UNDERSTANDING ROS2(FOXY)

SER515-Spring2022-TEAM 8

ROS 2 Documentation

The Robot Operating System (ROS) framework is a set of software libraries and tools for building robot applications. From drivers to state-of-the-art algorithms, and with powerful developer tools, ROS has what you need for your next robotics project. And it's all open source.

Since ROS was started in 2007, a lot has changed in the robotics and ROS community. The goal of the ROS 2 project is to adapt to these changes like high level robotic systems which grew in application spaces such as mobile outdoor robotics, drone swarms and even self-driving cars. leveraging what is great about ROS 1 and improving what isn't.

Here you will find the official documentation on **ROS 2**, the newest version of ROS.

So, we start with installing ROS 2(foxy) – we are using foxy not Galactic which is a newer version of ROS2 because it has many bugs that need to be fixed and installation is not done properly in windows, SO we have concluded to work with ROS 2 Foxy after our research.

ROS 2 Installation

Let's begin by installing **chocolatey**.

Chocolatey is a package manager for Windows, install it by following their installation instructions:

https://chocolatey.org/

You'll use Chocolatey to install some other developer tools.

Now use Chocolatey to install **Python** (3.10 64bit). Open command prompt and type

```
choco install -y python --version 3.10.2
```

ROS2 expect the path of python to be at **c:/python310** double check that it's installed at right path.

Installing **Visual C++ Redistributable**. Open command prompt and type

```
choco install -y vcredist2013 vcredist140
```

Now download/install **Win64 OpenSSL v1.1.1L** from this page

https://slproweb.com/products/Win32OpenSSL.html

Scroll down to the bottom of the page and download Win64 only not Win32 or Light versions.

Now set the environment variables

```
setx -m OPENSSL_CONF "C:\Program Files\OpenSSL-Win64\bin\openssl.cfg"
```

You will now need to append the OpenSSL-Win64 bin folder to your PATH. You can do this by clicking the Windows icon, typing “Environment Variables”, then clicking on “Edit the system environment variables”. In the resulting dialog, click “Environment Variables”, then click “Path” on the bottom pane, finally click “Edit” and add the path below.

- `C:\Program Files\OpenSSL-Win64\bin\`

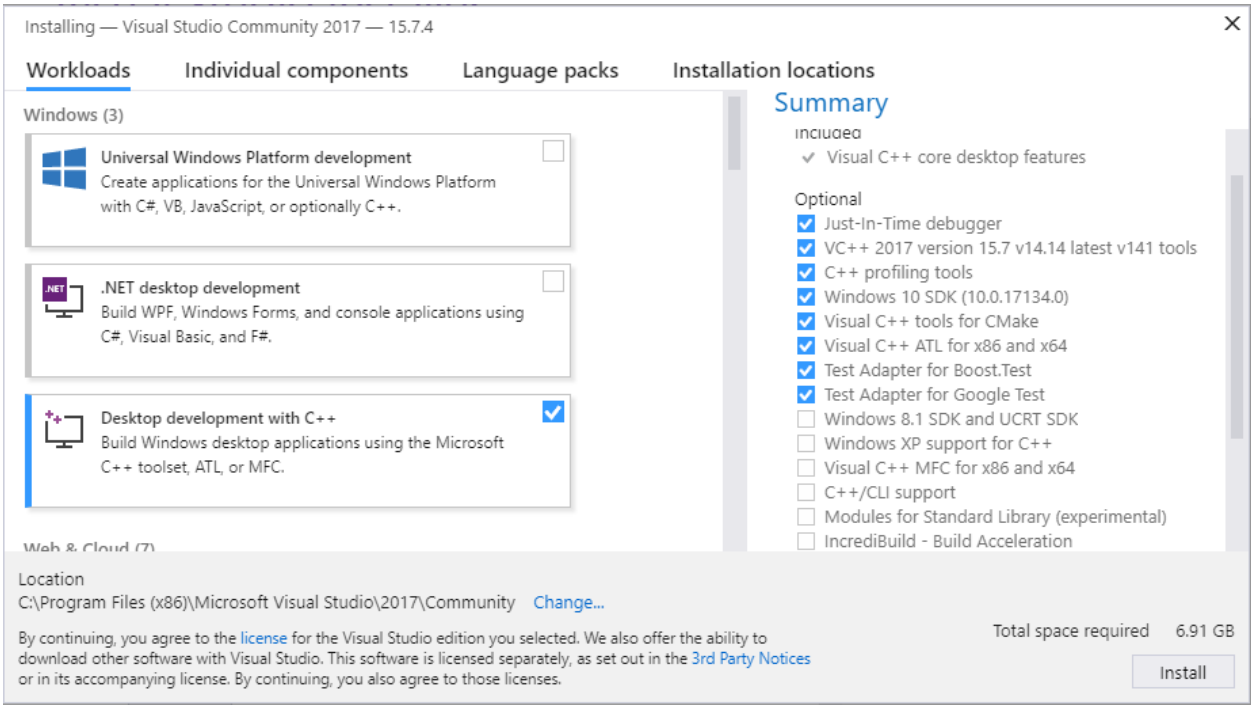
Install **Visual Studio 2019**.

