Unity & IoT- an MQTT implementation

Bachelor project for game development students

Important! Unity version: 2019.1.3f1

This is an application made in Unity Game Engine that uses MQTT-library to set up basic IoT-ready applications.

Summary

1. In the plugin folder in the project window you can find a .dll file with the nescessary libraries to setup an MQTT-connection to any broker of your choosing. In this project some classes make use of this library and accesses it through a wrapper class that encapsulates those functions that are necessary for a connection to be made. The wrapper class can utilize classes and functions created outside Unity as managed .NET assemblies. See the class named MqttClientExtension that implements MqttClient to see how this is implemented.

2. A connection to a broker requires the broker's IP address and port, a.k.a. socket. These two parameters are represented by two variables that can easily be changed in the “Unity Inspector”- view, or in the source code. When the plugin is first downloaded, the parameters are set to connect to the HiveMQ public broker. To change this go to the Broker Connection game object in the inspector and uncheck the boolean 'Use HiveMq Broker' and insert an IP address of your own choosing if you would like to use another broker for your application.

3. When a connection to a broker is established, MQTT-messages can be sent via topics. In the project demo you can see that different topics can be added in the topics menu under 'Topics'. To receive a message a client must be subscribed to the topic that the MQTT-message is being sent to. Created topics are saved as buttons. To bring up the topic menu press the button and a drop-down menu will show all your topics availabe during runtime. To create a topic just type any name into the input field, “Add new topic…”. This will automatically subscribe the user to the inserted topic with the selected Quality of Service level. To unsubscribe from a topic, bring up the topics menu again, and press the button of a subscribed topic (the text turns red instead of green).

4. To subscribe to a topic that is already in the list, just press the button of the specific topic again. It is also possible to remove topics by pressing the “X” marked button on the topic button.

5. Closing and restarting the app will reset any history of inserted topics. This can be solved with Unity’s PlayerPrefs class. See Unity's documentation.

6. To send messages to topics go to the message menu. In the topics drop down you choose a topic to send a message to and in the text area below you can type the message and be sent when pressing the “Send Message”-button. Typing "flashlight on" (without quotation) any andriod phone subcribing to this topic on the same broker will have its camera flashlight

turned on. Sending the message "flashlight off" will work the same except (yes you guessed it) the camera flashlight off again. Sent messages are stored for further use so that the you don’t have to retype the message every time you want to send the same one.

7. Take a look around in the project and play around with the prefabs and hopefully you'll become comfortable enough with the MQTT-library to create new script and your own implementations of the application.

Note: In order for some of the GUI functionality to be possible, the MQTT-library has been modified. For example, subscribing to a topic which creates a GUI button requires direct access to Unity’s Monobehaviour library in which all of Unity’s game objects can be found. As the MQTT library does not recognize Monobehaviour the subscribe function needed to be overridden inside the wrapper class.