Windows 10

Requirements

- Visual Studio 2019
- Git
- Make (version 3.81)
  - https://sourceforge.net/projects/gnuwin32/files/make/3.81/
- 7-Zip
  - https://www.7-zip.org/download.html
- Cmake (lastest version)
  - https://cmake.org/download/
- Python3 x64 (in my case use 3.7, no anaconda, because needs py launcher)
  - https://www.python.org/
- Unreal Engine 4.26 source version
  - git clone --depth 1 -b carla https://github.com/CarlaUnreal/UnrealEngine.git

Requirements

- Visual Studio 2019
  - If you install VS 2019, Check below
  - .NET desktop development and .NET Framework 4.6.2 development tools

Requirements

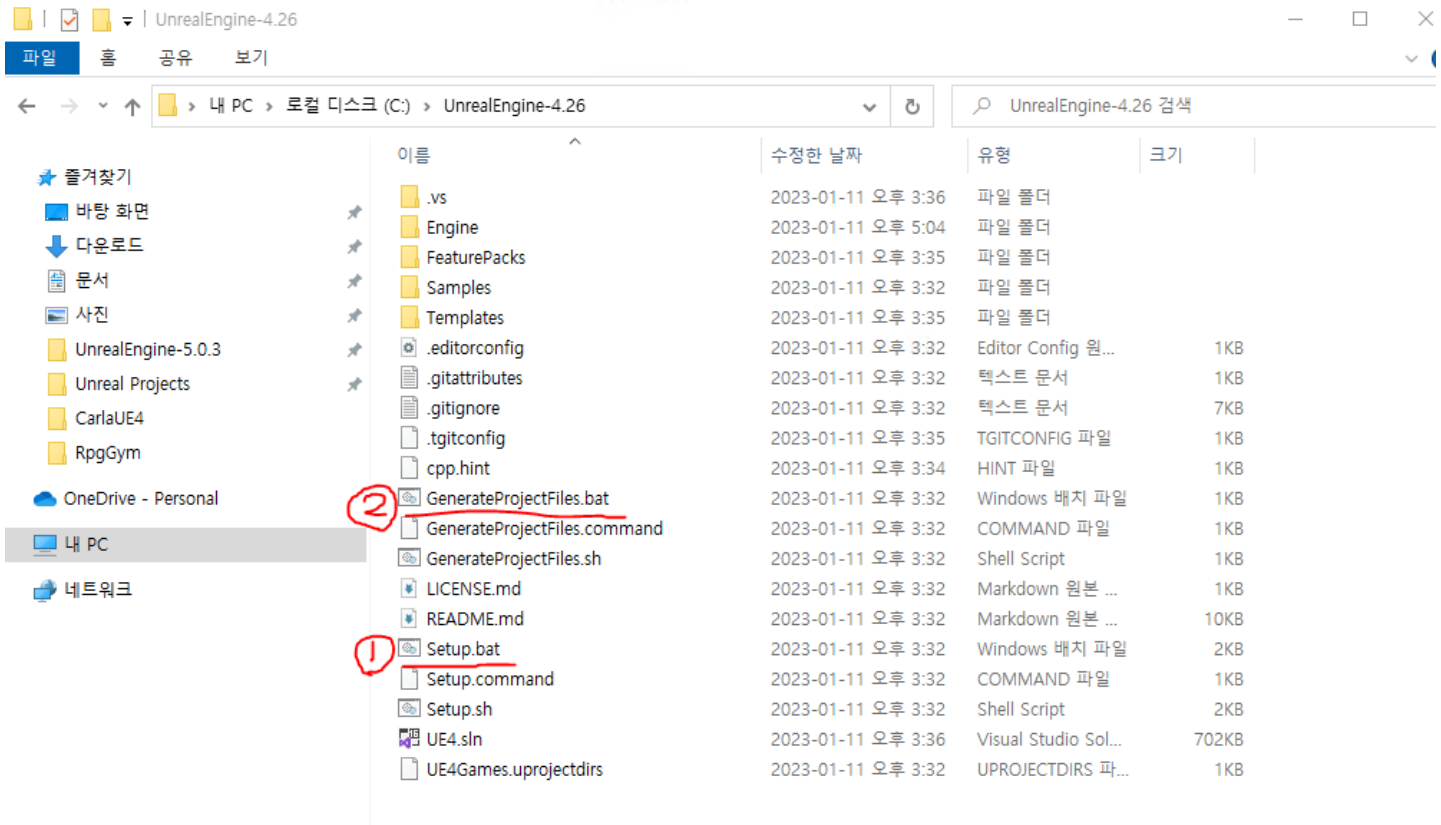
- Visual Studio 2019
  - If you install VS 2019, Check below
  - C++ desktop development and MSVC v140 – VS 2015 c++ build tools(v140)

Requirements

- Visual Studio 2019
  - If you install VS 2019, Check below
  - C++ game development

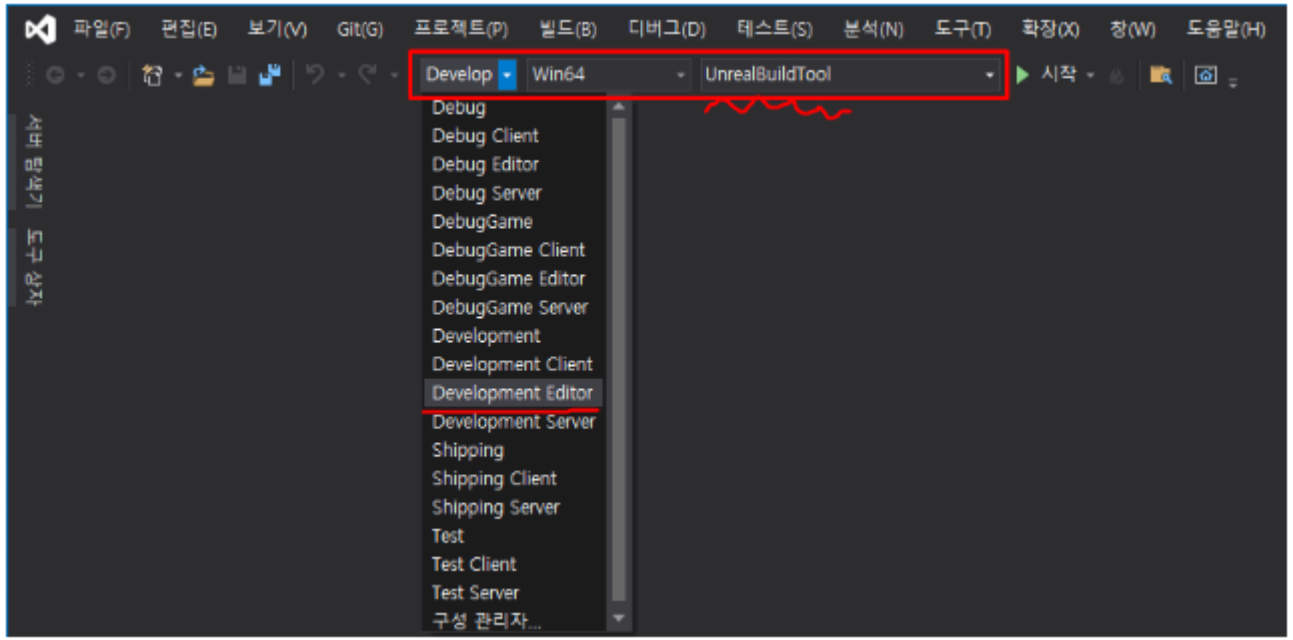
Build Unreal Engine 4.26 source version

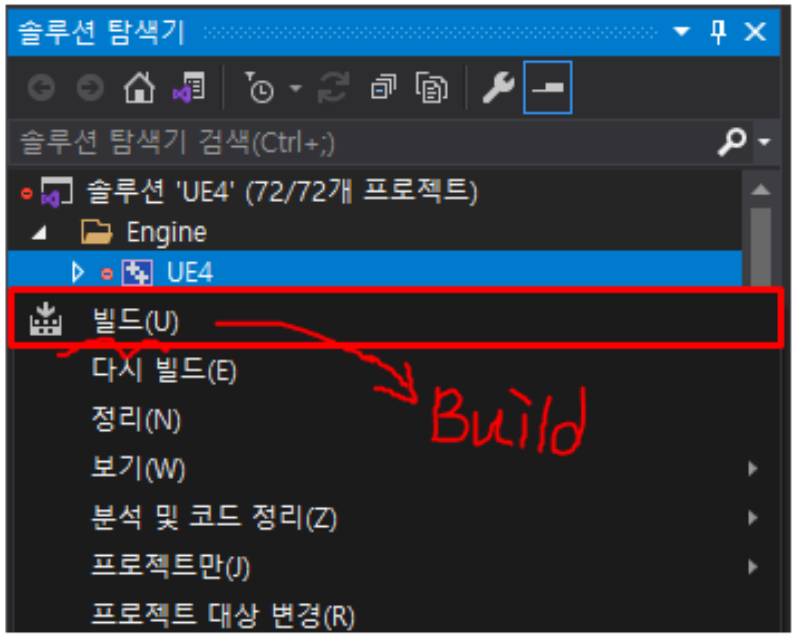
- git clone --depth 1 -b carla https://github.com/CarlaUnreal/UnrealEngine.git
- Finish git clone, go to installed unreal engine path
- First execute Setup.bat
- Second execute GenerateProjectFiles.bat