If you already have a paid version of Visual Studio 2019 (Professional, Enterprise), skip this step.

Microsoft provides a free of charge version of Visual Studio 2019, named Community, which can be used to build applications that use ROS 2. You can download the installer directly through this link.

Make sure that the Visual C++ features are installed.

An easy way to make sure they’re installed is to select the `Desktop development with C++` workflow during the install.



Installing — Visual Studio Community 2017 — 15.7.4

Windows (3)

□ Universal Windows Platform development
Create applications for the Universal Windows Platform with C#, VB, JavaScript, or optionally C++.

□ .NET desktop development
Build WPF, Windows Forms, and console applications using C#, Visual Basic, and F#.

☑ Desktop development with C++
Build Windows desktop applications using the Microsoft C++ toolset, ATL, or MFC.

Web & Cloud (7)

Summary

Included
✓ Visual C++ core desktop features

Optional
☑ Just-In-Time debugger
☑ VC++ 2017 version 15.7 v14.14 latest v141 tools
☑ C++ profiling tools
☑ Windows 10 SDK (10.0.17134.0)
☑ Visual C++ tools for CMake
☑ Visual C++ ATL for x86 and x64
☑ Test Adapter for Boost.Test
☑ Test Adapter for Google Test
□ Windows 8.1 SDK and UCRT SDK
□ Windows XP support for C++
□ Visual C++ MFC for x86 and x64
□ C++/CLI support
□ Modules for Standard Library (experimental)
□ IncrediBuild - Build Acceleration

Location
C:\Program Files (x86)\Microsoft Visual Studio\2017\Community Change...

By continuing, you agree to the license for the Visual Studio edition you selected. We also offer the ability to download other software with Visual Studio. This software is licensed separately, as set out in the 3rd Party Notices or in its accompanying license. By continuing, you also agree to those licenses.

Total space required 6.91 GB

Make sure that no C++ CMake tools are installed by unselecting them in the list of components to be installed.

Now let's install OpenCV which can be used during image and video processing from the rover. You can download a precompiled version of OpenCV 3.4.6 from the below link

https://github.com/ros2/ros2/releases/download/opencv-archives/opencv-3.4.6-vc16.VS2019.zip

Now unpack the zip file in to **c:/OpenCV**, then open command prompt as an admin and run this command

```
setx -m OpenCV_DIR C:\opencv
```

Now add the path C:\OpenCV\x64\vc16\bin into the Path similarly as we have added OpenSSL.

There are a few dependencies not available in the Chocolatey package database. To ease the manual installation process, we provide the necessary Chocolatey packages.

As some chocolatey packages rely on it, we start by installing CMake

```
Choco install -y cmake
```

You will need to append the CMake bin folder C:\Program Files\CMake\bin to your PATH.

Please download these packages from this GitHub repository.

- asio.1.12.1.nupkg
- bullet.2.89.0.nupkg
- cunit.2.1.3.nupkg
- eigen-3.3.4.nupkg
- tinyxml-usestl.2.6.2.nupkg
- tinyxml2.6.0.0.nupkg
- log4cxx.0.10.0.nupkg

Once these packages are downloaded, open an administrative shell, and execute the following command:

```
choco install -y -s <PATH\TO\DOWNLOADS> asio cunit eigen tinyxml-usestl tinyxml2 log4cxx bullet
```

Please replace `<PATH\TO\DOWNLOADS>` with the folder you downloaded the packages to.

