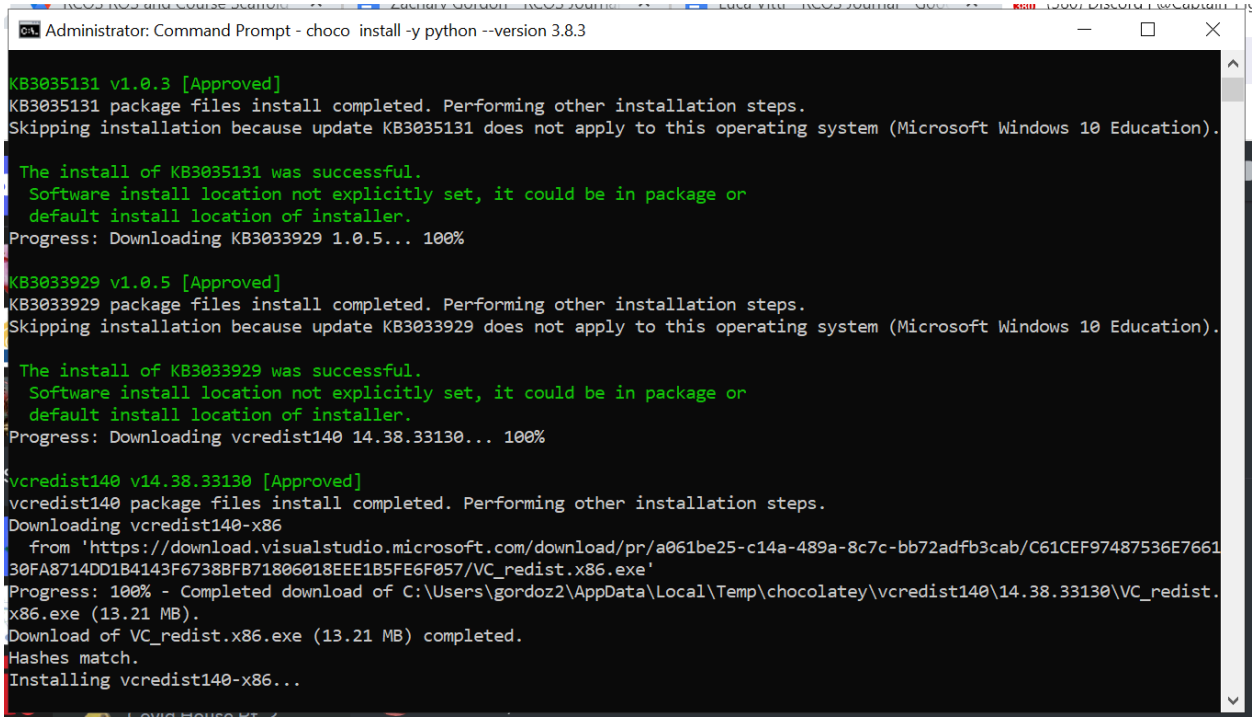
Zachary Gordon - RCOS S24 Journal

Date: 1/30/24

Today I followed Luca's YT guide for the ROS install. I was told nothing about the process, and I have never had ROS on my computer before, so I am a good test subject for the guide.

At the Python 3.8.3 installation step, my computer got hung up on the line "Installing vcredist140-x86". I let it sit for about 10 minutes, but it was unmoving. At that point I hit Ctrl+C, figuring I'd restart the step. Instead, it actually forced the process to move forward, and Python 3.8.3 installed successfully from there.



One difference between my computer and the guide is that my terminal told me I should reboot after the Chocolatey/Python 3.8.3 install. This may have had to do with the vcredist140 which got stuck. I did not reboot, and moved forward to attempt to follow the guide as closely as possible. The next step was a manual install of the C++ redistributables. Since vcredist140 had been installed automatically in the Python step, it did not get installed again, and a message appeared saying it was already installed.

During my installation, my computer sometimes did not allow copying of commands out of the Jupiter Notebook and into my Command Prompt window. I'm unsure if this was an issue specific to me.

Another question came up at the Visual Studio Installation Step. I already had an installation of VS 2022 on my laptop from years ago, and I'm sure it included some of the installed tools we didn't want. I attempted the download according to the instructions, and I found a difference. My download was only 5.15 GB, despite the fact I had everything checked as in the video. We probably should include some direct guidance for those who might already have a VS installation on their PC. Maybe make clear also that the version is VS Community 2019.

