

ros2arduino

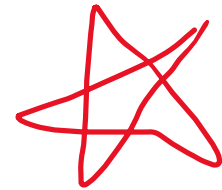
```
#include <ros2arduino.h>

#include <WiFi.h>
#include <WiFiUdp.h>
#define SSID      "xxxxxxxxxx"
#define SSID_PW   "xxxxxxxxxx"
#define AGENT_IP  "192.168.35.71"
#define AGENT_PORT 2018 //AGENT port number

#define PUBLISH_FREQUENCY 1 //hz

void publishString(std_msgs::String* msg, void* arg)
{
    (void)(arg);

    static int cnt = 0;
    sprintf(msg->data, "Hello ros2arduino %d", cnt++);
}
```



8/19/2023

Sangwon Lee

Contents

- ☐ Prerequisites
- ☐ Install Micro-XRCE-DDS-Agent
- ☐ Update ~/.bashrc file for 'ros2' and 'MicroXRCEAgent'
- ☐ AGENT_IP
- ☐ Publisher UDP example
- ☐ Test

Prerequisites

- ☐ Arduino IDE for Ubuntu
- ☐ ESP32 board files
- ☐ ros2arduino library
- ☐ ROS2 dashing

Install Micro-XRCE-DDS-Agent

```
$ git clone https://github.com/eProsima/Micro-XRCE-DDS-Agent.git
$ cd Micro-XRCE-DDS-Agent && git checkout v1.3.0
$ mkdir build && cd build
$ source /opt/ros/dashing/setup.bash # to share libraries with ros2
$ cmake ..
$ make
$ sudo make install
$ sudo ldconfig /usr/local/lib/
```

Update ~/.bashrc

```
# enable programmable completion features (you don't need to enable
# this, if it's already enabled in /etc/bash.bashrc and /etc/profile
# sources /etc/bash.bashrc).
if ! shopt -oq posix; then
  if [ -f /usr/share/bash-completion/bash_completion ]; then
    . /usr/share/bash-completion/bash_completion
  elif [ -f /etc/bash_completion ]; then
    . /etc/bash_completion
  fi
fi

source /opt/ros/humble/setup.bash
export PATH="~/Micro-XRCE-DDS-Agent/build:$PATH"

ros2dev1@ros2dev1-930XBE:~/Micro-XRCE-DDS-Agent/build$
```

```
source /opt/ros/humble/setup.bash
export PATH="~/Micro-XRCE-Agent/build:$Path"
```

AGENT_IP

- ❑ AGENT_IP is the ros2 system.
- ❑ 'ifconfig' shows IP address of PC.

```
#include <WiFi.h>
#include <WiFiUdp.h>
#define SSID "YOUR_SSID"
#define SSID_PW "YOUR_SSID_PASSWORD"
#define AGENT_IP "AGENT_IP_ADDRESS"
#define AGENT_PORT 2018 //AGENT port number

#define PUBLISH_FREQUENCY 2 //hz

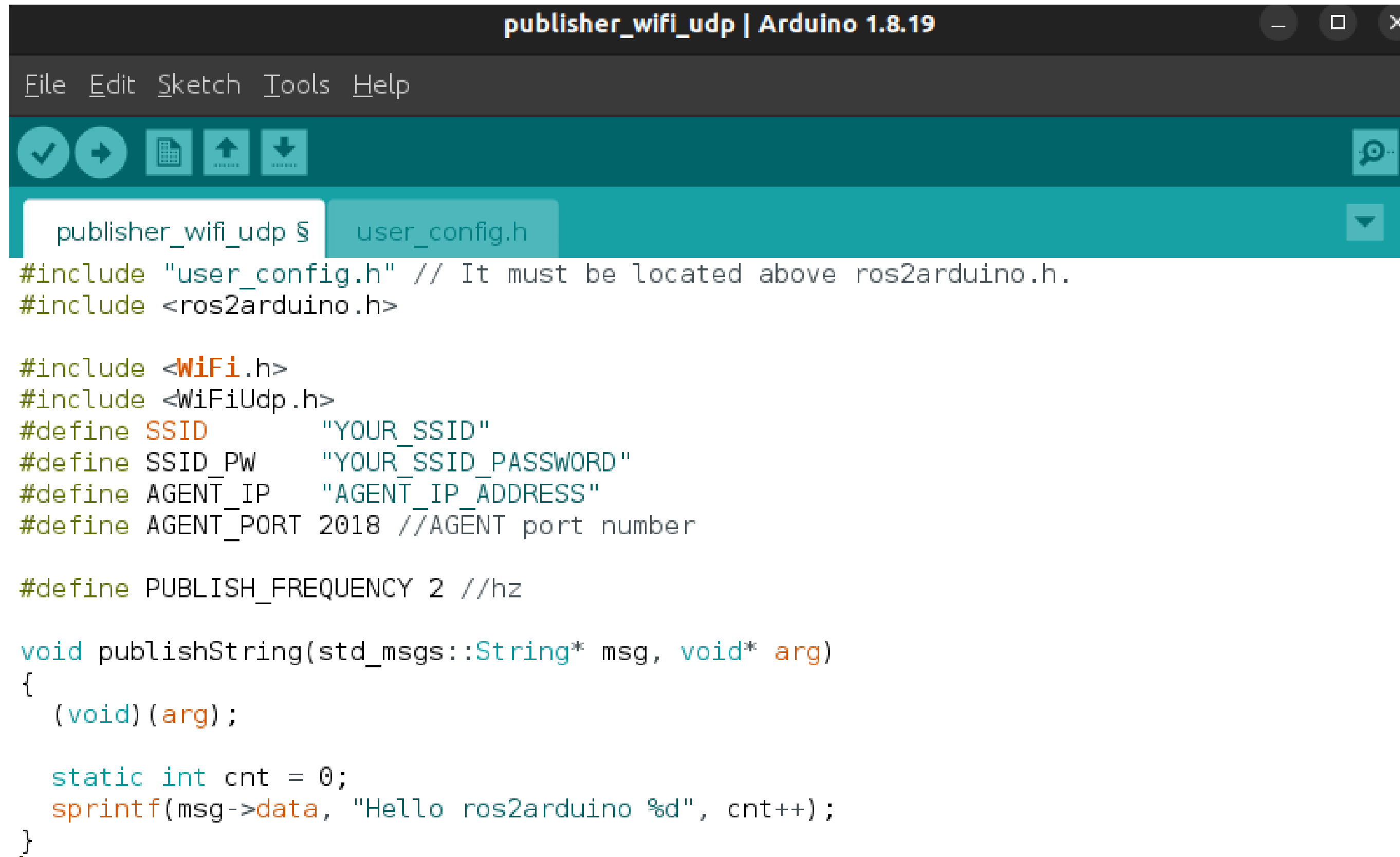
void publishString(std_msgs::String* msg, v
{
    (void)(arg);

    static int cnt = 0;
    sprintf(msg->data, "Hello ros2arduino %d"
}
```

```
ros2dev1@ros2dev1-930XBE:~$ ifconfig
lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 1354 bytes 167222 (167.2 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 1354 bytes 167222 (167.2 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

wlo1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.219.195 netmask 255.255.255.0 broadcast 192.168.219.255
    inet6 2406:5900:1061:9808:2471:bff:9b0f:b32f prefixlen 64 scopeid 0x0<global>
    inet6 fe80::7a32:201f:73f6:7730 prefixlen 64 scopeid 0x20<link>
    inet6 2406:5900:1061:9808:1af5:8ddd:db5d:f7a prefixlen 64 scopeid 0x0<global>
    ether f4:d1:08:5a:f9:5b txqueuelen 1000 (Ethernet)
```

