

Para instalar no Windows, faça o download de <https://mosquitto.org/files/binary/win64/mosquitto-2.0.10-install-windows-x64.exe> ou <https://mosquitto.org/files/binary/win32/mosquitto-2.0.10-install-windows-x86.exe> dependendo a versão do seu Windows

Prompt 1: Serviço (como admin)

```
C:\Program Files\mosquitto>mosquitto -v
```

Prompt 2: Cliente (subscribe)

```
C:\Program Files\mosquitto>mosquitto_sub -h localhost -p 1883 -t "teste"
```

-h (host) pode ser o IP

-p (porta)

-t (tópico)

-N (newline) desconsidera o caractere de fim de linha

-u (user) explicita o usuário

-P (Password) explicita a senha do usuário

-c (Desabilita o "clean session". Quando o cliente estiver inscrito em determinado tópico e se desconectar, o cliente receberá todas as mensagens que foram enviadas para aquele tópico, assim que se conectar novamente. É necessário que a publicação da mensagem e a inscrição tenham QoS > 0. Atenção: é obrigatório especificar a identificação, usando a opção abaixo, ao utilizar o -c. E, também, especificar a qualidade do serviço (ver mosquitto_pub) maior que 0 para que a opção surta efeito.

Prompt 3: Publicador (Publisher)

```
C:\Program Files\mosquitto>mosquitto_pub -h localhost -p 1883 -t "teste" -m "Sergio Portari"
```

-h (host) pode ser o IP

-p (porta)

-t (tópico)

-m (mensagem)

-n (mensagem vazia)

-r (record) salva a última mensagem enviada para repassar a novos clientes subscritos

-q (QoS) define o QoS (0 sem confirmação, 1 aguarda confirmação, ou 2 confirmação com garantia de entrega completa)

-u (user) explicita o usuário

-P (Password) explicita a senha do usuário

Mosquitto_passwd – Usado para definir usuários e senhas

C:\Program Files\mosquitto>mosquitto_passwd -c <caminho-path> <usuário> - cria em <caminho-path> o usuário especificado. A senha será solicitada após inserir o comando

C:\Program Files\mosquitto>mosquitto_passwd -D <caminho-path> <usuário> - apaga em <caminho-path> o usuário especificado.

Obs: Se os acentos estiverem errados, no prompt do Windows, digite chcp 65001 para passar a página de código para utf8 (padrão do mosquito)

Para funcionar no UBUNTU foi feito da seguinte forma:

As versões 20.04 em diante não precisa adicionar o repositório manualmente, mas nas antigas precisa adicionar o repositório manualmente. Se estiver usando uma versão anterior a 20.04 execute:

```
wget http://repo.mosquitto.org/debian/mosquitto-repo.gpg.key
sudo apt-key add mosquitto-repo.gpg.key
cd /etc/apt/sources.list.d/
sudo wget http://repo.mosquitto.org/debian/mosquitto-wheezy.list
sudo apt-get update
```

depois execute a instalação com:

```
sudo apt-get install mosquitto -y
sudo apt-get install mosquitto-clients -y
```

Em seguida é necessário abrir a porta a ser utilizada para entrada e saída de mensagens. (Padrão 1883)

```
sudo ufw allow 1883/tcp
sudo ufw allow 1883/udp
```

e reiniciar o firewall:

```
sudo ufw disable
sudo ufw enable
```

Para habilitar acesso externo, é preciso criar usuário e senha e alterar o arquivo mosquitto.conf em /etc/mosquitto. Para isso, vamos fazer:

```
sudo mosquitto_passwd -c /etc/mosquitto/passwd <user_name>
```

onde <user_name> deve ser trocado pelo nome do usuário desejado. A senha será solicitada logo em seguida. Cuidado, pois não aparecerão os caracteres digitados na hora de cadastro da senha.

Em seguida vamos alterar o arquivo mosquitto.conf:

```
sudo nano /etc/mosquitto/mosquitto.conf
```

obs: nano pode ser substituído pelo seu editor de texto preferido (gedit, vi, vim, ...)