You must also install some python dependencies for command-line tools:

```
python -m pip install -U catkin_pkg cryptography empy ifcfg lark-parser lxml netifaces numpy opencv-python pyparsing pyyaml setuptools rosdistro
```

RQt dependencies installation

```
python -m pip install -U pydot PyQt5
```

To run rqt_graph, you’ll need Graphviz.

```
choco install graphviz
```

You will need to append the Graphviz bin folder C:\Program Files\Graphviz\bin to your PATH, by navigating to “Edit the system environment variables” as described above.

Downloading ROS 2

We install ROS2 foxy through VS Command prompt:

1. From the start menu, look for x64 Native Tools Command Prompt for VS 2019.
2. Open the command prompt as administrator.
3. Run the following to install ROS 2 Foxy.

```
mkdir c:\opt\chocolatey

set ChocolateyInstall=c:\opt\chocolatey

choco source add -n=ros-win -s="https://aka.ms/ros/public" --priority=1

choco upgrade ros-foxy-desktop -y --execution-timeout=0 –pre
```

Now to build ROS2 we would need a Visual Studio Command Prompt (“x64 Native Tools Command Prompt for VS 2019”) running as Administrator and now run the following commands

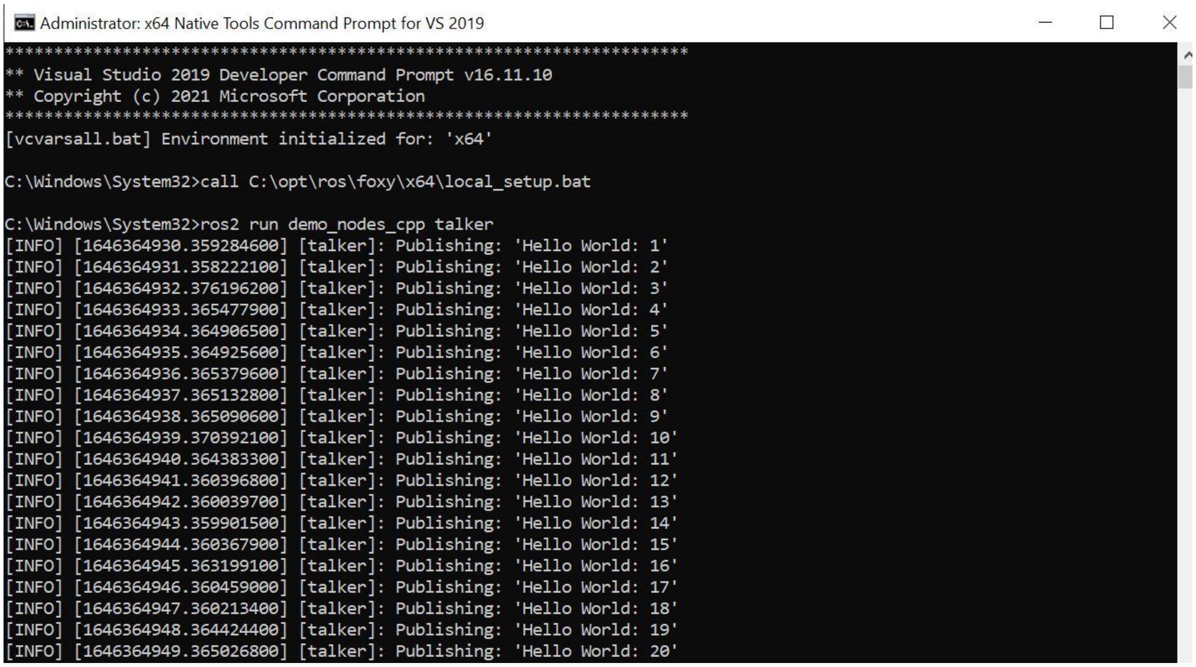
```
//activate the ROS 2 environment

c:\opt\ros\foxy\x64\setup.bat


//activate the Gazebo simulation environment

c:\opt\ros\foxy\x64\share\gazebo\setup.bat

set "SDF_PATH=c:\opt\ros\foxy\x64\share\sdformat\1.6"
```