Build Unreal Engine 4.26 source version

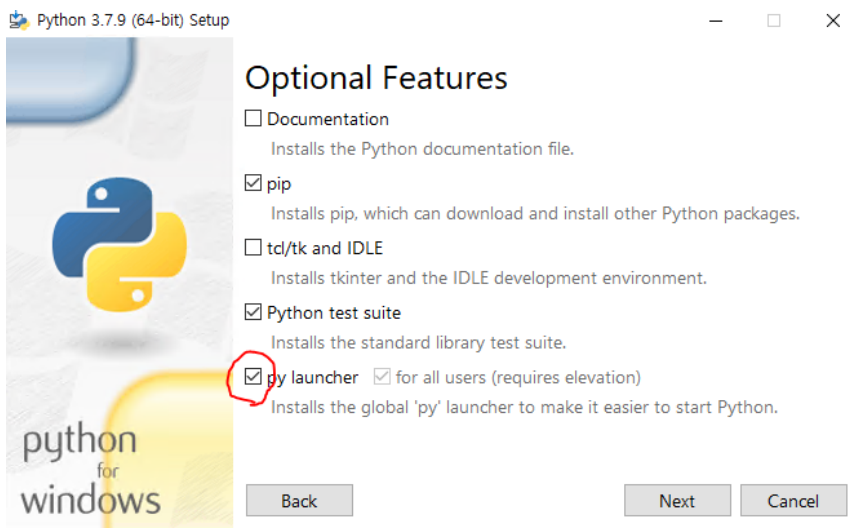
- Finish .bat files, you can see UE4.sln file
- Execute UE4.sln then open the VS 2019
- Change solution configuration (Development Editor) and Project (UnrealBuildTool)
- And build UE4 project (probably a hour)

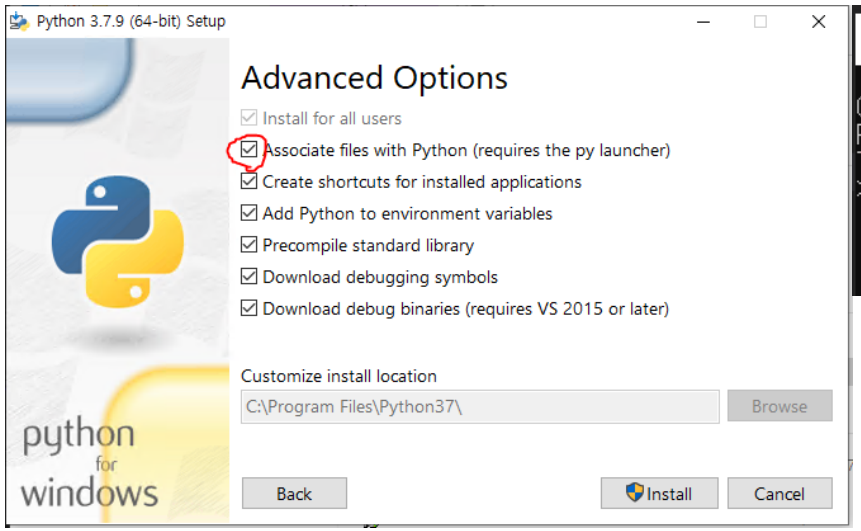


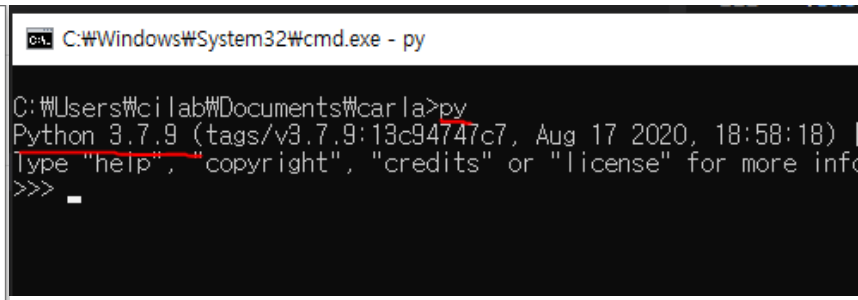


Requirements

- Python3
  - I use python 3.7
  - https://www.python.org/
  - When you install python check below
  - If you use anaconda or another python, have to edit batch files in BuildTools directory's files







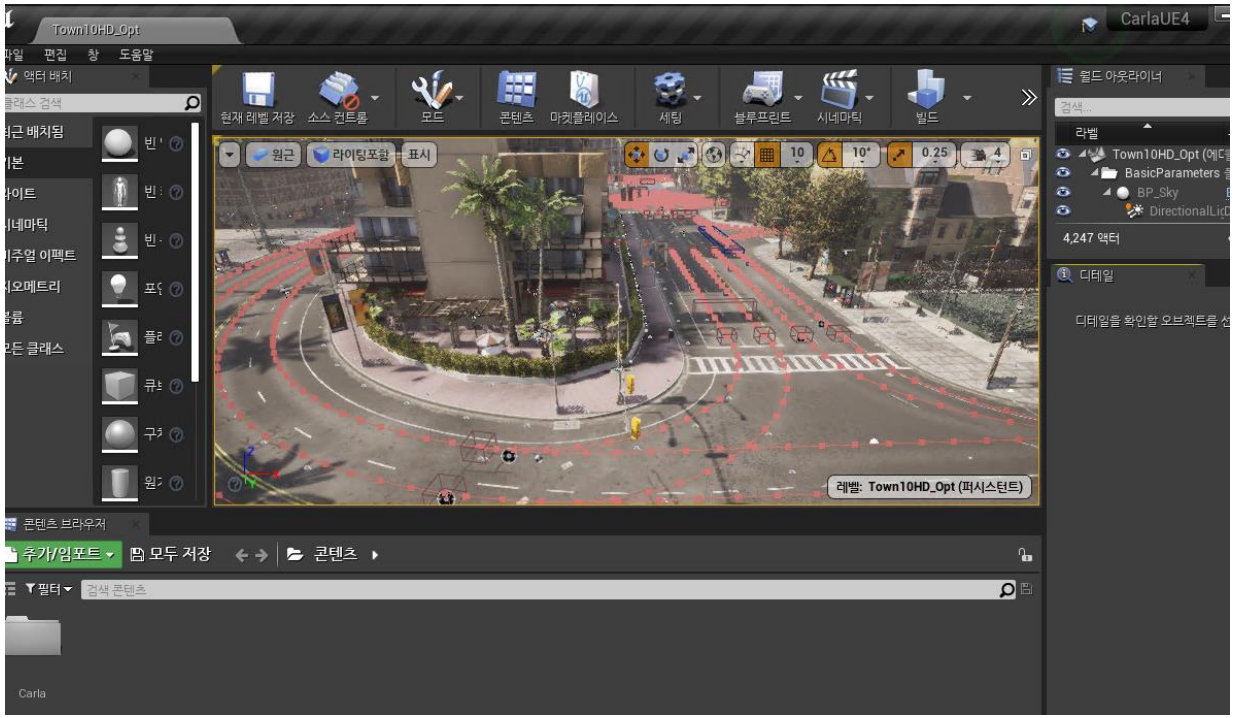
Build Calra (make launch)

- git clone –b 0.9.13 https://github.com/carla-simulator/carla.git
- Finish clone, go to YOUR_CARLA_PATH/Util>installersWins
- Edit install_zlib.bat (1.2.11 -> 1.2.13)
- And execute make launch (in x64 Native Tools Command Prompt for VS 2019)
- If finish normally, you can see Unreal Editor

