



# Carla 0.9.15 Source Build And Carla ROS2 Bridge Guide

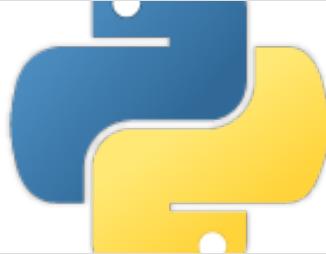
---

## Prerequisites for the Carla Source Build

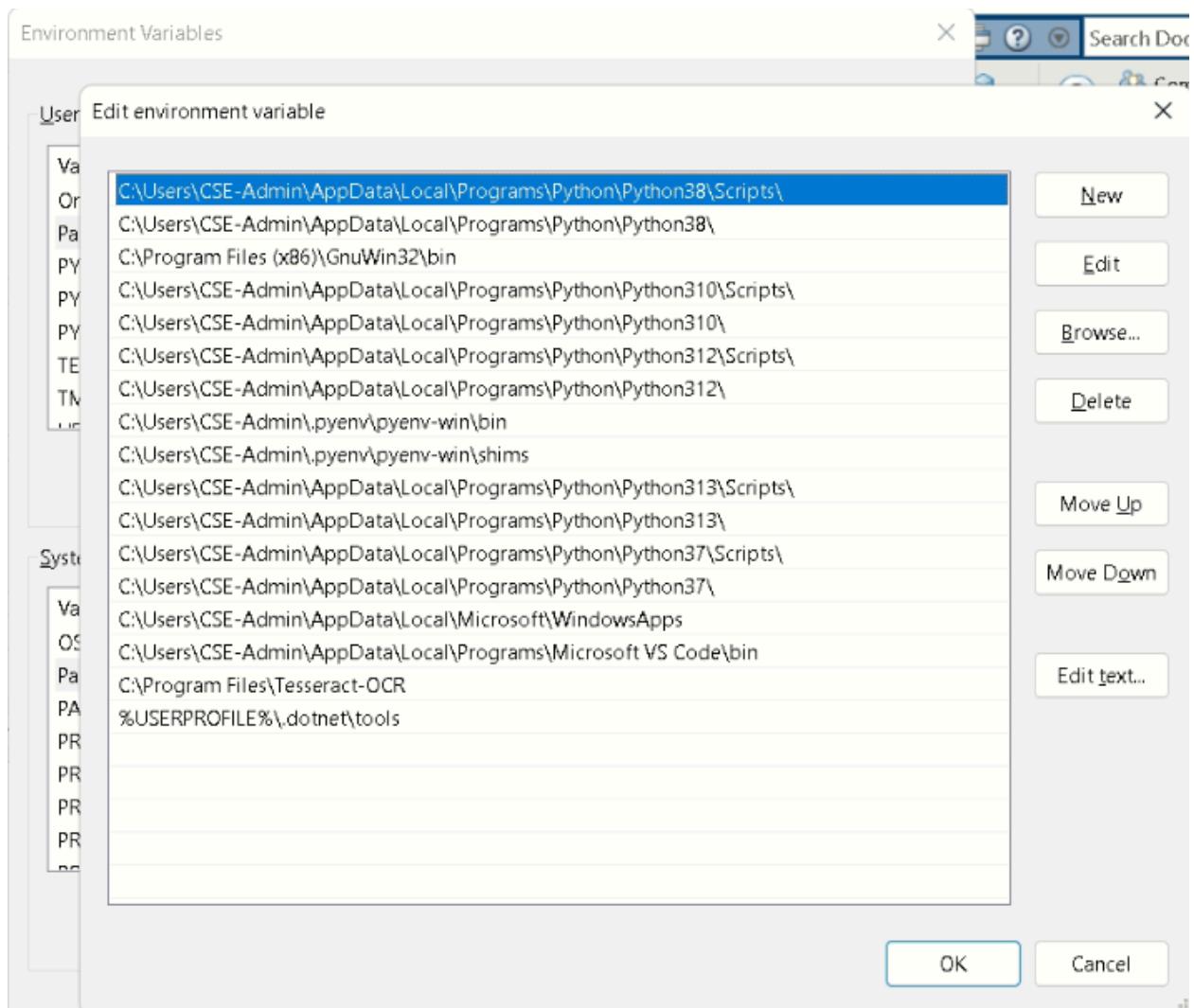
- Python 3.8.x installation. Using any other version of Python 3 (newer or older) will most likely lead to errors during the build process of the client.

Python Release Python 3.8.8  
The official home of the Python Programming Language

 <https://www.python.org/downloads/release/python-388/>



If you have any other Python installations, make sure that the Python 3.8 installation is at the top of the PATH list in your Windows environment system variables.