Now you are in the ROS 2 Developer command prompt. Let's test its working through running below Example:

To run the examples, first open a clean new Visual Studio Command Prompt ("x64 Native Tools Command Prompt for VS 2019") running as Administrator and set up the workspace by sourcing the local_setup.bat file. Then, run a C++ talker:

call C:\opt\ros\foxy\x64\local_setup.bat

ros2 run demo_nodes_cpp talker

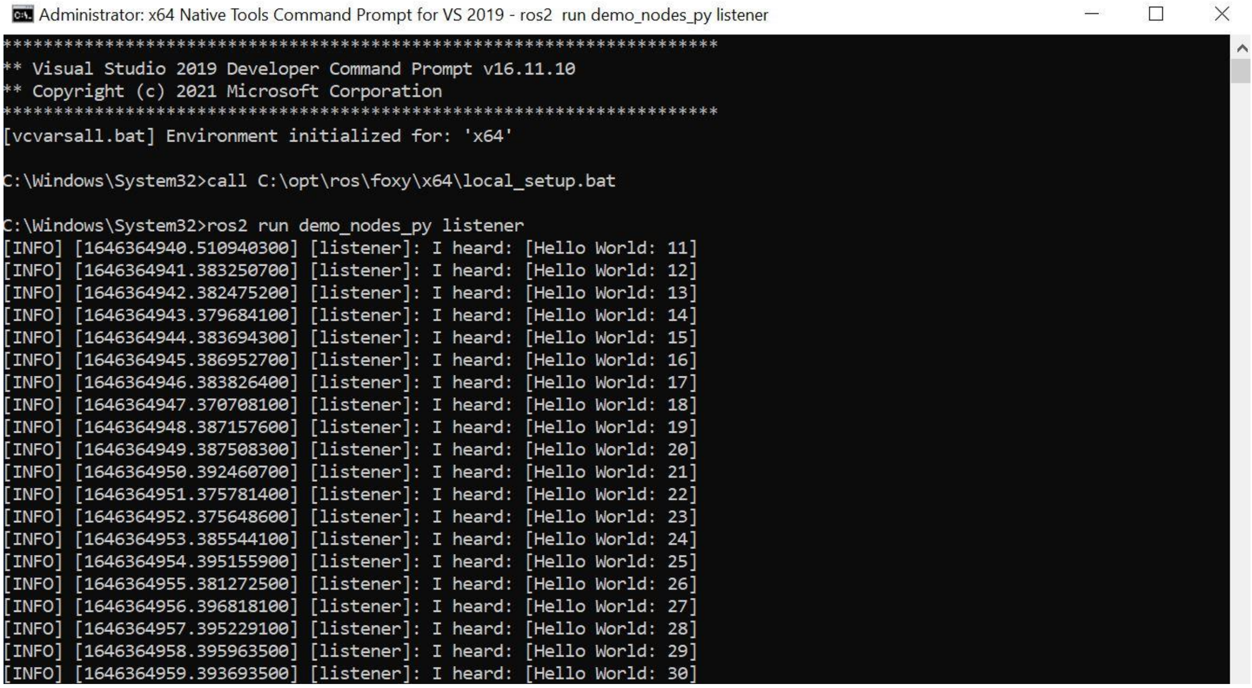


```
**********************************************************************
** Visual Studio 2019 Developer Command Prompt v16.11.10
** Copyright (c) 2021 Microsoft Corporation
**********************************************************************
[vcvarsall.bat] Environment initialized for: 'x64'

C:\Windows\System32>call C:\opt\ros\foxy\x64\local_setup.bat

C:\Windows\System32>ros2 run demo_nodes_cpp talker
[INFO] [1646364930.359284600] [talker]: Publishing: 'Hello World: 1'
[INFO] [1646364931.358222100] [talker]: Publishing: 'Hello World: 2'
[INFO] [1646364932.376196200] [talker]: Publishing: 'Hello World: 3'
[INFO] [1646364933.365477900] [talker]: Publishing: 'Hello World: 4'
[INFO] [1646364934.364906500] [talker]: Publishing: 'Hello World: 5'
[INFO] [1646364935.364925600] [talker]: Publishing: 'Hello World: 6'
[INFO] [1646364936.365379600] [talker]: Publishing: 'Hello World: 7'
[INFO] [1646364937.365132800] [talker]: Publishing: 'Hello World: 8'
[INFO] [1646364938.365090600] [talker]: Publishing: 'Hello World: 9'
[INFO] [1646364939.370392100] [talker]: Publishing: 'Hello World: 10'
[INFO] [1646364940.364383300] [talker]: Publishing: 'Hello World: 11'
[INFO] [1646364941.360396800] [talker]: Publishing: 'Hello World: 12'
[INFO] [1646364942.360039700] [talker]: Publishing: 'Hello World: 13'
[INFO] [1646364943.359901500] [talker]: Publishing: 'Hello World: 14'
[INFO] [1646364944.360367900] [talker]: Publishing: 'Hello World: 15'
[INFO] [1646364945.363199100] [talker]: Publishing: 'Hello World: 16'
[INFO] [1646364946.360459000] [talker]: Publishing: 'Hello World: 17'
[INFO] [1646364947.360213400] [talker]: Publishing: 'Hello World: 18'
[INFO] [1646364948.364424400] [talker]: Publishing: 'Hello World: 19'
[INFO] [1646364949.365026800] [talker]: Publishing: 'Hello World: 20'
```