install_zlib.bat M

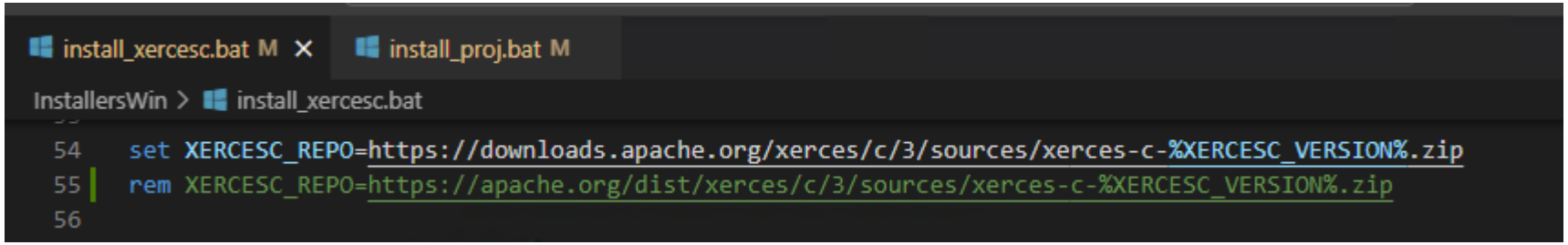
InstallersWin > install_zlib.bat

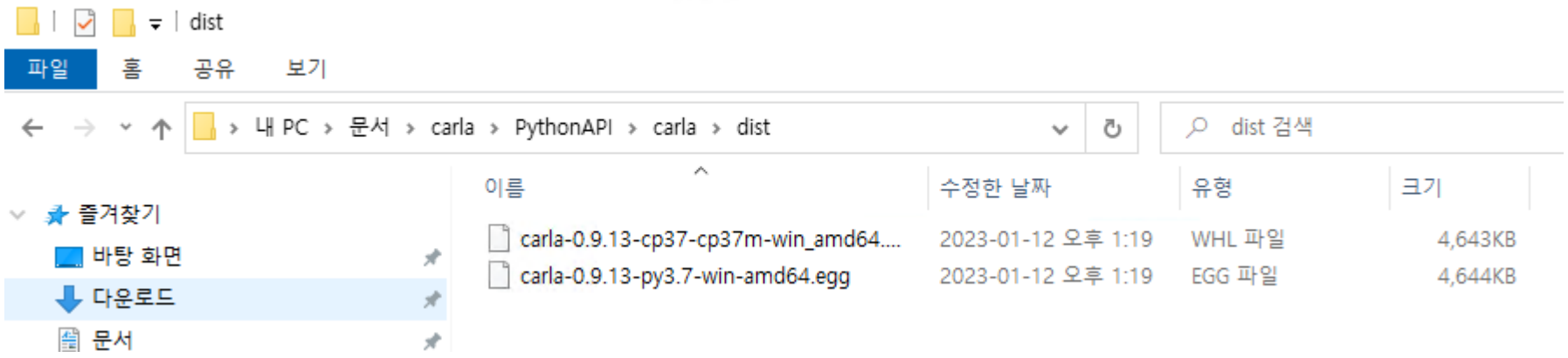
```
48  rem ================================================================
49
50  set ZLIB_BASENAME=zlib
51  set ZLIB_VERSION=1.2.13
52
53  set ZLIB_TEMP_FOLDER=%ZLIB_BASENAME%-%ZLIB_VERSION%
54  set ZLIB_TEMP_FILE=%ZLIB_TEMP_FOLDER%.zip
55  set ZLIB_TEMP_FILE_DIR=%BUILD_DIR%%ZLIB_TEMP_FILE%
56
```



Build Calra (make PythonAPI)

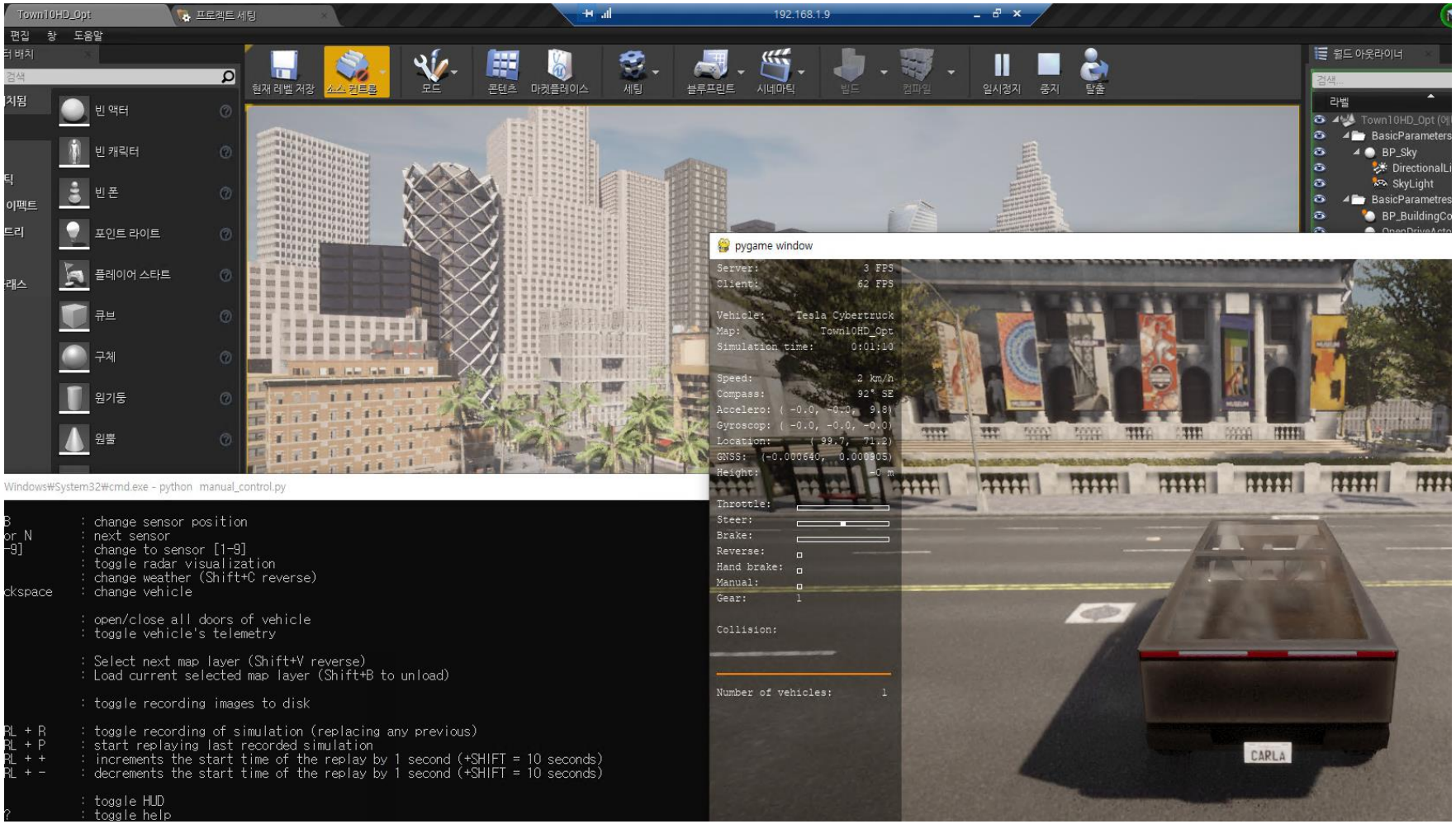
- Edit /YOUR_CARLA_PATH/Util/installersWin/install_xercesc.bat (line 55)
- (Probably, original download link is wrong)
- And execute make PythonAPI (in x64 Native Tools Command Prompt for VS 2019)
- If finish, you can see *.egg file





Test Carla

- Go to /CARLA/PythonAPI/examples
- pip install –r requirements.txt
- Run manual_control.py



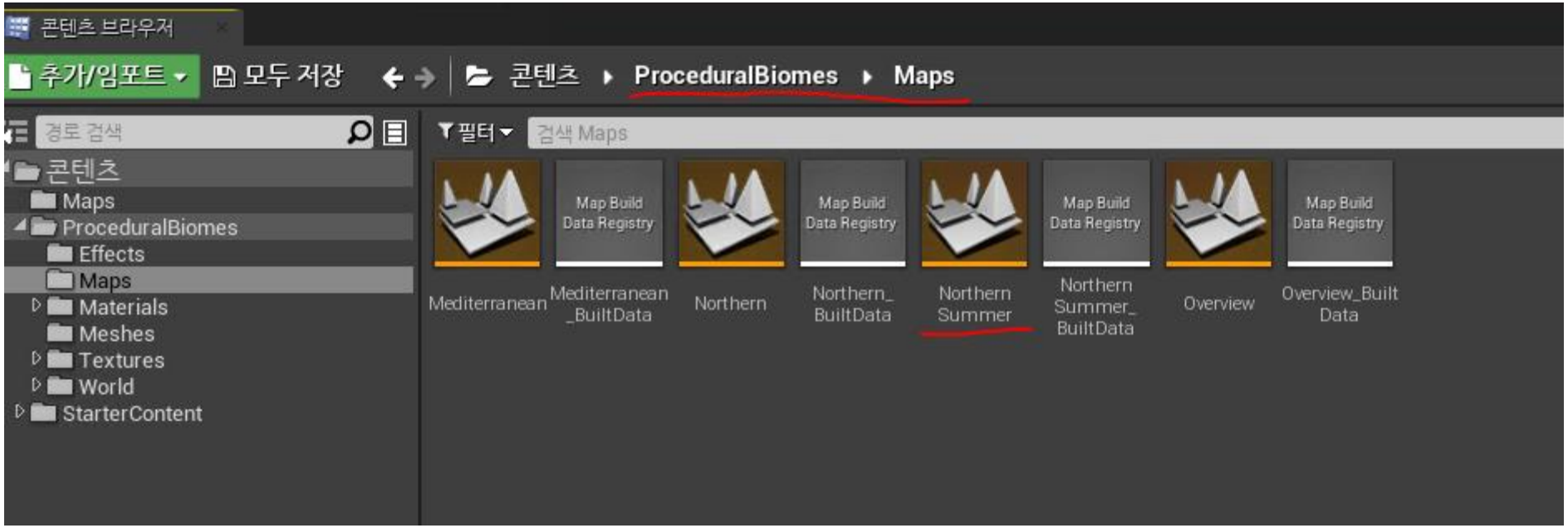
Add custom map

- Create new unreal project (4.26 version). In my case, create ‘CarlaMapProject’
- I introduce two methods.
- First, use free asset package provided epic games in this guide.
- Second, made very simple map (not use asset).