- **CMake:** Install a version later than 3.9 ( $\geq 3.9$ )

#### Download CMake

You can either download binaries or source code archives for the latest stable or previous release or access the current development (aka nightly) distribution through Git. This software may not be exported in violation of any U.S. export laws or regulations. For more information regarding Export

 <https://cmake.org/download/>

- **Git**

### Download GitHub Desktop

Simple collaboration from your desktop

 <https://desktop.github.com/download/>

- 7-zip. Will be used during the build to extract downloaded files.

### 7-Zip

English

Chinese Simpl.

Chinese Trad.

 <https://www.7-zip.org/>

- Make version 3.81. Make sure it is included in the PATH environment variable as with the Python 3.8 install.

make for Windows

make {whatsoever}

 <https://gnuwin32.sourceforge.net/packages/make.htm>

# Visual Studio 2019 Community Installation

**Before proceeding to install this version of VS, make sure you have completely uninstalled any other installed versions of VS (except for VS Code). The steps for the full VS uninstallation are available in the "Remove all with InstallCleanup.exe" section at the link below. While some Carla Build guides suggested that building Unreal and Carla from source was possible using VS Community 2022, in our findings, we faced many problems configuring Carla dependencies for VS Community 2022. Thus, we do not recommend using this version to build Carla or Unreal if your aim is to get the Carla build running worry-free and as soon as possible.**

### Uninstall or remove Visual Studio

Uninstall or remove your installation of Visual Studio along with its integrated suite of productivity tools for developers.

 <https://learn.microsoft.com/en-us/visualstudio/install/uninstall-visual-studio?view=vs-2022>

 Microsoft Learn



- This version of VS is not currently available on Microsoft's website, so please install it using the installer found in this link:

<https://drive.google.com/file/d/1zV-udnvdFQTtVoVvqju58e24NgnhMUI7/view?usp=sharing>

When running the installer, make sure you choose the following:

- .NET Desktop Development as a **workload**. **Make sure that** .NET Framework 4.6.2 Development Tools is also selected from the installer's Installation details drop-down menu on the right.

## Installation details

- ▶ Desktop development with C++
- ▼ .NET desktop development
  - ▶ Included
  - ▼ Optional
    - .NET development tools
    - .NET Framework 4 – 4.6 development tools
    - Blend for Visual Studio
    - Entity Framework 6 tools
    - .NET profiling tools
    - IntelliCode
    - Just-In-Time debugger
    - Live Share
    - ML.NET Model Builder (Preview)
    - .NET SDK (out of support)
    - F# desktop language support
    - PreEmptive Protection - Dotfuscator
    - .NET Framework 4.6.1 development tools
    - .NET Framework 4.6.2 development tools
    - .NET Framework 4.7 development tools
    - .NET Framework 4.7.1 development tools
    - .NET Framework 4.8 development tools
    - .NET Portable Library targeting pack
    - Windows Communication Foundation
    - SQL Server Express 2016 LocalDB

- Desktop development using C++ as a **workload**. Make sure that MSVC v142 - VS 2019 C++ x64/86 build... and the latest Windows 10 SDK are selected from the installer's Installation details drop-down menu on the right.

## Installation details

### ▼ Desktop development with C++

#### ▼ Included

✓ C++ core desktop features

#### ▼ Optional

MSVC v142 - VS 2019 C++ x64/x86 build t...

Windows 10 SDK (10.0.19041.0)

Just-In-Time debugger

C++ profiling tools

C++ CMake tools for Windows

C++ ATL for latest v142 build tools (x86 &...)

Test Adapter for Boost.Test

Test Adapter for Google Test

Live Share

IntelliCode

C++ AddressSanitizer

MSVC v142 - VS 2019 C++ ARM64 build to...

C++ MFC for latest v142 build tools (x86 &...)

C++/CLI support for v142 build tools (Latest)

C++ Modules for v142 build tools (x64/x86...)

C++ Clang tools for Windows (12.0.0 - x64...)

JavaScript diagnostics

Incredibuild - Build Acceleration

Windows 11 SDK (10.0.22000.0)

Windows 10 SDK (10.0.18362.0)

- Game Development Using C++ as a workload.

## Installation details

work item management.

### ▼ Game development with C++

#### ▼ Included

- ✓ C++ core features
- ✓ Windows Universal C Runtime
- ✓ C++ 2019 Redistributable Update
- ✓ MSVC v142 - VS 2019 C++ x64/x86 build t...

#### ▼ Optional

- C++ profiling tools
- C++ AddressSanitizer
- Windows 10 SDK (10.0.19041.0)
- IntelliCode
- Windows 11 SDK (10.0.22000.0)
- Windows 10 SDK (10.0.18362.0)
- Windows 10 SDK (10.0.17763.0)
- Windows 10 SDK (10.0.17134.0)
- Windows 10 SDK (10.0.16299.0)
- Incredibuild - Build Acceleration
- Cocos
- Unreal Engine installer
- Android IDE support for Unreal engine

# Unreal Engine 4 Source Build

Before cloning the Unreal Engine Carla fork, the GitHub account used on the PC where Carla is being built needs to be linked with an Epic Games account as stated in the following guide.