In a separate Visual Studio Command Prompt (“x64 Native Tools Command Prompt for VS 2019”) running as Administrator you can do the same, but instead run a Python listener:

call C:\opt\ros\foxy\x64\local_setup.bat

ros2 run demo_nodes_py listener


Administrator: x64 Native Tools Command Prompt for VS 2019 - ros2 run demo_nodes_py listener

```
**********************************************************************
** Visual Studio 2019 Developer Command Prompt v16.11.10
** Copyright (c) 2021 Microsoft Corporation
**********************************************************************
[vcvarsall.bat] Environment initialized for: 'x64'

C:\Windows\System32>call C:\opt\ros\foxy\x64\local_setup.bat

C:\Windows\System32>ros2 run demo_nodes_py listener
[INFO] [1646364940.510940300] [listener]: I heard: [Hello World: 11]
[INFO] [1646364941.383250700] [listener]: I heard: [Hello World: 12]
[INFO] [1646364942.382475200] [listener]: I heard: [Hello World: 13]
[INFO] [1646364943.379684100] [listener]: I heard: [Hello World: 14]
[INFO] [1646364944.383694300] [listener]: I heard: [Hello World: 15]
[INFO] [1646364945.386952700] [listener]: I heard: [Hello World: 16]
[INFO] [1646364946.383826400] [listener]: I heard: [Hello World: 17]
[INFO] [1646364947.370708100] [listener]: I heard: [Hello World: 18]
[INFO] [1646364948.387157600] [listener]: I heard: [Hello World: 19]
[INFO] [1646364949.387508300] [listener]: I heard: [Hello World: 20]
[INFO] [1646364950.392460700] [listener]: I heard: [Hello World: 21]
[INFO] [1646364951.375781400] [listener]: I heard: [Hello World: 22]
[INFO] [1646364952.375648600] [listener]: I heard: [Hello World: 23]
[INFO] [1646364953.385544100] [listener]: I heard: [Hello World: 24]
[INFO] [1646364954.395155900] [listener]: I heard: [Hello World: 25]
[INFO] [1646364955.381272500] [listener]: I heard: [Hello World: 26]
[INFO] [1646364956.396818100] [listener]: I heard: [Hello World: 27]
[INFO] [1646364957.395229100] [listener]: I heard: [Hello World: 28]
[INFO] [1646364958.395963500] [listener]: I heard: [Hello World: 29]
[INFO] [1646364959.393693500] [listener]: I heard: [Hello World: 30]
```

You should see the talker saying that it’s Publishing messages and the listener saying I heard those messages. This verifies both the C++ and Python APIs are working properly. Hooray!