Publisher UDP example



The screenshot shows the Arduino IDE interface with the title bar "publisher_wifi_udp | Arduino 1.8.19". The menu bar includes File, Edit, Sketch, Tools, and Help. The toolbar contains icons for checking, running, uploading, and downloading. The tab bar shows "publisher_wifi_udp" and "user_config.h". The main editor displays the following C++ code:

```
#include "user_config.h" // It must be located above ros2arduino.h.
#include <ros2arduino.h>

#include <WiFi.h>
#include <WiFiUdp.h>
#define SSID "YOUR_SSID"
#define SSID_PW "YOUR_SSID_PASSWORD"
#define AGENT_IP "AGENT_IP_ADDRESS"
#define AGENT_PORT 2018 //AGENT port number

#define PUBLISH_FREQUENCY 2 //hz

void publishString(std_msgs::String* msg, void* arg)
{
    (void)(arg);

    static int cnt = 0;
    sprintf(msg->data, "Hello ros2arduino %d", cnt++);
}
```

Publisher UDP example

```
class StringPub : public ros2::Node
{
public:
    StringPub()
    : Node("ros2arduino_pub_node")
    {
        ros2::Publisher<std_msgs::String>* publisher_ =
            this->createPublisher<std_msgs::String>("arduino_chatter");
        this->createWallFreq(PUBLISH_FREQUENCY,
            (ros2::CallbackFunc)publishString, nullptr, publisher_);
    }
};

WiFiUDP udp;

void setup()
{
    WiFi.begin(SSID, SSID_PW);
    while(WiFi.status() != WL_CONNECTED);

    ros2::init(&udp, AGENT_IP, AGENT_PORT);
}
```


Test

\$ MicroXRCEAgent udp6 -p 2018 -v

```
ros2dev1@ros2dev1-930XBE: ~  
ros2dev1@ros2dev1-930XBE: ~  
ros2dev1@ros2dev1-930XBE:~$ MicroXRCEAgent udp6 -p 2018 -v 5  
Press CTRL+C to exit  
[1694800727.475213] info      | UDPv6AgentLinux.cpp | init          | running...  
[1694800727.475960] info      | Root.cpp            | set_verbose_level | logger setup  
[1694800738.637266] debug     | UDPv6AgentLinux.cpp | recv_message    | [==>> UDP <==]  
[1694800738.637390] debug     | UDPv6AgentLinux.cpp | recv_message    | [==>> UDP <==]  
[1694800738.637612] info      | Root.cpp            | create_client   | create  
0x81  
[1694800738.637729] info      | SessionManager.hpp  | establish_session | session established  
000:0000:0000:0000:0000:ffff:c0a8:db6c:b822  
[1694800738.637835] info      | SessionManager.hpp  | establish_session | session re-established  
000:0000:0000:0000:0000:ffff:c0a8:db6c:b822  
[1694800738.637959] debug     | UDPv6AgentLinux.cpp | send_message    | [** <<UDP>> **]  
[1694800738.638000] debug     | UDPv6AgentLinux.cpp | send_message    | [** <<UDP>> **]
```

Test

```
ros2dev1@ros2dev1-930XBE: ~  
ros2dev1@ros2dev1-930XBE: ~  
ros2dev1@ros2dev1-930XBE:~$ ros2 node list -a  
/_ros2cli_daemon_0_413217034ad94c2f9176393624df2609  
ros2dev1@ros2dev1-930XBE:~$ ros2 topic list  
/arduino_chatter  
/parameter_events  
/rosout  
ros2dev1@ros2dev1-930XBE:~$ ros2 topic echo /arduino_chatter  
data: Hello ros2arduino 37  
---  
data: Hello ros2arduino 0  
---  
data: Hello ros2arduino 1  
---  
data: Hello ros2arduino 2  
---
```

Thank you