## Unreal Engine on GitHub

Find out how to access Unreal Engine source code on GitHub



<https://www.unrealengine.com/en-US/ue-on-github>



In a cmd prompt, run the following command to clone the Unreal Engine source fork for Carla. It is best to clone the fork as close to the C:\ directory as possible.

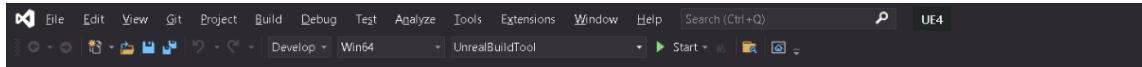
```
git clone --depth 1 -b carla https://github.com/CarlaUnreal/UnrealEngine.git .
```

Following this step, run the following commands:

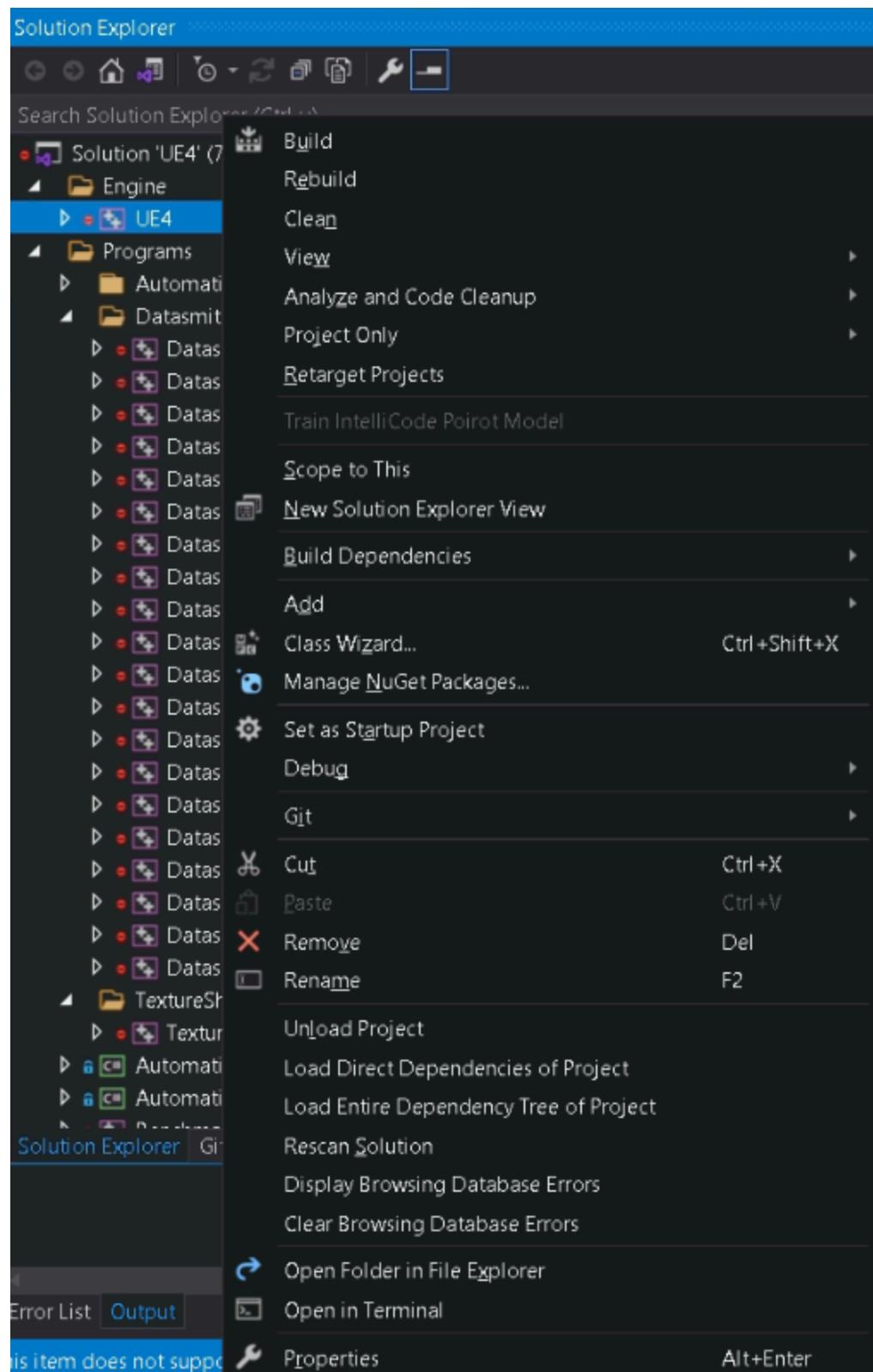
Setup.bat

GenerateProjectFiles.bat

Afterwards, open VS Community 2019 and open the file **UE4.sln** found in the directory where UE4 is placed. In the build bar, ensure that you have selected 'Development Editor', 'Win64', and 'UnrealBuildTool' options.