- In my opinion, if you create map using unreal engine (use landscape, create materials, put down asset and etc… you good at unreal engine) just create map
- But you are not, use asset package :)

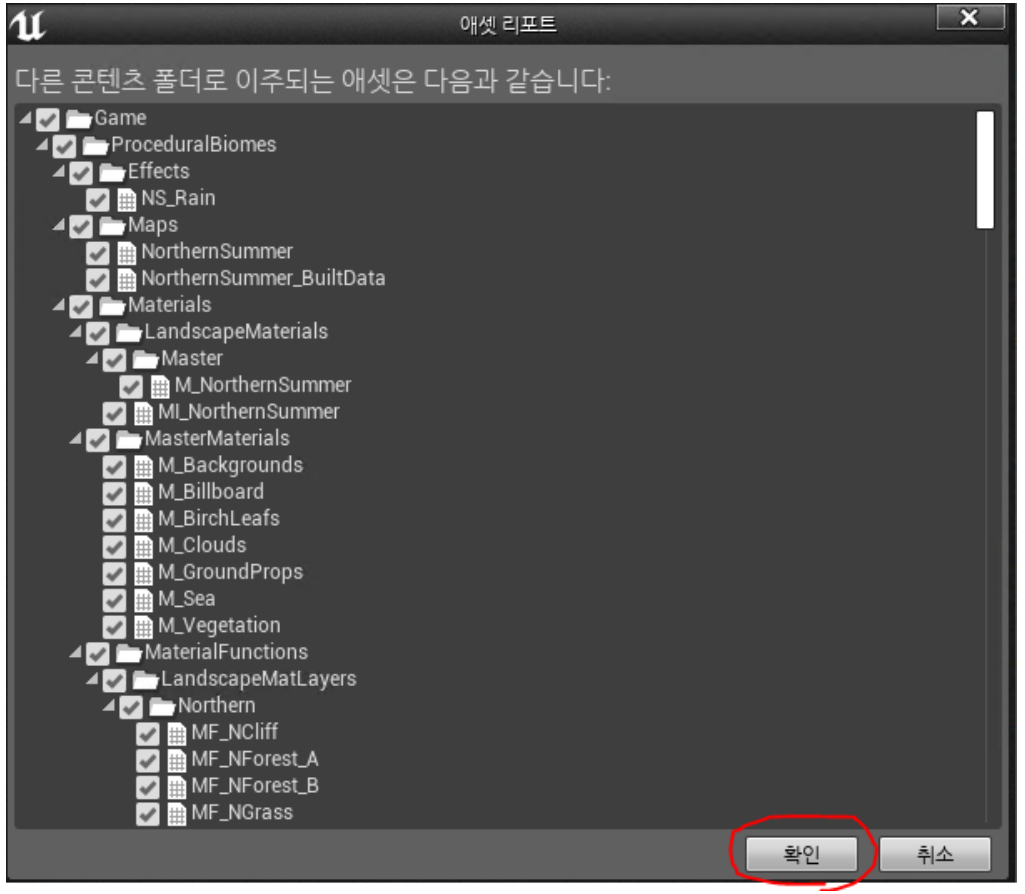
Add custom map (first method)

- I use ‘NorthernSummer’



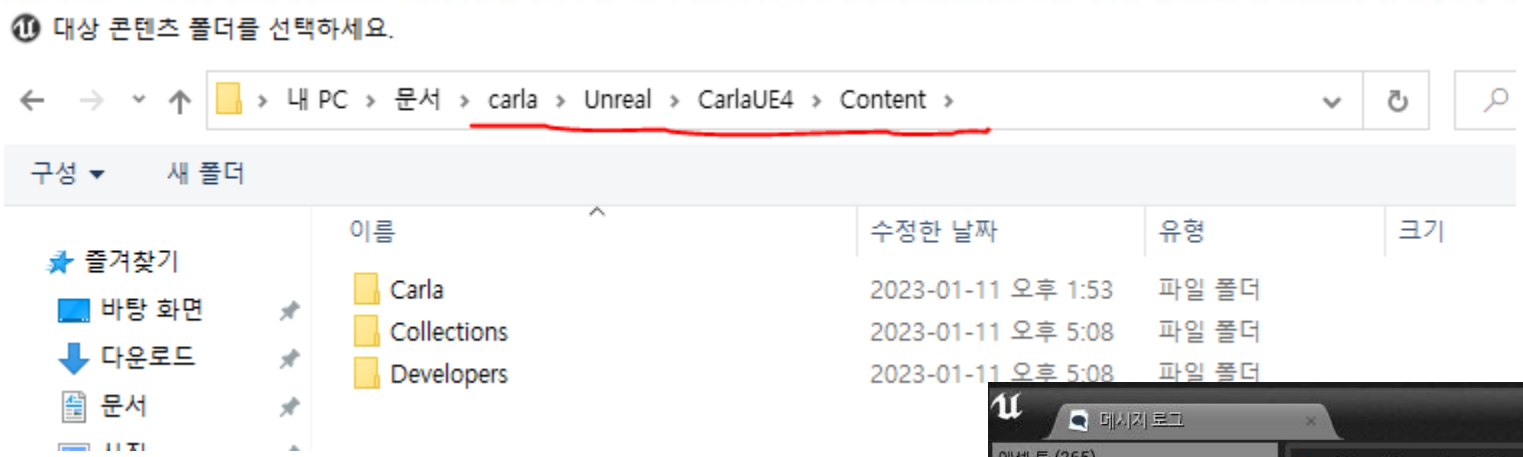
Add custom map (first method)

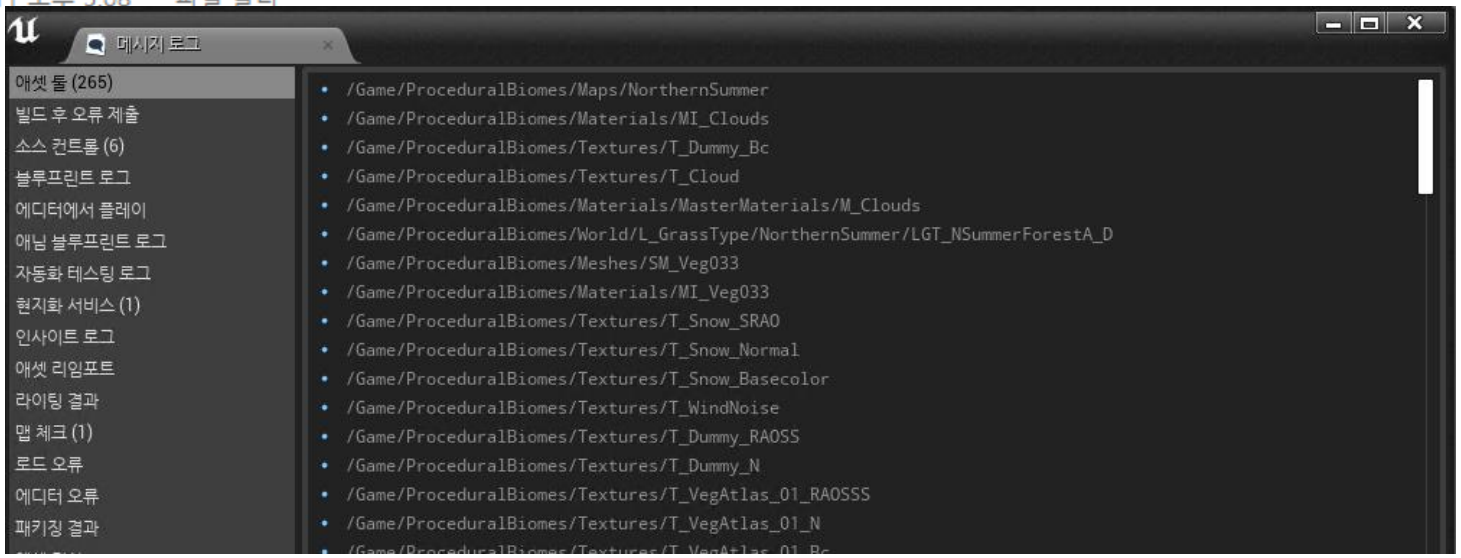
- This file right click and AssetAction > Migration click
- You can see asset report and click ok



Add custom map (first method)