No editor, acrescente ao final do arquivo, as seguintes instruções:

```
password_file /etc/mosquitto/passwd
allow_anonymous false
```

```
listener 1883
```

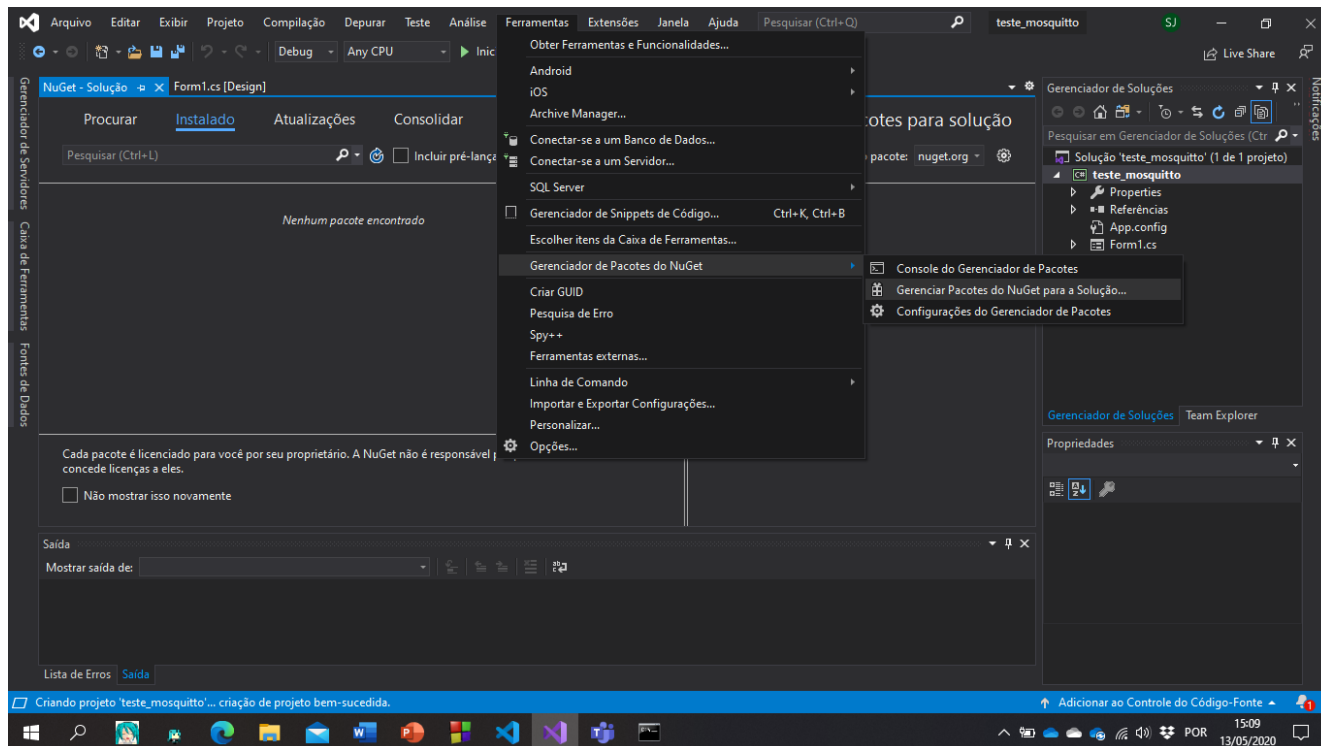
E por fim, deve-se reiniciar o mosquito:

```
sudo service mosquito stop
```