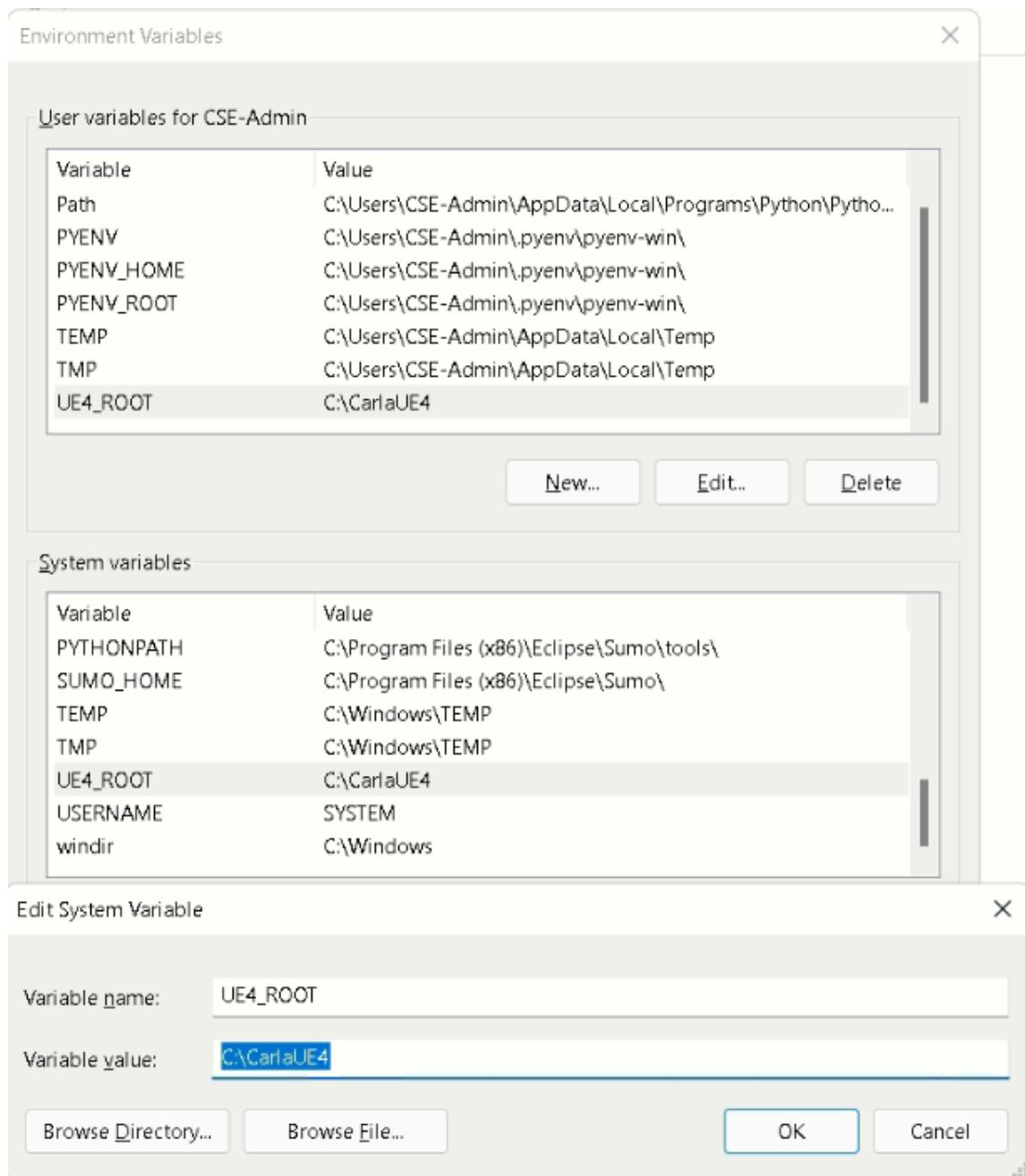
In the solution explorer on the right, right-click UE4 and select Build. Unreal 4 build. This process TAKES TIME, so please exercise patience.



To launch the UE4 Editor, run the engine by executing the following command in a command prompt:

```
path_to\CarlaUE4\Engine\Binaries\Win64\UE4Editor.exe
```

After the installation is complete, ensure that the root directory of the UE4 source build is added to the PATH environment variable in a similar manner to the figure below:



# Carla Source Build

## Important Notes

Having built Unreal Engine, all that remains is building Carla's PYTHONAPI and Server. This is a tedious process, and you will likely face many errors during the

build. In case you face any errors not included in this guide, search around the Carla official GitHub repository and the official Discord Server.