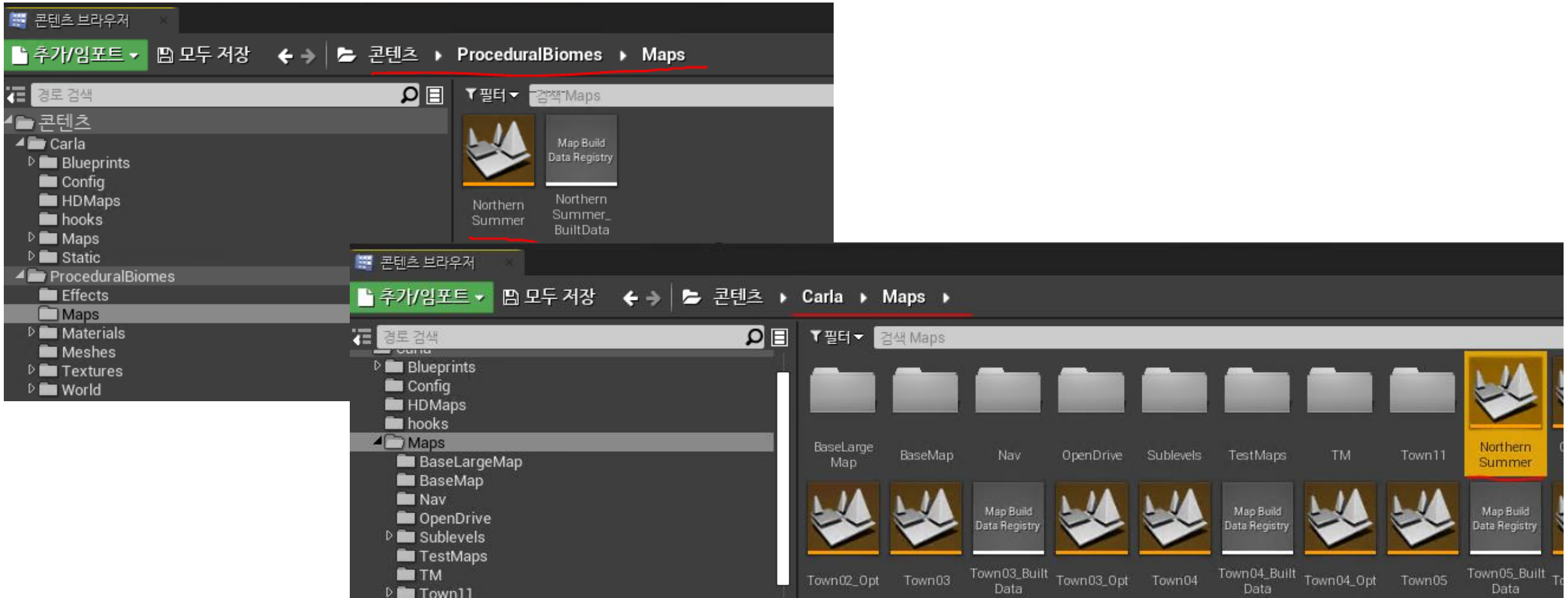
- You can see pop-up window
- Select /YOUR_CARLA/Unreal/CarlaUE4/Content
- If finish, you can see message log





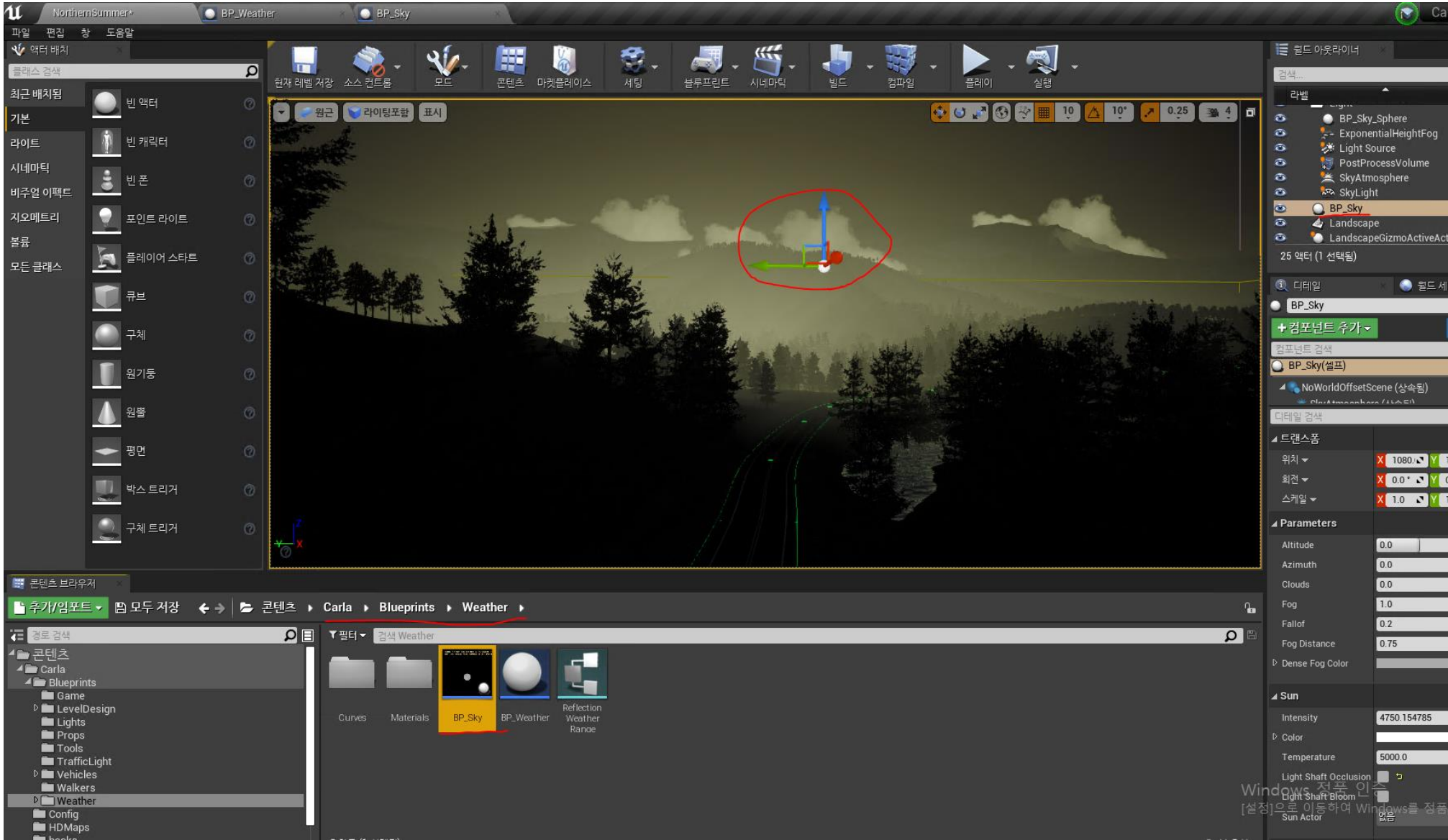
Add custom map (first method)

- Go to YOUR_CARLA and make launch
- You can see migrated files
- Copy or Move this file to Carla/Maps/ (I copy it)



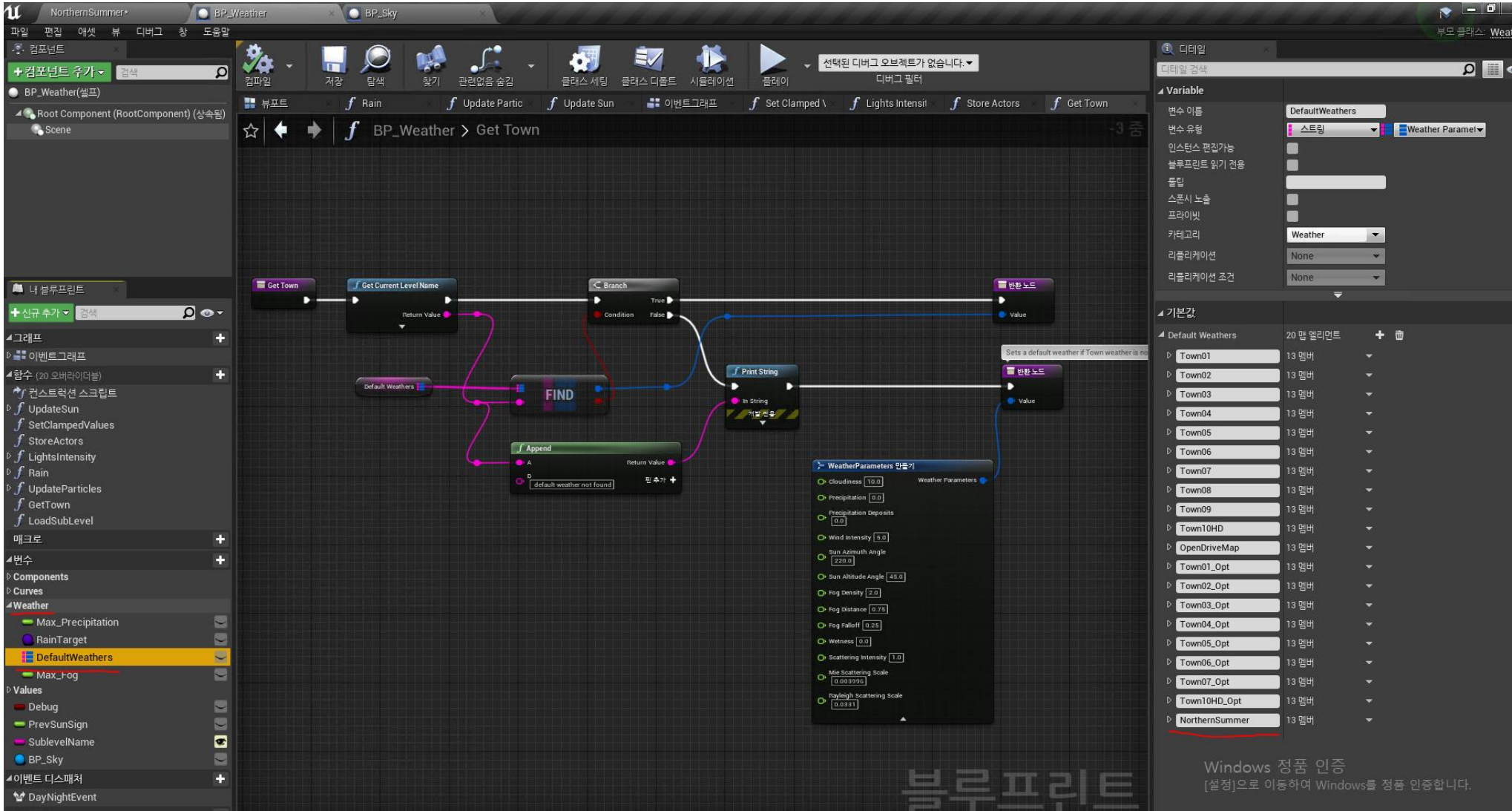
Add custom map (first method)