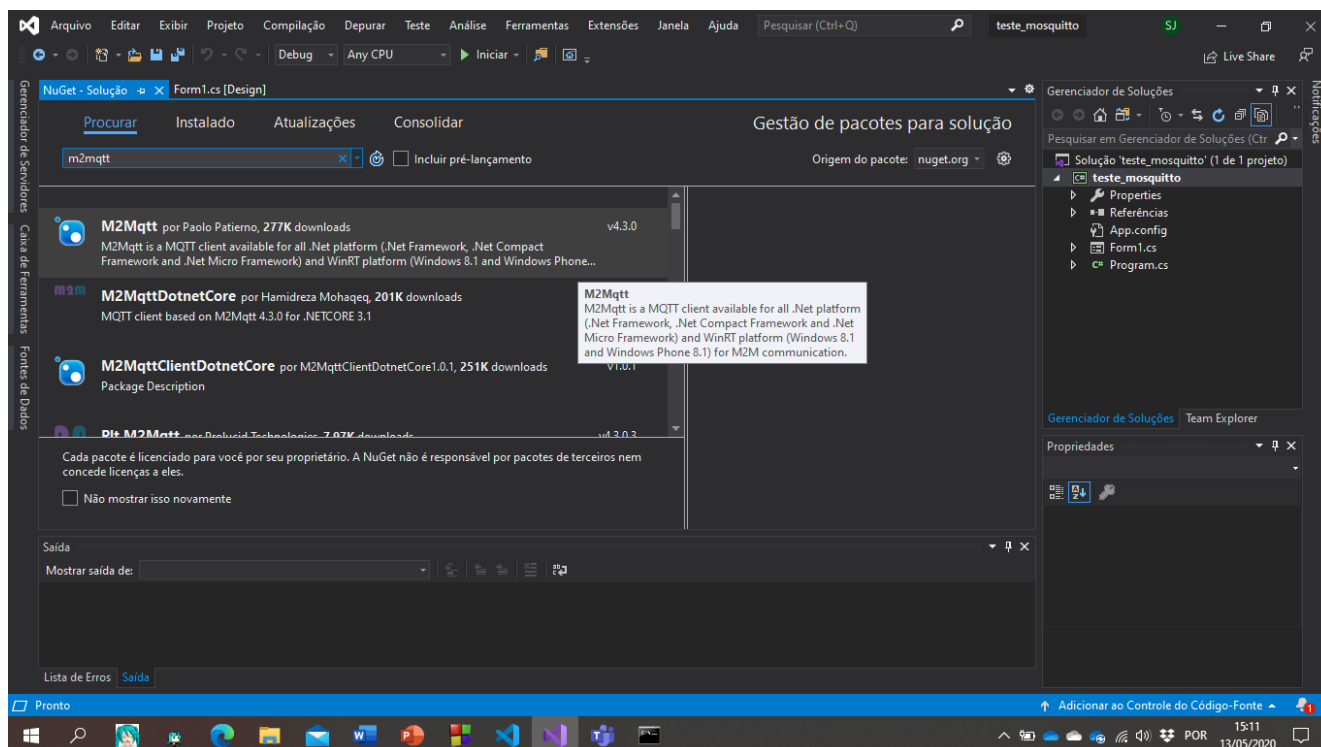
```
sudo service mosquito start
```

Montando uma aplicação prática no C# - exemplo

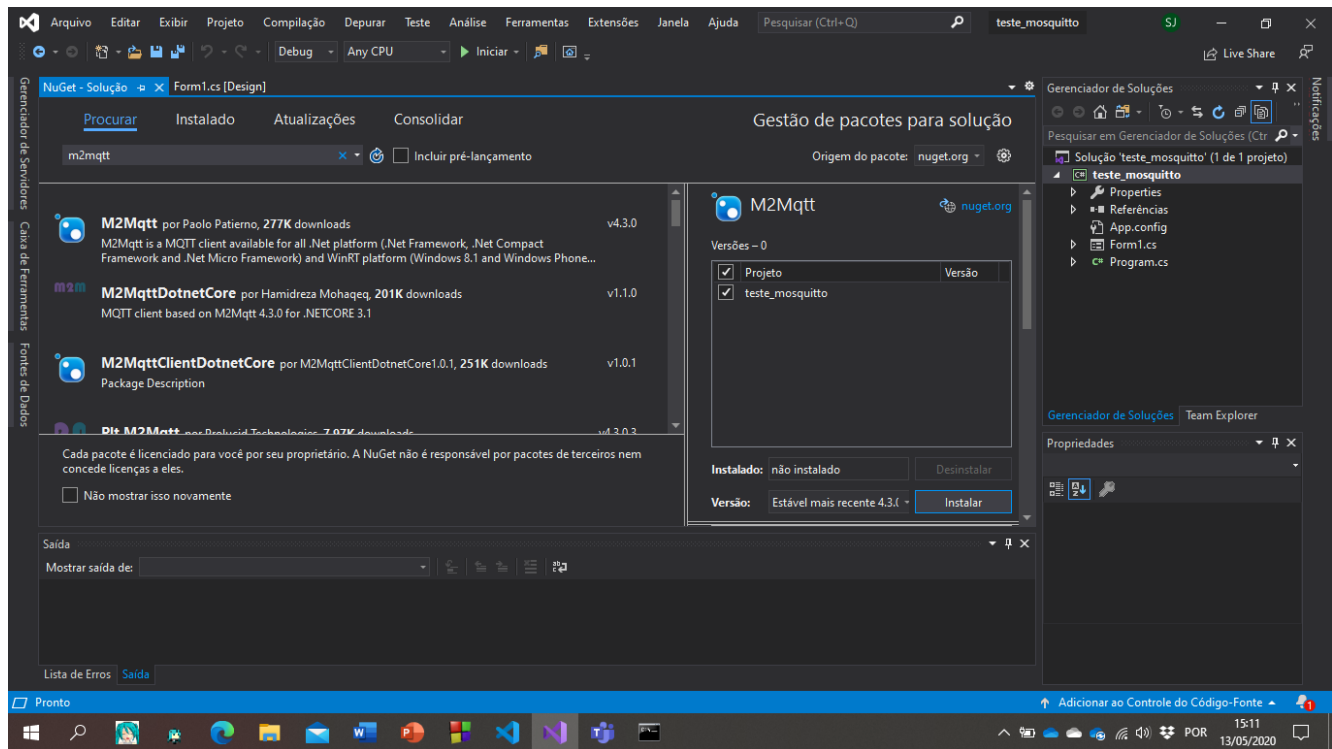
1 - Abra o gerenciador de pacotes NuGet



2 – Em Procurar, busque por M2MQTT



3 – Clique sobre ele, selecione Projeto na janela ao meio e em Instalar



4 – Desenho do form

M2MQTT - Exemplo

Conectar

Subscrever

Mensagem:

Publicar

Recebido:

Desconectar

Sair

5 – adicione uma classe chamada parametros Código completo da classe:

```

namespace teste_mosquitto
{
    class parametros
    {
        public static string recebido;
    }
}

```

6 – Adicione o M2MQTT nas referências (e a rede caso vá utilizar endereços IP)

```

using System;
using System.Text;
using System.Windows.Forms;
using uPLibrary.Networking.M2Mqtt; //adiciona recursos do M2MQTT
using uPLibrary.Networking.M2Mqtt.Messages; //manipulação de mensagens do M2MQTT
using System.Net; //recursos de conversão do IP

namespace mosquitto_teste_2020
{
    public partial class Form1 : Form
    {
        string usuario = "testes";
        string senha = "pepperoni";

        MqttClient cliente = new MqttClient("sergioportari.ddns.net"); //pode usar domínio ou
IP //MqttClient cliente = new MqttClient(IPAddress.Parse("192.168.0.x"));
        public Form1()
        {
            InitializeComponent();
        }

        private void label1_Click(object sender, EventArgs e)
        {

        }

        private void Form1_Load(object sender, EventArgs e)
        {
            // O operador += está adicionando um handler ao evento em tempo de execução.
            // Ou seja, associa um método para ser executado automaticamente quando o evento
ocorrer.
            cliente.MqttMsgPublishReceived += cliente_MqttMsgPublishReceived; //necessário
            cliente.MqttMsgSubscribed += cliente_MqttMsgSubscribed; //opcional 1
            cliente.MqttMsgUnsubscribed += cliente_MqttUnsubscribed; //opcional 2
        }

        void cliente_MqttMsgPublishReceived(object sender, MqttMsgPublishEventArgs e)
        {
            parametros.recebido = Encoding.UTF8.GetString(e.Message);
        }

        private void btn_conectar_Click(object sender, EventArgs e)
        {
            cliente.Connect(Guid.NewGuid().ToString(), usuario, senha); //gera um Id Hexa
aleatório,
            // mas podemos especificar um ID se quisermos
            MessageBox.Show("Conectado ao servidor Mosquitto", "Conectar");
        }
    }
}

```

```

private void btn_desconectar_Click(object sender, EventArgs e)
{
    cliente.Disconnect();
    MessageBox.Show("Desconectado do servidor Mosquitto", "Desconectar");
}

private void btn_subscriver_Click(object sender, EventArgs e)
{
    string[] topico = { "teste/sergio" }; //especifique o topico na string
    byte[] nivelQoS = { MqttMsgBase.QOS_LEVEL_AT_MOST_ONCE }; //QoS 0

    cliente.Subscribe(topico, nivelQoS); //fazer a subscricao
}

private void btn_publicar_Click(object sender, EventArgs e)
{
    cliente.Publish("teste/sergio", Encoding.UTF8.GetBytes(txt_mensagem.Text));
}

private void btn_sair_Click(object sender, EventArgs e)
{
    if (cliente.IsConnected)
        cliente.Disconnect();
    Environment.Exit(0);
}

private void timer1_Tick(object sender, EventArgs e)
{
    if(parametros.recebido != "")
    {
        lbl_recebido.Text = parametros.recebido;
        parametros.recebido = "";
    }
}

void cliente_MqttMsgSubscribed(object sender, MqttMsgSubscribedEventArgs e)
//opcional 1
{
    if (e.MessageId == 1)
        MessageBox.Show("Subscrição feita com sucesso", "Subscriver");
    else
        MessageBox.Show("Erro ao fazer subscrição", "Subscriver");
}

void cliente_MqttUnsubscribed(object sender, MqttMsgUnsubscribedEventArgs e)
//opcional 2
{
    if (e.MessageId == 1)
        MessageBox.Show("Unsubscrição feita com sucesso", "Unsubscriver");
    else
        MessageBox.Show("Erro ao fazer unsubscrição", "Unsubscriver");
}
}

```