<https://github.com/carla-simulator/carla>

Join the carlasim Discord Server!

Check out the carlasim community on Discord - hang out with 5517 other members and enjoy free voice and text chat.

 <https://discord.com/invite/8kqACuC>

## Carla Build Process

**\*\* All the following commands should run in the Carla root folder and using the x64 Native Tools Command Prompt for VS 2019. Expect this build process to take A LOT OF TIME as your mileage may vary with the build errors. Further errors were faced when attempting the build with VS 2022 thus will not be mentioned in this guide. \*\***



The first step of the build process is to clone the Carla UE4 GitHub Repository.

```
git clone https://github.com/carla-simulator/carla
```

After the git clone is done, run the following command to install the latest Carla assets:

```
Update.bat
```

After the assets are done downloading, run the following command to compile the Carla Client:

make PythonAPI

## OSM2ODR.h Error

The following error is typically faced when running the PYTHONAPI. It arises near the end of the build:

```
C:\carla\PythonAPI\carla\source\libcarla\OSM2ODR.cpp(7): fatal error C1083: Ca
```

osm2odr is a tool that allows the user to generate maps with OpenStreetMap. It converts the generated maps from the .osm format to .xodr, which is used by Carla.

To fix this, first remove the following 2 directories:

```
c:\carla\Build\osm2odr-visualstudio  
c:\carla\Build\osm2odr-source
```

Download the .lib file below and paste it in this folder:

```
C:\carla\Unreal\CarlaUE4\Plugins\Carla\CarlaDependencies\lib
```

Then download the header file below and paste it in this folder:

```
C:\carla\Unreal\CarlaUE4\Plugins\Carla\CarlaDependencies\include
```

Download the osm2odr-source zip file and extract it inside the Build directory:

```
C:\carla\Build
```

You will also need to install **xerces-c**, which is an XML parser written in C++. This parser is available to download and install using Anaconda and its lightweight version Miniconda. It is recommended to use Miniconda as it is more suitable for this purpose given we don't need the entire Anaconda installation.

Download the Miniconda Python 3.8 installer

repo.anaconda.com

[https://repo.anaconda.com/miniconda/Miniconda3-py38\\_23.11.0-2-Windows-x86\\_64.exe](https://repo.anaconda.com/miniconda/Miniconda3-py38_23.11.0-2-Windows-x86_64.exe)

Install Xerces by running

```
conda install -c anaconda xerces-c
```

Navigate to this directory

Path\_To\_miniconda3\pkgs\xerces-c-3.2.3-ha925a31\_0\Library

Copy all the folders in this directory into the following folder (if the folder is not found in the build folder create it and paste the copied folders inside it)

c:\carla\Build\xerces-c-3.2.3-install

Copy the following file

c:\carla\Build\xerces-c-3.2.3-install\lib\xerces-c\_3.lib

Paste the lib file here

c:\carla\PythonAPI\carla\dependencies\lib

Navigate back to the root Carla Folder and re-compile the PYTHONAPI

```
cd c:\carla  
make PYTHONAPI
```