- Put down BP_Sky in level



Add custom map (first method)

- Add custom map name (Carla/Blueprints/Weather/BP_Weather)

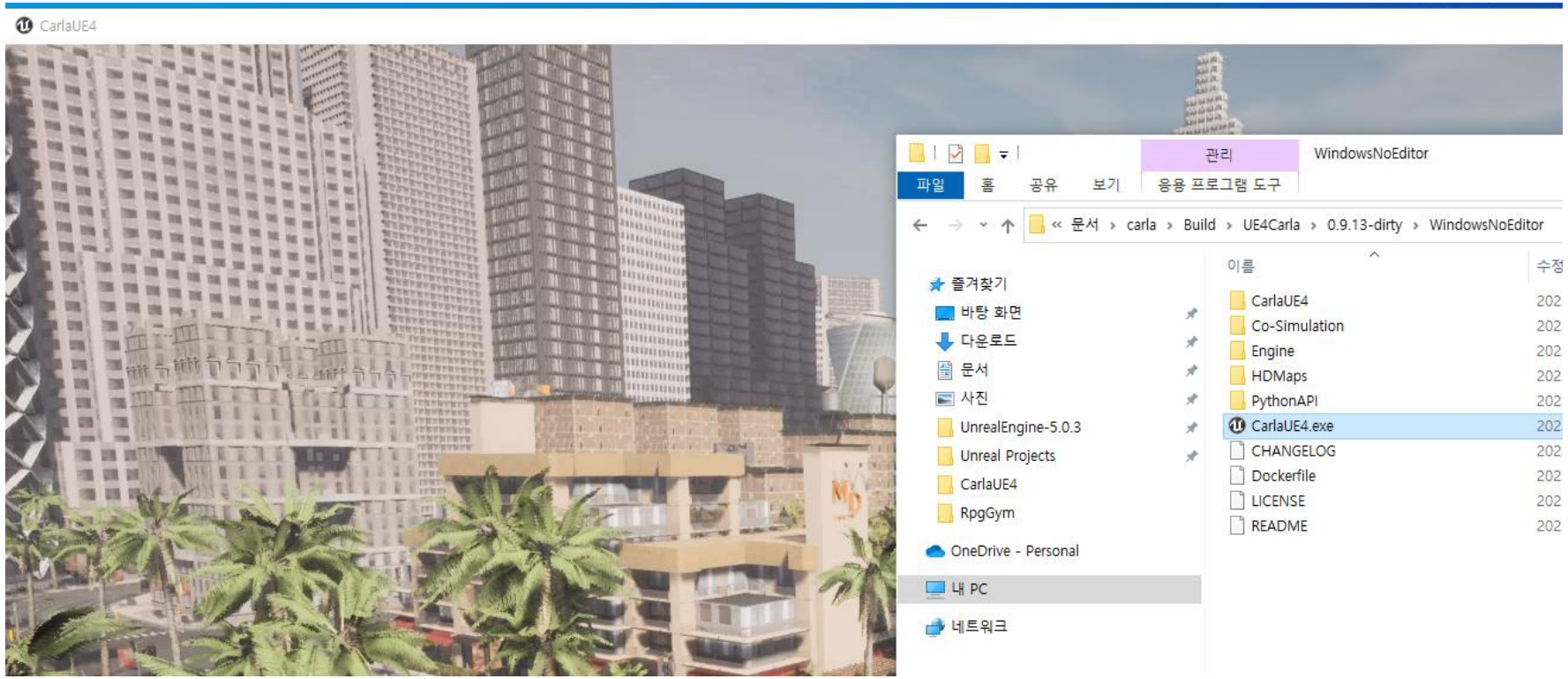


Add custom map (first method)

- Open project setting and go to packaging
- Open tab advanced and “list of maps to include in an packaged build”
- Add custom map (In this add NorthernSummer.umap)
- Close unreal engine editor
- Run make package (in x64 Native Tools Command Prompt for VS 2019)

Add custom map (first method)

- Finish make package, you can see
- Run Carla binary file



Add custom map (first method)

- Run test code (TestCustomMap.py for loading custom map)

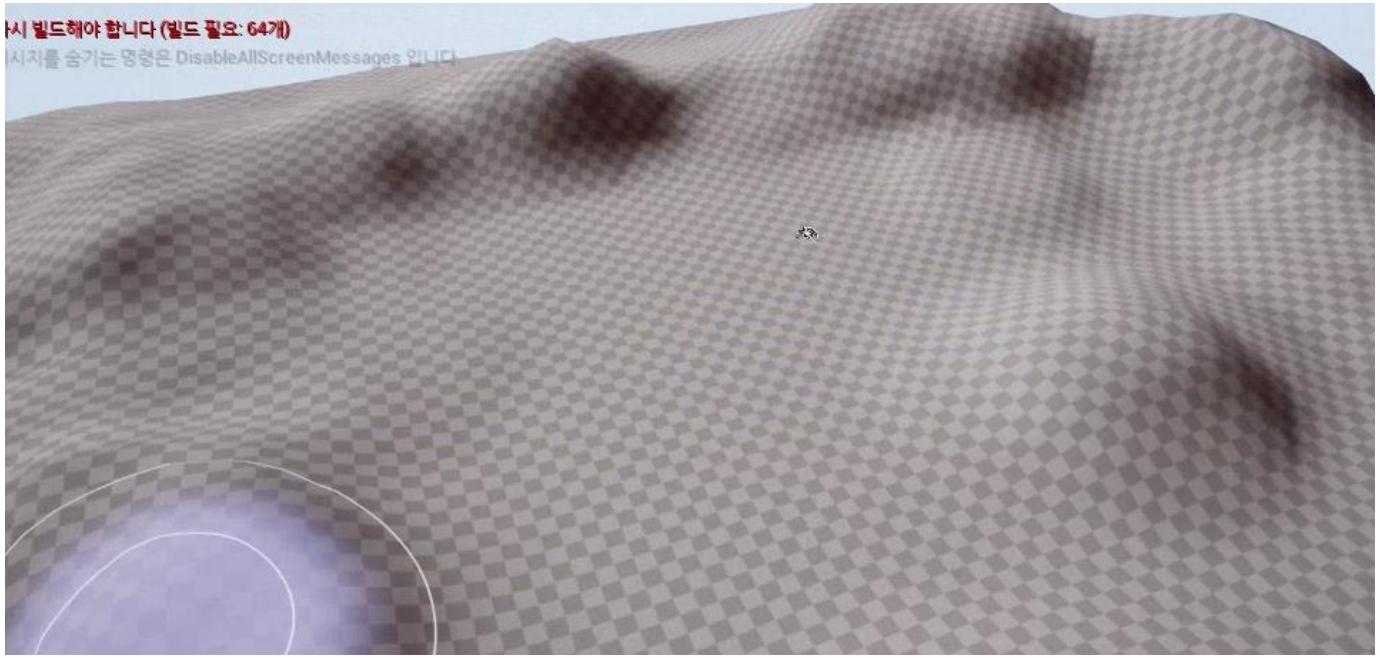
TestCustomMap.py 2, U

TestCustomMap.py > ...

```
24  # ==============================================
25
26
27  import carla
28
29  from carla import ColorConverter as cc
30
31  import argparse
32  import collections
33  import datetime
34  import logging
35  import math
36  import random
37  import re
38  import weakref
39
40  client = carla.Client('127.0.0.1', 2000)
41  world = client.load_world("NorthernSummer")
```

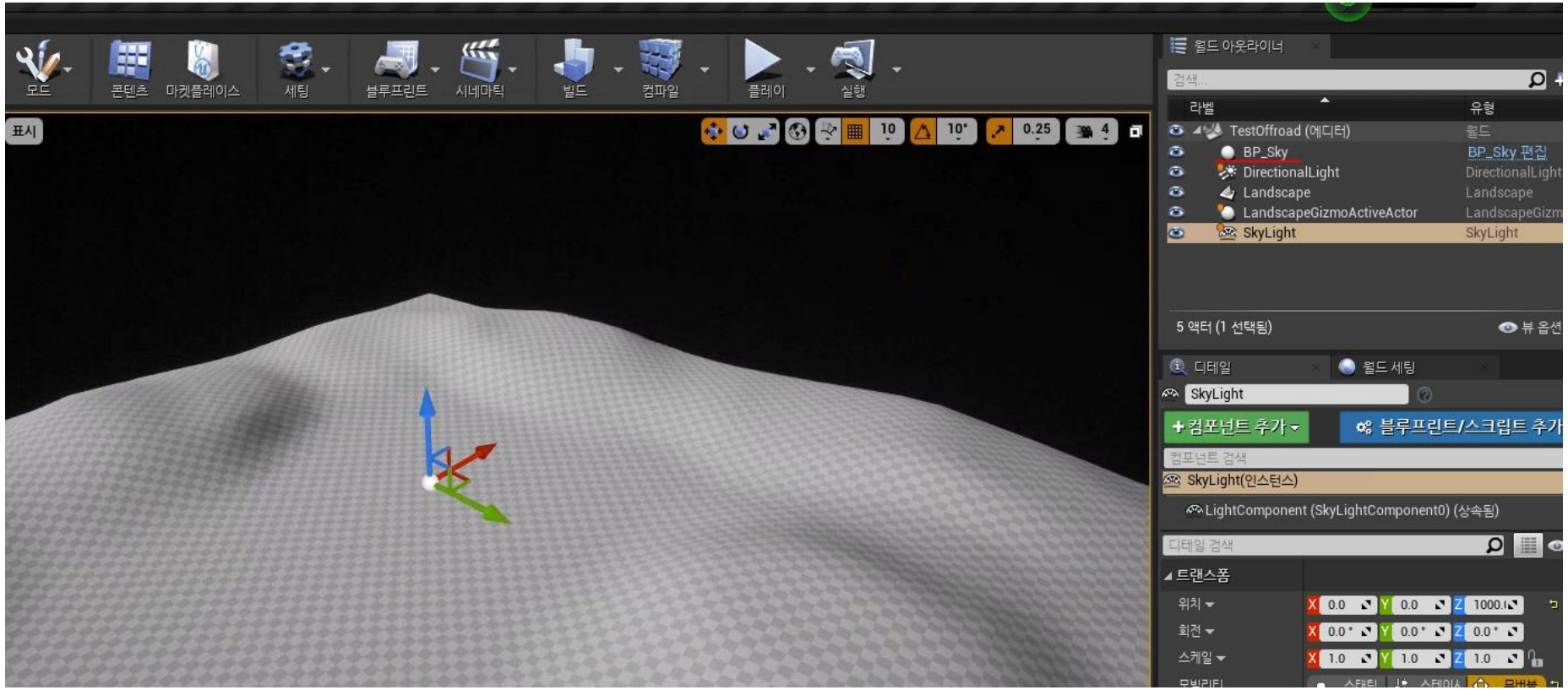
Add custom map (second method)

