odnaleźć tu:

http://www.wch.cn/download/CH341SER_EXE.html	– strona driverów producenta konwertera (Windows)
http://www.wch.cn/download/CH341SER_MAC_ZIP.html	– strona driverów producenta konwertera (Mac)
https://sparks.gogo.co.nz/ch340.html	– instrukcja instalacji/deinstalacji na wszystkie platformy

13. Wybierz Narzędzia -> Port i wybierz port szeregowy odpowiadający podłączonej płytce, np. jak poniżej:



14. Wybierz Narzędzia->Upload Speed i ustaw na 460800. W razie problemów z komunikacją z płytą możesz zmniejszyć prędkość na 115200.



15. Wgraj skompilowany program przyciskiem:



16. Sprawdź, czy nie pojawiły się błędy. Po wgraniu płytki powinna stale migać niebieską diodą LED z częstotliwością około 1Hz.



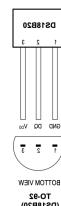
17. Jeśli wybierzesz Narzędzia -> Monitor portu szeregowego, a następnie ustawisz prędkość portu 115200 baud, to będziesz mógł zobaczyć komunikaty wysyłane przez płytę do portu szeregowego (polecenia Serial.print).

18. Jeśli płytki miga, to udało Ci się przygotować środowisko programistyczne i możesz przejść do dalszej części ćwiczenia.

4. PODŁĄCZAMY CZUJNIK TEMPERATURY

Uwaga: ze względu na możliwość uszkodzenia czujnika dokładnie sprawdź poprawność połączeń!!!

Do ćwiczeń będziemy używali sensora DS18B20 firmy Maxim, który jest scalonym termometrem cyfrowym mierzącym temperaturę w zakresie -55 do 125 stopni Celsjusza, z dokładnością ok. +/-0.5 stopnia w zakresie -10..+85 stopni i nieco mniejszą w pozostałym przedziale.



Nasz termometr ma trzy wyprowadzenia:

- czerwone – zasilanie 3V
- czarne – masa (gnd)
- brązowe – komunikacja 1Wire



Najpierw odłącz płytkę od komputera, wszystkie podłączenia przewodów KONIECZNIE wykonuj po odłączeniu kabla USB! Jest to niezwykle ważne, nie możesz tego kroku pominąć!

19. Odłącz kabel usb od płytki ESP8266

20. Przewody termometru należy podłączyć do płytki ESP w następujący sposób:

- **Czarny** – do złącza **G**
- **Czerwony** – do złącza **3V**
- **Brązowy** – do złącza **D4**



21. Sprawdź ponownie poprawność połączeń!

22. Podłącz ponownie kabel USB do płytki

23. Utwórz nowy *sketch*



i skopiuj w okno edytora kod programu pobrany z:

<https://github.com/maciejs20/IoTWorkshop-part2/tree/master/02.tempTest/tempTest>

24. Skompiluj kod i zaprogramuj płytke



25. Otwórz „Narzędzia”->„Monitor portu szeregowego”, ustaw prędkość 115200 i obserwuj działanie programu

```
load 0x04010f000, len 1384, room 00
Start: 5
Start: 4
Start: 3
Start: 2
Start: 1
Setup.
Temperatura: 25.81
Temperatura: 25.81
Temperatura: 25.81
Temperatura: 25.81
Temperatura: 25.87
Temperatura: 27.00
Temperatura: 28.12
Temperatura: 28.00
```

26. Sprawdź, czy po dotknięciu termometru temperatura się zmienia.

5. PODŁĄCZAMY WIFI, MQTT I WYSYŁAMY DANE POMIAROWE

27. Utwórz nowy *sketch*

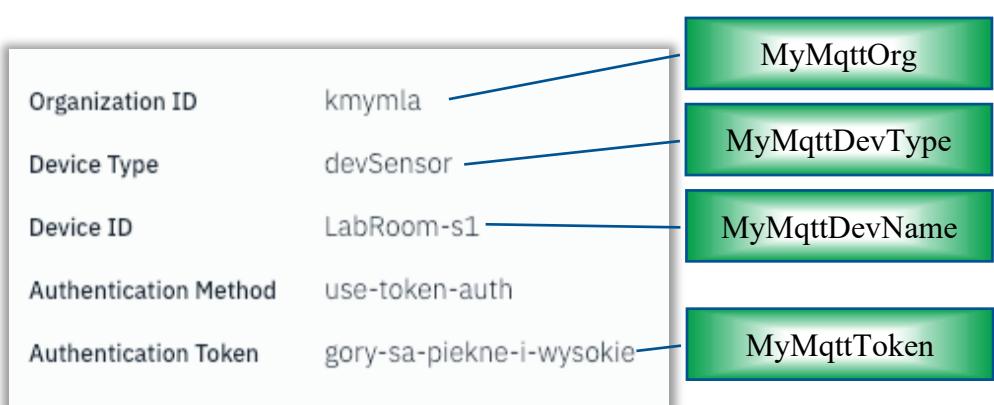


i skopiuj w okno edytora kod programu pobrany z:

<https://github.com/maciejs20/IoTWorkshop-part2/tree/master/03.mqttConn/mqttConn>