If the compilation was successful you should be able to see the following at the end of it

```

x64 Native Tools Command Prompt for VS 2019
running build
running build_ext
installing to build\bdist.win-amd64\wheel
running install
running install_lib
creating build\bdist.win-amd64\wheel\carla
copying build\lib.win-amd64-cpython-311\carla\command.py > build\bdist.win-amd64\wheel\carla
copying build\lib.win-amd64-cpython-311\carla\libcarla.cp311-win_amd64.pyd > build\bdist.win-amd64\wheel\carla
copying build\lib.win-amd64-cpython-311\carla\__init__.py > build\bdist.win-amd64\wheel\carla
running install_egg_info
Copying source\carla.egg-info to build\bdist.win-amd64\wheel\carla-0.9.15-py3.11.egg-info
running install_scripts
C:\Users\4486J\AppData\Roaming\Python\Python311\site-packages\wheel\bdist_wheel.py:108: RuntimeWarning: Config variable 'Py_DEBUG' is unset, Python ABI tag may be incorrect
  if get_flag("Py_DEBUG", hasattr(sys, "gettotalrefcount"), warn=(impl == "cp")):
creating build\bdist.win-amd64\wheel\carla-0.9.15.dist-info\WHEEL
creating 'dist\carla-0.9.15-cp311-cp311-win_amd64.whl' and adding 'build\bdist.win-amd64\wheel' to it
adding 'carla\__init__.py'
adding 'carla\command.py'
adding 'carla\libcarla.cp311-win_amd64.pyd'
adding 'carla-0.9.15.dist-info\METADATA'
adding 'carla-0.9.15.dist-info\WHEEL'
adding 'carla-0.9.15.dist-info\top_level.txt'
adding 'carla-0.9.15.dist-info\RECORD'
removing build\bdist.win-amd64\wheel

-[BuildPythonAPI]: Carla lib for python has been successfully installed in "C:\carla\PythonAPI\carla\dist"!
C:\carla>

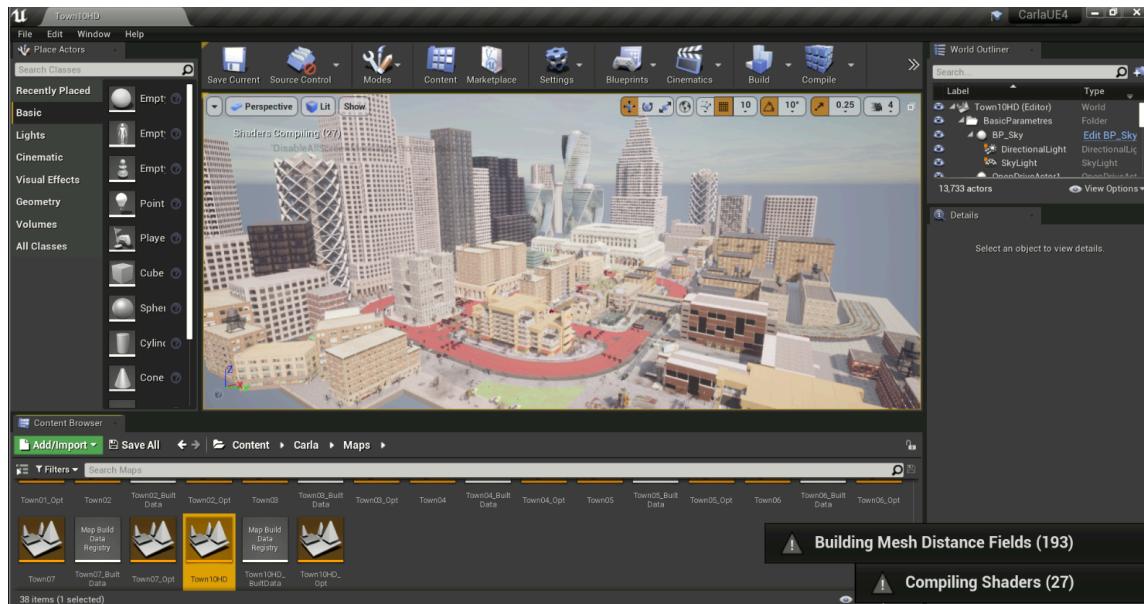
```

## CARLA Server Build

To compile the server after building the PYTHONAPI simply run this command

make launch

This will compile the carla server and launch the environment in the Unreal Engine 4 Editor. The first run the process will take a lot of time (several hours) as the PC's GPU renders the shaders found in all the Carla assets available at the build folder.



If you want to run the compiled binary version instead of running Carla through the editor, you can launch the following .exe file

```
C:\carla\Unreal\CarlaUE4\Binaries\Win64\CarlaUE4.exe
```

## Carla ROS2 Bridge Container

The carla-ros-bridge is the component that facilitates the powerful combination of CARLA and ROS. It retrieves data from the simulation to publish it on ROS topics while simultaneously listening on different topics for requested actions, which are translated to commands to be executed in CARLA. It communicates via DDS to interact with ROS, and via RPC to interact with the CARLA Python API.

### 1. Prerequisites

#### 1. Windows Features

- **WSL 2 and Virtual Machine Platform** enabled
  - Open PowerShell as Administrator and run:

```
dism.exe /online /enable-feature /featurename:Microsoft-Windows  
-Subsystem-Linux /all /norestart  
dism.exe /online /enable-feature /featurename:VirtualMachinePlatf  
orm /all /norestart
```

- Reboot, then install Ubuntu 22.04 from the Microsoft Store.

#### 2. Docker Desktop

- Download & install from <https://www.docker.com/products/docker-desktop>
- In Settings → **General** enable “Use the WSL 2 based engine”
- In Settings → **Resources → WSL Integration**, turn on integration for the Ubuntu distro.

- In Settings → **Resources** → **File Sharing**, ensure the **C:** drive (or whatever drive hosts the bag folder) is shared.

### 3. X Server for Windows (to forward any GUI from the container)

- Install **VcXsrv** (<https://sourceforge.net/projects/vcxsrv/>) or **Xming**.
- Launch it (e.g. with “Disable access control”), so that the container can connect to `host.docker.internal:0.0`.

### 4. Local bags folder

- Create a folder on the C: drive for recordings, e.g. `C:\rosbags`.

### 5. Pulled Docker Image for the ROS2 CARLA Bridge

hub.docker.com

 <https://hub.docker.com/r/rwthika/carla-ros-bridge?uuid=EC25BB51-7DE8-4BA0-9843-072BF2A202AB>

`docker pull rwthika/carla-ros-bridge`

## 2. Configure PowerShell Environment

1. Open a plain **PowerShell** (not WSL) as the regular user.
2. Set the **DISPLAY** variable so that Linux GUI apps know where to forward:

```
# point at the X server you launched
$env:DISPLAY = "host.docker.internal:0.0"
```

### 3. Enable Docker CLI to see the WSL folders

Docker Desktop will map `C:\rosbags` automatically.