- You don’t need to asset migration
- Create new level named ‘TestOffroad’ in Carla/Maps
- Add light, sky, and etc…
- Add Landscape



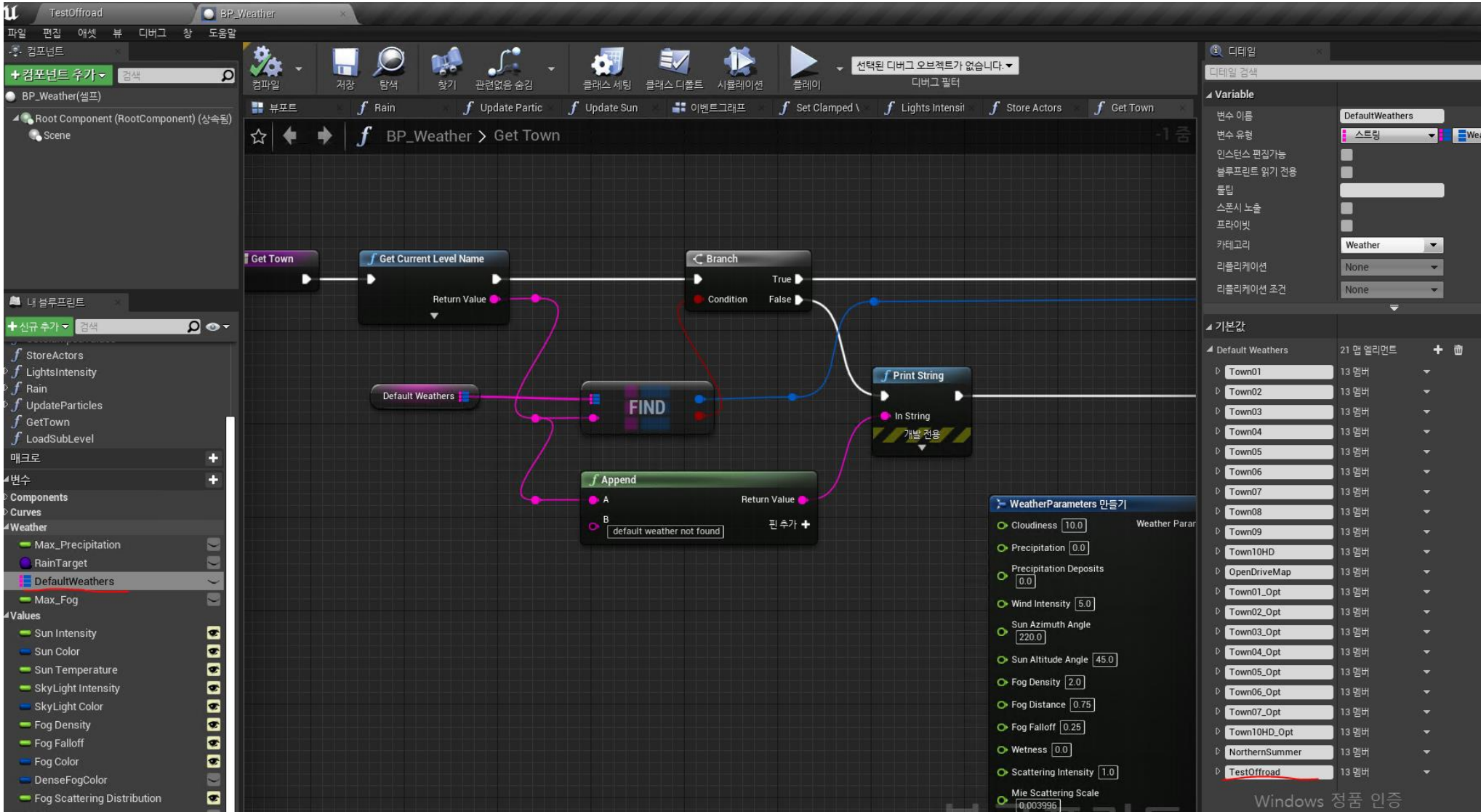
Add custom map (first method)

- Put down BP_Sky in level



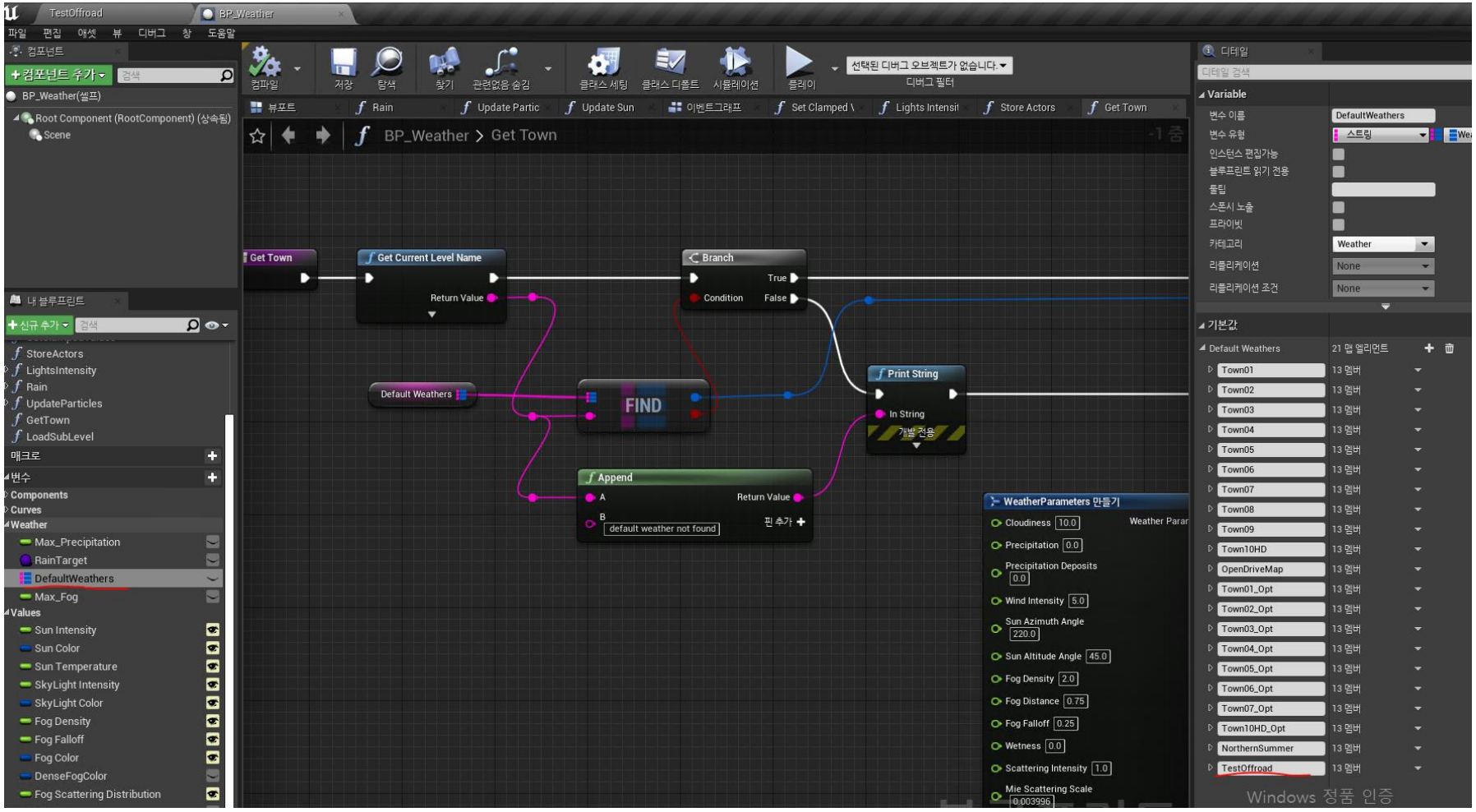
Add custom map (first method)