I ran into one other issue right at the end. In the video, a non-admin command prompt is opened for the listener while the admin (system32) command prompt is used for the caller. Doing this will mean that the listener cannot respond to the talker. It's a poor showcase and probably should be updated in the video. I tested using admin (system32) prompts for both the talker and listener and non-admin prompts for both the talker and listener. In each of these cases, ros2 responded as it should have.

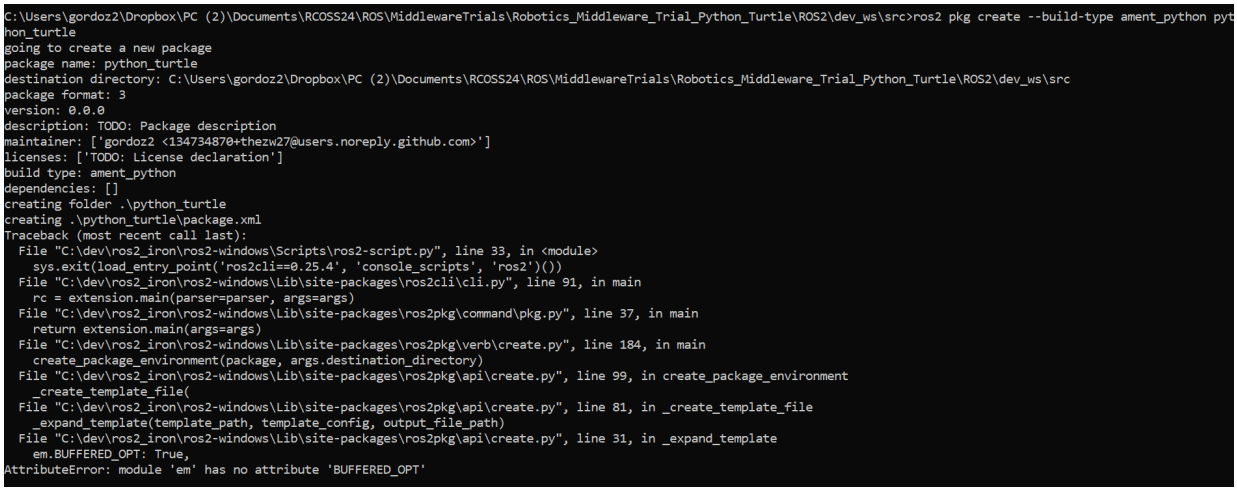
OVERALL REVIEW:
Using the video, I was able to install ros2 on Windows successfully. It was mostly very helpful, with good narration and showing every single step. Besides the few issues outlined in the notes above, my other comments have to do with small annoyances. The repeated editing of the Environment variables might be able to be streamlined into a single step. Constantly going in and out got old and took up a lot of time. The other thing to watch out for are some small inaccuracies in the video. Examples include the wrong pip command at 18:58, the wrong (non-universal) file size at 11:47, and the mentioned Command Prompt mismatch at 31:24. That last one especially needs to be updated, as the video ends with an instruction that will never work. We also might want to add a time estimate about the install - mine took nearly two hours.

Date: 02/02/24

Following Instructions from
https://github.com/robotraconteur-contrib/Robotics_Middleware_Trial_Python_Turtle/blob/master/ROS2/Trial_instruction.md

1. Clone/pull the repository, you will need it to follow these steps.
    a. Use the repo URL https://github.com/robotraconteur-contrib/Robotics_Middleware_Trial_Python_Turtle.git
    b. Use either GitHub desktop or the command line to clone the repo to a local directory
2. Find and copy the absolute path to your local repository
3. Open x64 VS Command prompt as administrator

    a. Type “x64” in the Windows search Bar. “x64 Native Tools Command Prompt for VS 2019” should come up. Right click it, and select Run as Administrator.
4. Enter “cd PATH_TO_REPO\Robotics_Middleware_Trial_Python_Turtle>cd ROS2\dev_ws\src”
5. Assuming you installed ROS2 according to Luca’s directions, enter “call C:\dev\ros2_iron\ros2-windows\local_setup.bat”. Otherwise, enter “call C:\YOUR_PATH\ros2-windows\local_setup.bat” You may see the warning: "[rti_connext_dds_cmake_module][warning] RTI Connext DDS environment script not found (\resource\scripts\rtisetenv_x64Win64VS2017.bat). RTI Connext DDS will not be available at runtime, unless you already configured PATH manually." IGNORE IT
6. Run the command “pip install empy==3.3.4”
7. Type “ros2 pkg create --build-type ament_python python_turtle”. This creates a new directory in your src folder called python_turtle. You should see the following information:



8. Enter “ros2 pkg create --build-type ament_cmake turtle_interfaces”. This creates a new directory in your src folder called turtle_interfaces. You should see output

like the following:

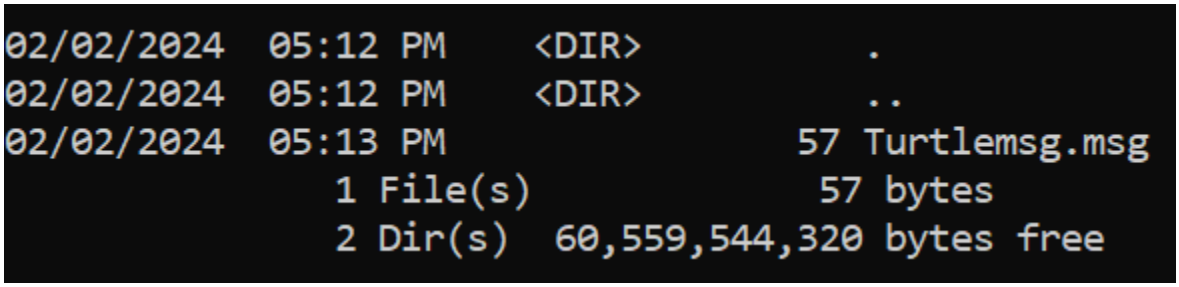
```
C:\Users\gordoz2\Dropbox\PC (2)\Documents\RCOSS24\ROS\MiddlewareTrials\Robotics_Middleware_Trial_Python_Turtle\ROS2\dev_ws\src>ros2 pkg create --build-type ament_cmake turt
le_interfaces
going to create a new package
package name: turtle_interfaces
destination directory: C:\Users\gordoz2\Dropbox\PC (2)\Documents\RCOSS24\ROS\MiddlewareTrials\Robotics_Middleware_Trial_Python_Turtle\ROS2\dev_ws\src
package format: 3
version: 0.0.0
description: TODO: Package description
maintainer: ['gordoz2 <134734870+thezw27@users.noreply.github.com>']
licenses: ['TODO: License declaration']
build type: ament_cmake
dependencies: []
creating folder .\turtle_interfaces
creating .\turtle_interfaces\package.xml
Traceback (most recent call last):
  File "C:\dev\ros2_iron\ros2-windows\Scripts\ros2-script.py", line 33, in <module>
    sys.exit(load_entry_point('ros2cli==0.25.4', 'console_scripts', 'ros2')())
  File "C:\dev\ros2_iron\ros2-windows\Lib\site-packages\ros2cli\cli.py", line 91, in main
    rc = extension.main(parser=parser, args=args)
  File "C:\dev\ros2_iron\ros2-windows\Lib\site-packages\ros2pkg\command\pkg.py", line 37, in main
    return extension.main(args=args)
  File "C:\dev\ros2_iron\ros2-windows\Lib\site-packages\ros2pkg\verb\create.py", line 184, in main
    create_package_environment(package, args.destination_directory)
  File "C:\dev\ros2_iron\ros2-windows\Lib\site-packages\ros2pkg\api\create.py", line 99, in create_package_environment
    _create_template_file(
  File "C:\dev\ros2_iron\ros2-windows\Lib\site-packages\ros2pkg\api\create.py", line 81, in _create_template_file
    _expand_template(template_path, template_config, output_file_path)
  File "C:\dev\ros2_iron\ros2-windows\Lib\site-packages\ros2pkg\api\create.py", line 31, in _expand_template
    em.BUFFERED_OPT: True,
AttributeError: module 'em' has no attribute 'BUFFERED_OPT'
```