### 3. Launch the CARLA Simulator

1. In a separate window, navigate to the CARLA folder, and run:

```
cd C:\carla\Unreal\CarlaUE4\Binaries\Win64\CarlaUE4.exe  
.\\CarlaUE4.exe
```

2. Wait until the server starts and you see the viewport.

### 4. Run the ROS 2–CARLA Bridge Container

In a WSL PowerShell session (with `$env:DISPLAY` set) run the following:

```
docker run --rm -it --name carla_bridge \  
-e CARLA_HOST=host.docker.internal \  
-e CARLA_TIMEOUT=60 \  
-e RMW_IMPLEMENTATION=rmw_fastrtps_cpp \  
-e DISPLAY=$env:DISPLAY \  
-e QT_X11_NO_MITSHM=1 \  
-e LIBGL_ALWAYS_SOFTWARE=1 \  
-v C:\\rosbags:/bags \  
-p 2000:2000/tcp -p 2000:2000/udp \  
-p 11811:11811/udp \  
-p 7400:7400/udp -p 7410:7410/udp \  
rwthika/carla-ros-bridge:ros2 bash
```

#### Notes:

- This mounts the Windows path `C:\\rosbags` to `/bags` inside the container.
- Ports **2000**, **11811**, **7400**/7410 (DDS) are forwarded so that CARLA  $\leftrightarrow$  ROS 2 can communicate.

## 5. Inside the Container: Launching ROS 2 Nodes

Open a **WSL** terminal to start running nodes on the container

- 1. Start the core bridge in the first opened container terminal**

```
ros2 launch carla_ros_bridge carla_ros_bridge.launch.py host:=${CARLA_A_HOST} port:=2000 timeout:=${CARLA_TIMEOUT} town:=Town02 spawn_sensors:=True
```

- 2. Run the ros2 node to spawn an ego vehicle**

```
docker exec -it carla_bridge bash  
ros2 launch carla_spawn_objects carla_spawn_objects.launch.py
```

- 3. Run this script to give throttle command to the ego vehicle**

```
docker exec -it carla_bridge bash  
ros2 topic pub /carla/ego_vehicle/vehicle_control_cmd carla_msgs/msg/CarlaEgoVehicleControl "{throttle: 0.3, steer: 0.0, brake: 0.0, hand_brake: false, reverse: false, gear: 0, manual_gear_shift: false}" --rate 10
```

- 4. list the topics in the environment**

```
docker exec -it carla_bridge bash  
ros2 topic list
```