28. Znajdź w kodzie programu sekcję oznaczoną jako „Zmień”:

29. Na podstawie podręcznika do poprzednich ćwiczeń (dwa tygodnie temu) zaloguj się do swojej platformy IBM Watson IOT Platform i utwórz nowy typ urządzenia i nowy identyfikator urządzenia. Po ich utworzeniu odpowiednie wartości wpisz do konfiguracji programu:



30. Skompiluj kod i zaprogramuj płytę



31. Otwórz „Narzędzia”->„Monitor portu szeregowego”, ustaw prędkość 115200 i obserwuj działanie programu

```
load 0x4010f000, len 1384, room 000*  
Start: 5  
Start: 4  
Start: 3  
Start: 2  
Start: 1  
Setup.  
Podłączam się do Wifi.  
Podłączony do: muezin-ioths, IP:10.10.10.49 RSSI:-62dBm  
Podłączam się do: fzxluu.messaging.internetofthings.ibmcloud.com jako d:fzxluu:iotWorkshop:iot-1  
Podłączycię się do mqtt.  
mqtt_pub OK dla: {"d": {"temp": 25.875, "stamp": 28}}  
mqtt_pub OK dla: {"d": {"temp": 25.8125, "stamp": 44}}  
mqtt_pub OK dla: {"d": {"temp": 25.8125, "stamp": 61}}  
mqtt_pub OK dla: {"d": {"temp": 25.75, "stamp": 77}}  
mqtt_pub OK dla: {"d": {"temp": 25.75, "stamp": 93}}  
mqtt_pub OK dla: {"d": {"temp": 25.75, "stamp": 109}}  
mqtt_pub OK dla: {"d": {"temp": 25.75, "stamp": 125}}
```

32. Jeśli pojawiły się błędy – sprawdź ich przyczynę i skoryguj je. Zwykle mają one związek z błędnym ustawieniem parametrów połączenia MQTT

33. Zaloguj się do platformy IBM MQTT i sprawdź, czy dane są odbierane przez platformę:

DEVICE DRILLDOWN

Connection Information

Recent Events

State

Device Information

Metadata

Extension Configuration

Diagnostics

Connection Logs

Device Actions

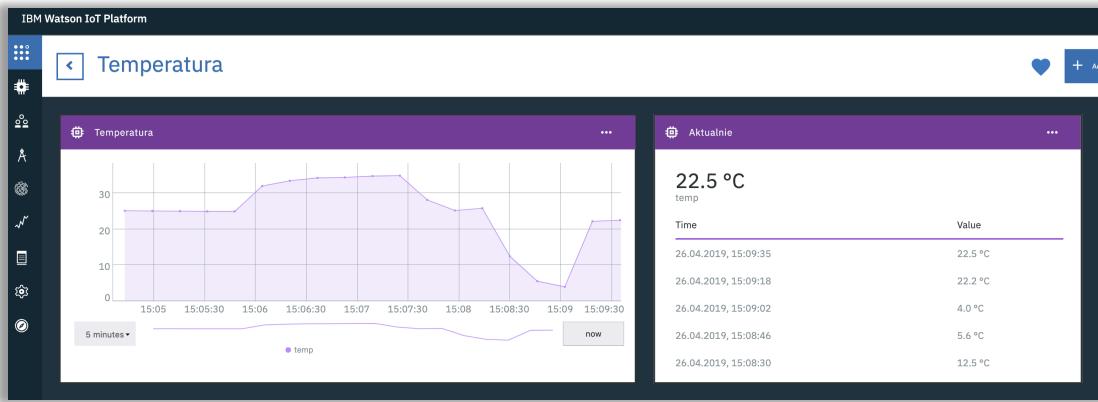
State

This table shows a list of data points that are reported by this device.

Showing Raw Data | No Interfaces Available

Property	Value	Type	Event	Last Received
d		Object	status	kilka sekund temu
temp	25.5	Number	status	kilka sekund temu
stamp	167	Number	status	kilka sekund temu

34. Na podstawie instrukcji z poprzednich ćwiczeń dobuduj prostą wizualizację w platformie lub w Node-RED, np. taką jak poniżej:



6. ROZBUDOWUJEMY PROGRAM

35. Jeśli czujesz się na siłach, to spróbuj rozbudować nasz program o wysyłanie dodatkowych informacji:

- Aktualnego SSID do którego podłączona jest platforma ESP

```
| IPAddress WiFi.localIP();
```

przykład użycia:

```
| if (WiFi.status() == WL_CONNECTED)
{
    Serial.print("Connected, IP address: ");
    Serial.println(WiFi.localIP());
```

- Adresu MAC interfejsu pobieranego poleceniem:

```
| uint8_t[6] WiFi.macAddress();
```

przykład użycia:

```
| uint8_t macAddr[6];
WiFi.macAddress(macAddr);
Serial.printf("Connected, mac address: %02x:%02x:%02x:%02x:%02x:%02x\n", macAddr[0],
macAddr[1], macAddr[2], macAddr[3], macAddr[4], macAddr[5]);
```

- Mocy sygnału WiFi pobieranego poleceniem:

```
| int32_t WiFi.RSSI();
```

przykład użycia:

```
| Serial.printf("RSSI: %d dBm\n", WiFi.RSSI());
```

- Prędkości zegara procesora pobieranego poleceniem:

```
| system_get_cpu_freq();
```

po dodaniu pliku nagłówkowego:
| #include "user_interface.h"