9. Create a new directory named msg within your turtle_interfaces directory called “msg”. This can be accomplished from File Explorer or with the following commands (assuming you begin in your PATH_TO_REPO\Robotics_Middleware_Trial_Python_Turtle>cd ROS2\dev_ws\src directory)
    a. “cd turtle_interfaces”
    b. “mkdir msg”
10. Check whether the folder has been created by entering the command “dir”. You should see this output:

```
02/02/2024  05:01 PM    <DIR>          .
02/02/2024  05:01 PM    <DIR>          ..
02/02/2024  05:12 PM    <DIR>          msg
               0 File(s)              0 bytes
               3 Dir(s)  60,557,242,368 bytes free
```

11. Now, create a new file named “Turtlemsg.msg” inside of the new “msg” directory. To do this:
    a. Enter “cd msg”
    b. Enter “nul > Turtlemsg.msg”

12. Check whether the file has been created by entering the command “dir”. You should see this output (though your file should show 0 bytes):



13. Open Turtlemsg.msg in Visual Studio or some other editor. (To do this, you may have to right-click the file in File Explorer and change its default opening application). Enter this data into the file, then save and close:

```
string name
geometry_msgs/Pose turtle_pose
string color
```

Date: 02/06/24

Continued translating/following Instructions from https://github.com/robotraconteur-contrib/Robotics_Middleware_Trial_Python_Turtle/blob/master/ROS2/Trial_instruction.md

NOTE: STEPS CONTINUED FROM ABOVE

14. Open the file “CMakeLists.txt” in the turtle_interfaces directory. Paste the following code into the file before the line “ament_package()” Do not delete any existing code. Be sure to save after editing the file.

```
find_package(geometry_msgs REQUIRED)
find_package(rosidl_default_generators REQUIRED)

rosidl_generate_interfaces(${PROJECT_NAME}
  "msg/Turtlemsg.msg"
  DEPENDENCIES geometry_msgs
)
```

15. Open the file “package.xml” under the turtle_interfaces directory. Paste the following code into the file between the line “</export>” and the line “</package>” Do not delete any existing code. Be sure to save after editing the file.

```
<build_depend>rosidl_default_generators</build_depend>
<depend>geometry_msgs</depend>

<exec_depend>rosidl_default_runtime</exec_depend>

<member_of_group>rosidl_interface_packages</member_of_group>
```

16. At this point, the Turtlemsg type should be built when building the package. These can be included like other ROS messages.

```
from python_turtle import Turtlemsg
from geometry_msgs import Pose
```

Turtlemsg objects are created like this:

```
turtle_msg=Turtlemsg()
turtle_msg.name="myturtle"
turtle_msg.turtle_pose=Pose()
turtle_msg.color="red"
```

17. Create a new folder in the turtle_interfaces directory called “srv”. This can be accomplished in the terminal with the following commands (assuming you begin in your PATH_TO_REPO\Robotics_Middleware_Trial_Python_Turtle>cd ROS2\dev_ws\src directory):
    a. “cd turtle_interfaces”
    b. “mkdir srv”
18. Now, navigate into the “srv” directory and create two new files, “Setpose.srv” and “Setcolor.srv”
    a. cd turtle_interfaces\srv
    b. nul > Setpose.srv
    c. nul > Setcolor.srv
19. In the file “Setpose.srv”, paste the following code:

```
geometry_msgs/PoseStamped turtle_pose
---
int8 ret
```

20. In the file “Setcolor.srv”, paste the following code:
    string color
    ---
    int8 ret

21. Open CMakeLists.txt in the turtle_interfaces directory. Add the following two lines between ”msg/Turtlemsg.msg” and “DEPENDENCIES geometry_msgs”. Be sure to save after editing the file.
    "srv/Setpose.srv"
    "srv/Setcolor.srv"

