Ce TP s'effectue sous Windows mais peut être facilement adapté sous Linux.

**Prérequis** : programmation objet en C++, bibliothèque Qt, et EDI Qt créator.

La bibliothèque MQTT n'est pas inclue dans la version gratuite de Qt. Il est cependant possible de l'y ajouter.

Un interpréteur PERL est nécessaire pour installer QtMQTT, <https://www.perl.org/get.html>.

La dernière version de MQTT pour Qt pose parfois des problèmes de compilation. **Il est préférable d'utiliser la version 5.14.2.**

****

Dans Qt\5.14.2\Src\ ouvrir une fenêtre de commande (entrer CMD dans le chemin du dossier de la fenêtre Windows) :

SI vous disposer de GIT :

git clone https://code.qt.io/qt/qtmqtt.git --branch 5.14.2

sinon récupérer le zip ici <https://github.com/qt/qtmqtt/tree/5.14.2> et créer un dossier qtmqtt dans SCR,

et le décompresser dans le dossier MQTT.

Ouvrir avec un explorer le dossier qtmqtt et lancer MQTT.pro.

Ajouter une étape build avec l'argument INSTALL :



Lancer la compilation (cliquer sur le marteau).
S'il n'y a pas d'erreur de compilation, la bibliothèque MQTT est installé.

Pour l'utiliser il faudra s'assurer que network et mqtt sont liés à Qt dans le fichier de configuration.pro du projet.

Exemple :



qtmqtt a été installé dans le dossier SRC de Qt, **il n'est pas possible** de compiler les projets vers un exécutable dans ce dossier.

### Premier essai de QtMqtt

Recopier le dossier simpleclient vers votre dossier de travail :

Dans le dossier simpleclient de votre espace de travail lancer simpleclient.pro

Compiler et exécuter le projet en mode release ou debug (flèche verte)

Entrer **test.mosquitto.org** comme hôte mqtt :
Cliquer "connect" ;
State 0 : Deconnecté ;
State 1 : Connexion ;
State 2: Connecté.

Cliquer "Subscibe" pour un abonnement au topic topic1.
Cliquer "Publish" le message est envoyé à mosquitto qui le retourne puisque l'on est abonné au topic.
Cliquer "Disconnect" pour déconnecter de mosquitto, le state repasse à 0.



Le fichier mainwindow.cpp présente un exemple de méthodes de la bibliothèque qtmqtt pour créer un client mqtt.

m\_client = new QmqttClient(this); m\_client est une instance d'un objet de type QmqttClient

m\_client->setHostname(ui->lineEditHost->text()); on entre ici le nom du broker

m\_client->setPort(ui->spinBoxPort->value()); on entre ici le numéro du port

Pour donner l'adresse du broker sans passer par lineEdit :

m\_client->setHostname("test.mosquitto.org");

Pour donner le numéro de port sans passer par spinBox :

m\_client->setPort(1883);

La méthode MqttClient::messageReceived retourne un Qstring nommé message contenant le payload mqtt.

Les méthodes stateChanged et disconnected revoient vers les méthodes d'ouverture et de fermeture de connexion.

Le bouton **buttonConnect** appelle essentiellement les méthodes :

m\_client->connectToHost() ;

et
m\_client->disconnectFromHost() ;

qui établissent ou suppriment la liaison mqtt.

Le bouton **buttonSubscribe**  réalise la souscription à un topic :

auto subscription = m\_client->subscribe(ui->lineEditTopic->text()) ;

Le bouton **buttonPublish** publie le payload de lineEditMessage :

m\_client->publish(ui->lineEditTopic->text(), ui->lineEditMessage->text().toUtf8())

ou

m\_client->publish(nom\_du\_topic, payload).

La documentation de la bibliothèque qtmqtt est ici : <https://doc.qt.io/QtMQTT/index.html> .

### Exercice

À partir du node réalisé sur Raspberry Pi et de l'image ci-dessous :

* Réaliser un programme avec Qt affichant la température sur un QtLCD et disposant d'un bouton permettant d'allumer ou d’éteindre la LED.
* Pour cela recopier à nouveau l'exemple "simpleclient" dans un nouveau dossier de votre dossier de travail.
* Dans Qt creator, cliquer-droit sur la fenêtre d'édition puis "mettre en page -> casser la mise en page" ce qui vous permettra d'ajouter des widgets.

Ajouter m\_client->setClientId(ui->lineEditID->text()) ;

pour donner une identification sur mosquitto qui sera différente de celle sur le Rpi ou MQTT.fx.