- Add custom map name (Carla/Blueprints/Weather/BP_Weather)



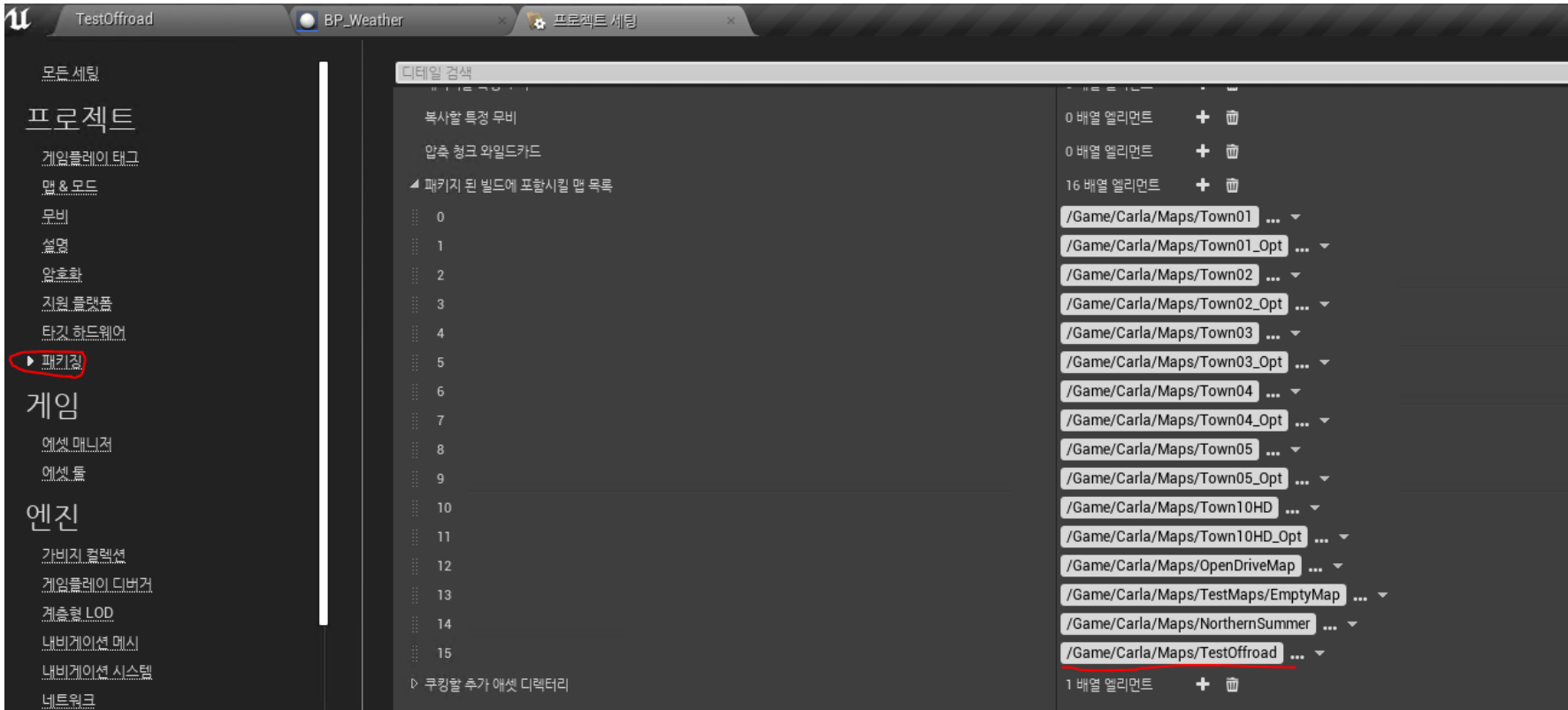
Add custom map (second method)

- Open project setting and go to packaging
- Open tab advanced and “list of maps to include in an packaged build”
- Add custom map (In this add TestOffroad)



Add custom map (second method)

- Open project setting and go to packaging
- Open tab advanced and “list of maps to include in an packaged build”
- Add custom map (In this add NorthernSummer.umap)
- Close unreal engine editor
- Run make package (in x64 Native Tools Command Prompt for VS 2019)



Add custom map (first method)

- Run test code (TestCustomMap.py for loading custom map)

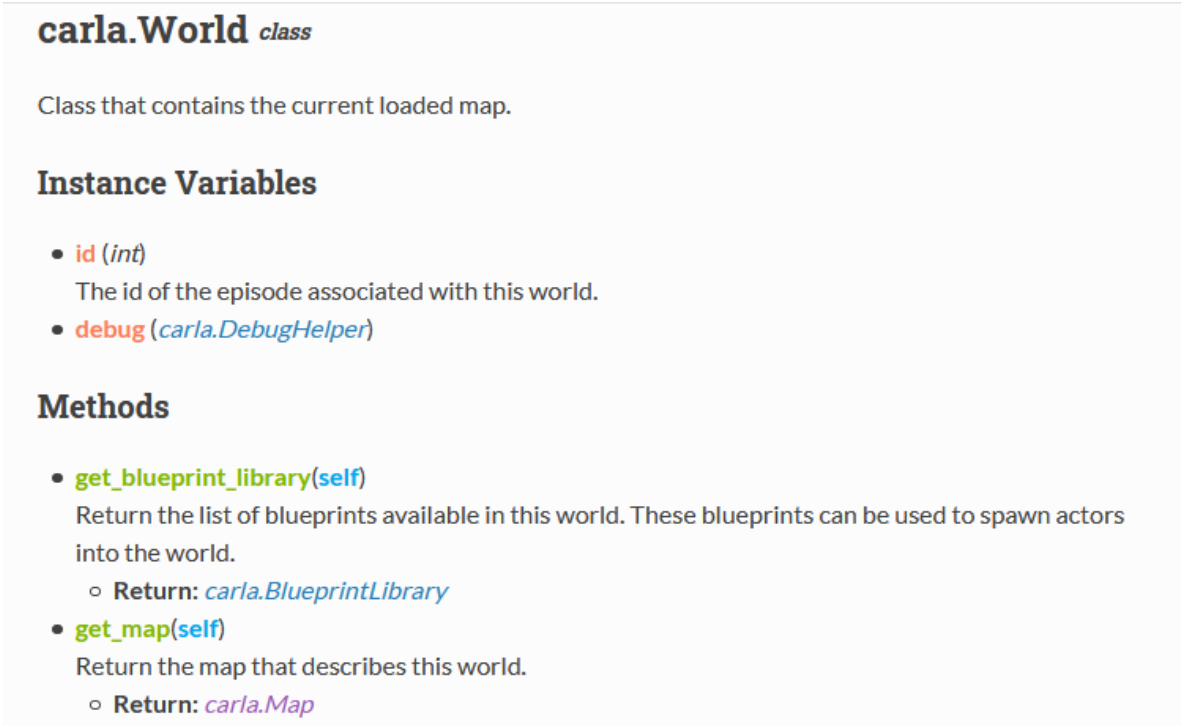
TestCustomMap.py 2, U

TestCustomMap.py > ...

```
24  # ==============================================
25
26
27  import carla
28
29  from carla import ColorConverter as cc
30
31  import argparse
32  import collections
33  import datetime
34  import logging
35  import math
36  import random
37  import re
38  import weakref
39  import time
40
41  client = carla.Client('127.0.0.1', 2000)
42  world = client.load_world("NorthernSummer")
43
44  time.sleep(5)
45
46  world = client.load_world("TestOffroad")
```

Caution

- Carla's API get_map( ) can not use in custom map because we does not use .xodr file
- So if you get object location, starting location and so on, you have to get directly in unreal engine
- In my paper, I get waypoint and etc… directly in unreal engine and apply python code
- Carla provide 'RoadRunner' and if you use it then you can create map and get .xodr file but 'RoadRunner' is hard to make off-road environment.



carla.World *class*

Class that contains the current loaded map.

Instance Variables

- id (*int*)
  The id of the episode associated with this world.
- debug (*carla.DebugHelper*)

Methods

- get_blueprint_library(self)
  Return the list of blueprints available in this world. These blueprints can be used to spawn actors into the world.
  - **Return:** *carla.BlueprintLibrary*
- get_map(self)
  Return the map that describes this world.
  - **Return:** *carla.Map*

Caution

- Carla's API get_map( ) can not use in custom map because we does not use .xodr file
- So if you get object location, starting location and so on, you have to get directly in unreal engine
- In my paper, I get waypoint and etc… directly in unreal engine and apply python code

Ubuntu

Build Carla

- Unfortunately, I don't use Ubuntu. So, I can't give you a detailed guide like Windows
- However, as I know, no problems building Carla (make launch) and PythonAPI(make PythonAPI) on Ubuntu
- Almost the same way in Windows and Ubuntu.
- First of all, you just follow official guide
- https://carla.readthedocs.io/en/latest/build_linux/

Add custom map

- If you finish building Carla(make launch) and PythonAPI(make PythonAPI), make map by using unreal engine.
- In this case, I create map on windows using unreal engine and move unreal engine project to Ubuntu
- And use asset migration (on Ubuntu)
- After migration, you follow step (p.16 ~ p.19)
- And make package