22. Assuming the same x64 terminal is still open and the ROS2 environment is set up (“call C:\YOUR_PATH\ros2-windows\local_setup.bat” in the ROS2 dev_ws directory if not), we are now ready to build the workspace and packages. Enter the following code:

    colcon build --merge-install

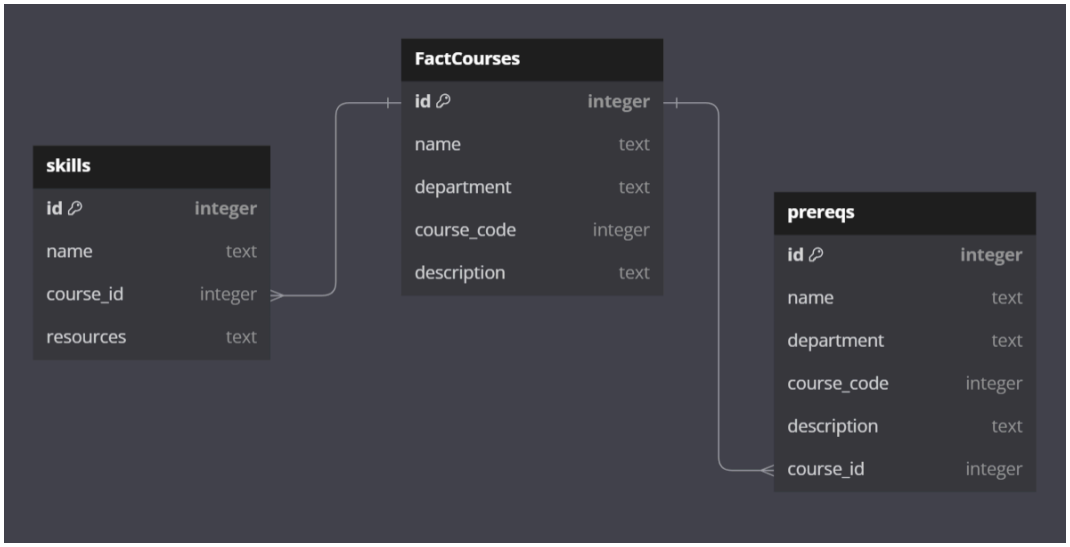
Date: 02/09/24

Debugged colcon issues

1. Need to do pip install -U colcon-common-extensions before step 22
2. Need to do pip install setuptools==58.2.0 before step 22

Date: 02/13/24

Reviewing old work from the Course Scaffolding Project. Took a look at previous Database notes and used leftover course information to generate a new proposed structure and basic SQL file. Intention is to talk more on Friday about it with the group



NOTE: This would hold repeat data for some pre-reqs/courses, but SQL is made for massive data management.

Date: 02/16/24

Spent the day attempting to debug the CMake issue. Looked into the path length issue, but I don't think that's actually the root problem. Also discussed Database with Dr. Wen.

Date: 02/23/24

Spent the day attempting to debug the CMake issue. Looked into the path length issue, but I don't think that's actually the root problem. Also discussed Database with Dr. Wen.

Date: 02/27/24

After speaking with Dr. Wen on Friday, redirecting focus to Ubuntu ROS2 installation. Today, I'm starting from scratch following to Ubuntu install directions and progressing through Checkpoint 1 as Luca has. Hope to be fully caught up with him by Friday so we can begin forward progress on Checkpoint 2 together.

Checkpoint 1 Steps:

1. Find the Robotics_Middleware…/ROS2/dev_ws/src directory
2. Create new directories for python_turtle and turtle_interfaces by entering:
    a. source /opt/ros/iron/setup.bash
    b. ros2 pkg create --build-type ament_python python_turtle
    c. ros2 pkg create --build-type ament_cmake turtle_interfaces
3. Cd into the turtle_interfaces directory and enter command "mkdir msg"
4. Cd into the new msg directory and create a new file name "Turtlemsg.msg". To do so, enter the command "vi Turtlemsg.msg" (if you have vim installed)
5. This should open the vim editor for the new file. Press the I key to switch to insert mode, the enter these contents into the file:

    string name
    geometry_msgs/Pose turtle_pose
    string color

6. Now, hit the Esc key to exit insert mode. Next, enter ":wq" to save the file and close it.

Worked until colcon build, new CMake Error

Date: 03/01/24

Finished writing up Checkpoint 1 completely in Ubuntu. Currently attempting to fix an error involving the CMake file for turtle_interfaces. There's an extra parenthesis somewhere that is blocking the makefile.