- 5. Log the simulation in a ros2 bag to rerun the simulation later**

```
docker exec -it carla_bridge bash  
ros2 bag record -e "/carla/.*" -o /bags/sensors_run
```

```
Windows PowerShell x root@e48d94db448d:/docke + + entationCamera[id=164] [bridge-1] [INFO] [1747786955.284393165] [carla_ros_bridge]: Created DepthCamera(id=165) [bridge-1] [INFO] [1747786955.289521877] [carla_ros_bridge]: Created DVS Camera(id=166) [bridge-1] [INFO] [1747786955.293761426] [carla_ros_bridge]: Created GnsS(id=167) [bridge-1] [INFO] [1747786955.297396359] [carla_ros_bridge]: Created ImuSensor(id=168) [bridge-1] [INFO] [1747786955.302705446] [carla_ros_bridge]: Created CollisionSensor(id=169) [bridge-1] [INFO] [1747786955.306088112] [carla_ros_bridge]: Created LaneInvasionSensor(id=170) [bridge-1] [INFO] [1747786955.308685342] [carla_ros_bridge]: Created TFSensor(id=10006) [bridge-1] [INFO] [1747786955.310346930] [carla_ros_bridge]: Created ObjectSensor(id=10007) [bridge-1] [INFO] [1747786955.314524943] [carla_ros_bridge]: Created OdometrySensor(id=10008) [bridge-1] [INFO] [1747786955.316861611] [carla_ros_bridge]: Created SpeedometerSensor(id=10009) [bridge-1] [INFO] [1747786955.319663558] [carla_ros_bridge]: Created ActorControl(id=10010)

root@e48d94db448d:/docke-ros/wst# ros2 topic pub /carla/ego_vehicle/vehicle_controller_cmd carla_msgs/msg/CarlaEgoVehicleControl "[throttle: 0.3, steer: 0.0, brake: 0.0, hand_brake: false, reverse: false, gear: 0, manual_gear_shift: false]" --rate 10

root@e48d94db448d:/docke-ros/wst# docker-ros/wst:ros2 topic pub /carla/ego_vehicle/vehicle_controller_cmd carla_msgs/msg/CarlaEgoVehicleControl "[throttle: 0.3, steer: 0.0, brake: 0.0, hand_brake: false, reverse: false, gear: 0, manual_gear_shift: false]" --rate 10

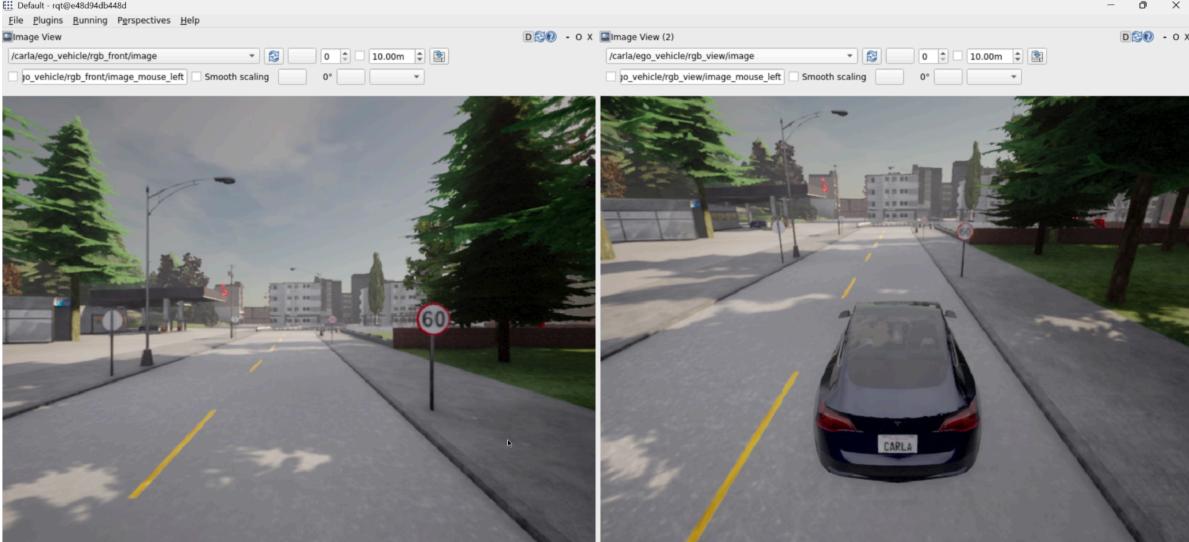
bject (type='sensor.camera.dvs', id='dvs_front') spawned successfully as 166. [carla_spawn_objects-1] [INFO] [1747786955.072399412] [carla_spawn_objects]: 0 bject (type='sensor.other.gnss', id='gnss') spawned successfully as 167. [carla_spawn_objects-1] [INFO] [1747786955.077385439] [carla_spawn_objects]: 0 bject (type='sensor.other.imu', id='imu') spawned successfully as 168. [carla_spawn_objects-1] [INFO] [1747786955.082484311] [carla_spawn_objects]: 0 bject (type='sensor.other.colision', id='collision') spawned successfully as 169. [carla_spawn_objects-1] [INFO] [1747786955.087430919] [carla_spawn_objects]: 0 bject (type='sensor.other.lane_invasion', id='lane_invasion') spawned successfully as 170. [carla_spawn_objects-1] [INFO] [1747786955.090226258] [carla_spawn_objects]: 0 bject (type='sensor.pseudo_tf', id='tf') spawned successfully as 10006. [carla_spawn_objects-1] [INFO] [1747786955.092395162] [carla_spawn_objects]: 0 bject (type='sensor.pseudo.objects', id='objects') spawned successfully as 10007. [carla_spawn_objects-1] [INFO] [1747786955.094698382] [carla_spawn_objects]: 0 bject (type='sensor.pseudo_odom', id='odom') spawned successfully as 10008. [carla_spawn_objects-1] [INFO] [1747786955.097171280] [carla_spawn_objects]: 0 bject (type='sensor.pseudo.speedometer', id='speedometer') spawned successfully as 10009. [carla_spawn_objects-1] [INFO] [1747786955.100020557] [carla_spawn_objects]: 0 bject (type='actor.pseudo.control', id='control') spawned successfully as 10010. [carla_spawn_objects-1] [INFO] [1747786955.100482227] [carla_spawn_objects]: All objects spawned.

root@e48d94db448d:/docke-ros/wst# rqt QStandardPaths: XDG_RUNTIME_DIR not set! defaulting to '/tmp/runtime-root'. XcbConnection: XCB error: 147 (Unknown ), sequence: 181, resource id: 0, major code: 141 (Unknown), minor code: 20
```

The terminal shell view of the opened shells

## 6. Run RQT to view the camera view of the vehicle

```
docker exec -it carla_bridge bash  
rqt
```



## Camera views of the ego vehicle spawned in the carla simulator

