MQTT Broker & Firebase Manager

# MQTT Broker

## Użyte biblioteki

MQTTnet – biblioteka .NET o wysokiej wydajności implementująca podstawową kominukację przez protokół MQTT. Zapewnia wsparcie zarówno dla klienta jak i serwera.

Główne cechy:

* Wsparcie operacji asynchronicznych
* Niskopoziomowa implementacja MQTT (Lightweight)
* Optymalizacja wydajności
* Wbudowane testy jednostokowe

Wspierane Frameworki:

* .NET Standard 1.3+
* .NET Core 1.1+
* .NET Core App 1.1+
* .NET Framework 4.5.2+ (x86, x64, AnyCPU)
* Mono 5.2+
* Universal Windows Platform (UWP) 10.0.10240+ (x86, x64, ARM, AnyCPU, Windows 10 IoT Core)
* Xamarin.Android 7.5+
* Xamarin.iOS 10.14+

## Kompilacja i działanie Brokera

MQTT Broker został zbudowany na platformę .NET Core i procesory ARM. Kod aplikacji jest dostępny na github`ie.

Krótki opis fukncji:

* StartBroker() – Uruchomienie brokera
* MqttServer\_ApplicationMessageRecevied() – event handler odpowiedzialny za odbiór wszystkich wiadomości wysłanych do servera MQTT
* MqttServer\_ClientConnected() – event handler odpowiedzialny za podłączenie do servera nowych klientów
* ConvertPayloadToDouble() – konwersja otrzymanego payloadu wiadomości z czujnika na tablice zmiennych typu double.

# Firebase Manager

DLL (Dynamic linked library), czyli biblioteka odpowiedzialna za przesyłanie danych z serwera do bazy danych w chmurze google`a czyli Firebase.

## Użyte biblioteki

Firesharp – Firebase REST API wrapper dla platform .NET i Xamarin.

Główne feature`y:

* SET
* PUSH
* GET
* Update
* Delete
* PushTaskAsync

Firesharp zapewnia pełną obsługę bazy danych z poziomu aplikacji opartej o .NET.

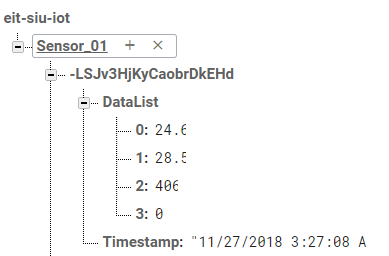
## Kompilacja i działanie Firebase Manager

Firebase Manager został zbudowany na platformę .NET Core i procesory ARM. Kod aplikacji jest dostępny na github`ie.

Krótki opis (ze względów bezpieczeństwa dokument nie będzie zawierał fragmentów kodu źródłowego):

Firebase Manager łączy się do Firebase przy u zyciu uniklanego klucza przypisanego bezpośrednio do aplikacji. Wszystkie wiadomości są wysyłane asynchronicznie dzięki czemu nie ma problemów z gubieniem wiadomości.

## Schemat



* Eit-siu-iot – nazwa bazy danych
* Sensor\_ID – sensor o konkretnym ID
* Uniklany klucz wpisu
* Lista danych i Timestamp

# Podsumowanie

Obie aplikacje są zainstalowane na Raspberry Pi 3, z systemem oprecyjnym Win 10 IoT. Serverem można zarządzać bezpośrednio z raspberry lub przez protokół SSH.

Aplikacje natywnie wyświetłają wszyskie wiadomości i zdarzenia, w celu wykrycia ewentulanych błędów.