7. Cd into the turtle_interfaces folder - Open the CMakeLists.txt file (command code CMakeLists.txt)
    a. Enter the following code before the ament_package() line

```
find_package(geometry_msgs REQUIRED)
find_package(rosidl_default_generators REQUIRED)

rosidl_generate_interfaces(${PROJECT_NAME}
  "msg/Turtlemsg.msg"
  "srv/Setpose.srv"
  "srv/Setcolor.srv"
  DEPENDENCIES geometry_msgs
)
```

8. Cd into the turtle_interfaces folder - Open the package.xml file (command code package.xml)
    a. Enter the following code after the <>

```
<build_depend>rosidl_default_generators</build_depend>
<depend>geometry_msgs</depend>

<exec_depend>rosidl_default_runtime</exec_depend>

<member_of_group>rosidl_interface_packages</member_of_group>
```

9. Cd into the turtle_interfaces folder, enter the command “mkdir srv”
10. Cd into the srv directory, then enter the command “vi Setpose.srv” (if you have vim installed)
11. This should open the vim editor for the new file. Press the I key to switch to insert mode, the enter these contents into the file:

```
geometry_msgs/PoseStamped turtle_pose
---
int8 ret
```

12. Now, hit the Esc key to exit insert mode. Next, enter “:wq” to save the file and close it.
13. Cd into the srv directory, then enter the command “vi Setcolor.srv” (if you have vim installed)
14. This should open the vim editor for the new file. Press the I key to switch to insert mode, the enter these contents into the file:

```
string color
---
int8 ret
```

15. Now, hit the Esc key to exit insert mode. Next, enter “:wq” to save the file and close it.
16. Cd into the dev_ws directory
17. Enter the command “pip install setuptools==58.2.0”
18. Run the command “colcon build”
19. Run the following command “find . -type f -exec sed -i 's/script-dir/script_dir/g' {} + find . -type f -exec sed -i 's/install-scripts/install_scripts/g' {}”

20. Run the command “colcon build”

Date: 03/12/24

I spent today debugging the Makefile issue leftover from my previous attempts at Checkpoint 1. I kept running into an issue with the Makefile parser. It turns out it was a pathing issue created by the fact that Dropbox created a separate instance of PC local storage named PC (2) on my machine. CMake really did not like the parenthesis, especially since it had automatically tried to escape the whitespace character. The fix for this was to go into Robotics_Middleware_Trial_Python_Turtle\ROS2\dev_ws\build\turtle_interfaces\CMakeFiles\aeent_cmake_python_copy_turtle_interfaces.dir\build.make and edit line 70. This line needed to be changed so that every instance of PC (2) in the paths was surrounded by quotes.

This is an annoying issue that hopefully doesn’t pop up again. As far as I know, there’s not an easy way for me to remove the parentheses myself. Going forward, I will look for this specific problem with any future errors that may arise.

Date: 03/15/24

I spent today working on the Jupyter Notebook file for the WSL version of checkpoint 1. I am about hallways done with it. I also completed the Mid-Semester Commit Summary for RCOS.

Date: 03/29/24

Note: source ros2 from local install, not the setup.bash in the install directory. It does not redirect correctly for some reason

WSL Cams links: https://github.com/dorssel/usbipd-win/wiki/WSL-support#usbip-client-tools

List devices in Command Prompt Admin: powershell "Get-PnpDevice -PresentOnly | Where-Object { $_.InstanceId -match '^USB' }"

Date: 04/05/24

Links for usbipd and camera install

https://github.com/dorssel/usbipd-win
https://github.com/dorssel/usbipd-win/wiki/WSL-support
https://devblogs.microsoft.com/commandline/connecting-usb-devices-to-wsl/

Commands:

- Usbipd list (note the busid)
- Usbipd bind --busid x-x
- Usbpd attach --wsl --busid x-x

Date: 04/12/24

- Must source both the /opt/…setup.bash
- Must also source install/setup.bash

Hitting this issue while trying to run the webcam code - OpenCV can’t open the camera at all
[WARN:0@0.021] global cap_v4l.cpp:997 open VIDEOIO(V4L2:/dev/video0): can't open camera by index
[ERROR:0@0.021] global obsensor_uvc_stream_channel.cpp:159 getStreamChannelGroup Camera index out of range

Date: 04/24/24

Final Update: As it stands, WSL2 functionality with the Python Turtle checkpoints on ROS2 Iron is about 50% complete. The biggest roadblock to the end of that checkpoint is that WSL2 does not want to open any camera, even one successfully bound and shared with the Linux subsystem. Future work needs to uncover whether this is a WSL2 issue or an OpenCV issue.