02

Publicando mensagem

Transcrição

Falta agora implementar o código referente ao MQTT. Primeiramente, importamos a sua biblioteca e criamos uma constante com o seu endereço:

```
// --- MQTT ---
#include <PubSubClient.h>
const char* mqtt_Broker = "iot.eclipse.org";

// restante do código omitido
```

Primeiramente faremos o teste para o servidor público. Para testar no servidor do Raspberry Pi, basta trocar o valor da constante para o seu IP.

Após isso, criamos um objeto, passando o cliente Wi-Fi para ele. Com esse objeto em mãos, nós inicializamos o servidor, passando o endereço e a porta para a função `setServer`:

```
// --- MQTT ---
#include <PubSubClient.h>
const char* mqtt_Broker = "iot.eclipse.org";
PubSubClient client(nodemcuClient);

void setup() {
  configurarDisplay();
  conectarWifi();
  client.setServer(mqtt_Broker, 1883);
}

// restante do código omitido
```

Para os tópicos que receberão as mensagens, criamos duas constantes com o valor de cada um:

```
// --- MQTT ---
#include <PubSubClient.h>
const char* mqtt_Broker = "iot.eclipse.org";
PubSubClient client(nodemcuClient);
const char* topicoTemperatura = "labrmerces/temperatura";
const char* topicoUmidade = "labrmerces/umidade";

// restante do código omitido
```

Agora, criamos uma função para nos reconectarmos ao servidor MQTT:

```
/* Exibindo somente a função
reconectarMQTT(), omitindo o
restante do código */
```

```
void reconectarMQTT() {
  while (!client.connected()) {
    client.connect();
  }
}
```

Chamaremos essa função dentro da função `loop()` mais à frente. Falta agora criar a função de publicação no tópico, chamada `publicarTemperaturaUmidadeNoTopico()`. Para publicar algo no tópico, chamamos a função `publish`, passando para ela o tópico, a mensagem (que é uma string, logo precisamos converter os valores de temperatura e umidade), e o valor `true`, para que a mensagem fique salva no *broker* até a próxima publicação:

```
/* Exibindo somente a função
publicarTemperaturaUmidadeNoTopico(),
omitindo o restante do código */

void publicarTemperaturaUmidadeNoTopico() {
  client.publish(topicoTemperatura, String(temperatura).c_str(), true);
  client.publish(topicoUmidade, String(umidade).c_str(), true);
}
```

Falta agora chamar dentro de `loop()` as funções que acabamos de criar. Vamos verificar se o cliente está conectado, caso não esteja nós chamamos a função de reconexão, e com o cliente conectado, publicamos as mensagens no tópico:

```
/* Exibindo somente a função
loop(), omitindo o restante do código */

void loop() {
  if (!client.connected()) {
    reconectarMQTT();
  }
  medirTemperaturaUmidade();
  mostrarTemperaturaUmidade();
  publicarTemperaturaUmidadeNoTopico();
}
```

Com o código pronto, vamos ajustá-lo no próximo vídeo.