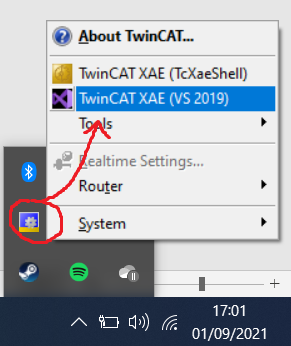
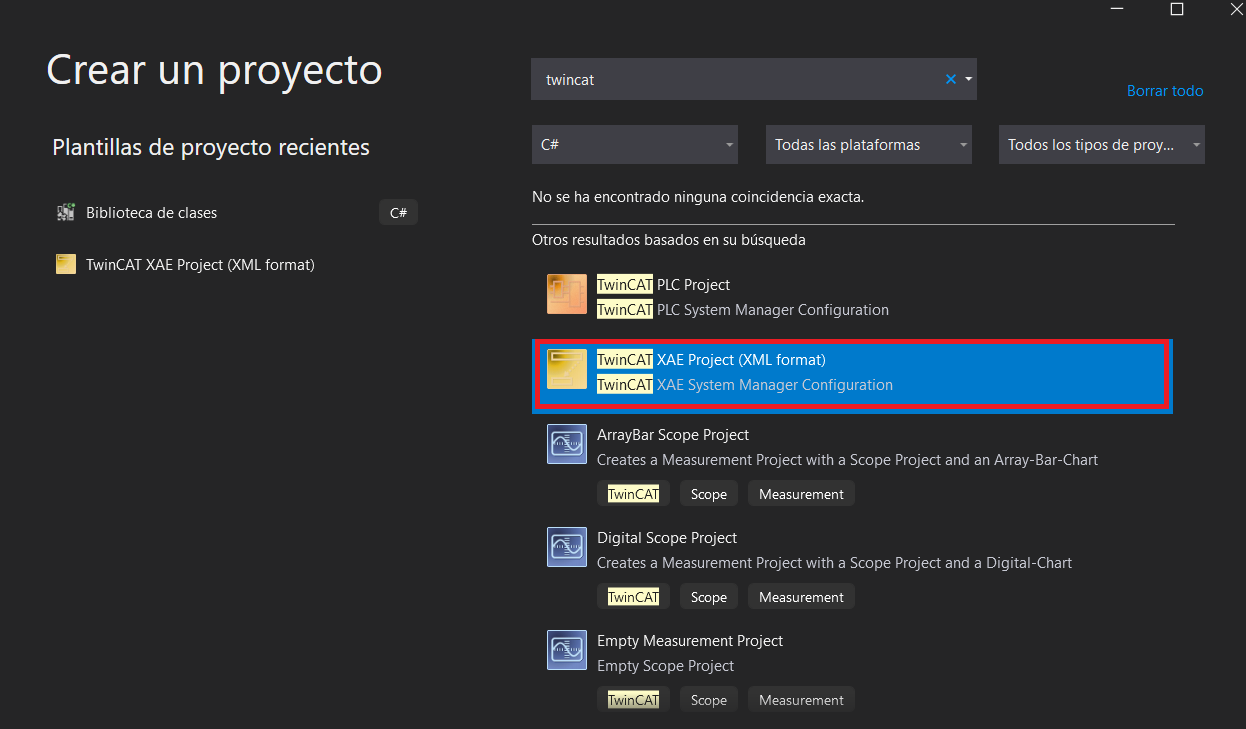
TWINCAT ADS - UNITY FACTORY PROJECT

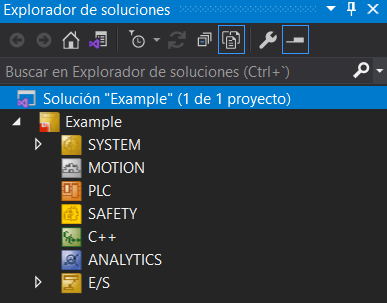
1. Twincat set up (page 2 to 3)
2. Unity set up (page 4 to 5)
3. My project´s scripts (page 6 to 11)
4. Explanations and bibliography (page 12)

1- TWINCAT SET UP

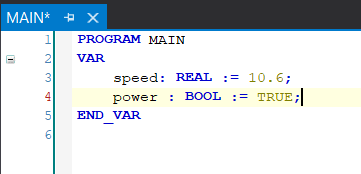
First of all we need to create a Twincat Simple PLC Project in visual studio. Follow these steps:

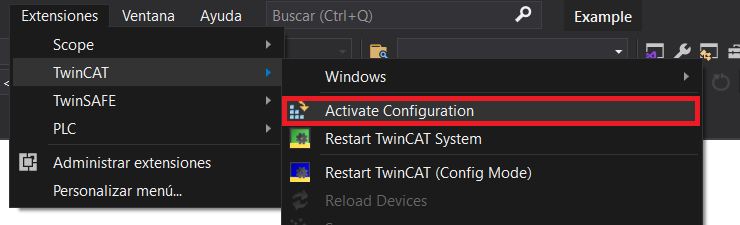


Then you will have something like this:



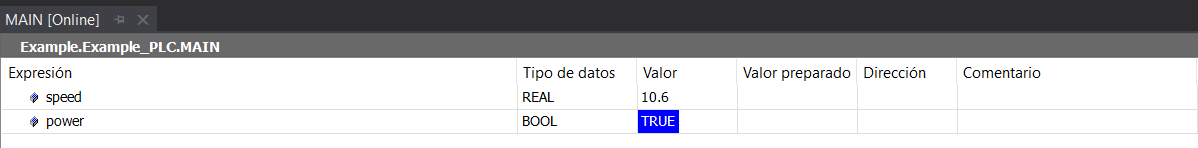
Now right-click on PLC, add new element, and create a standard PLC Project. After that, you have to create a POU (in my case I´ll use the MAIN) and create the variables you wish (you can add them later too) just like this:





Each time you create or modify a variable you have to update it by activating configuration (remember to start the twincat system too):

You can now run the PLC and go online to test it and you will see something like this:

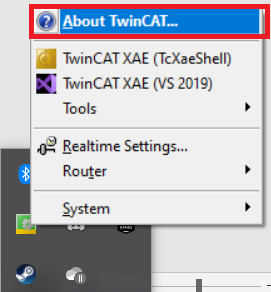
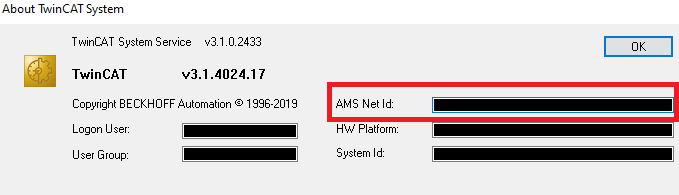


2- UNITY SET UP

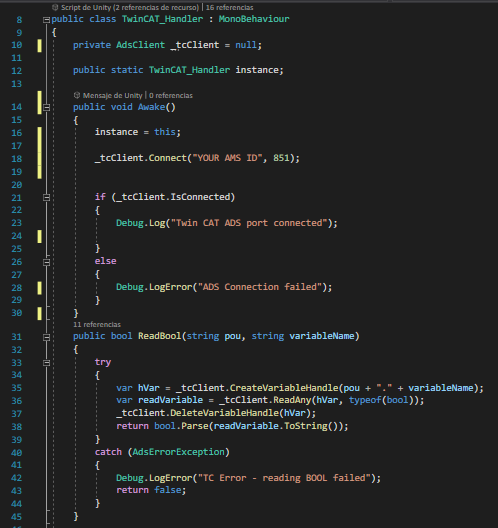
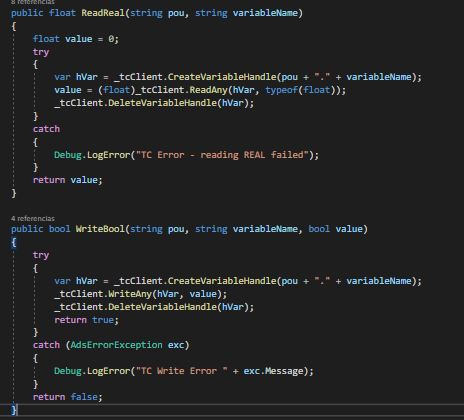
First of all you need to create a scene with al the gameobjects that you will need, in my case I have this scene, which is a simple conveyor belt:



Secondly, we need a handler that will allow us to communicate with the PLC. To do that, I created a script with a static instance so every script can access to it. All you need is the PLC port which is usually 851 and the AMS Net ID which can be seen here:



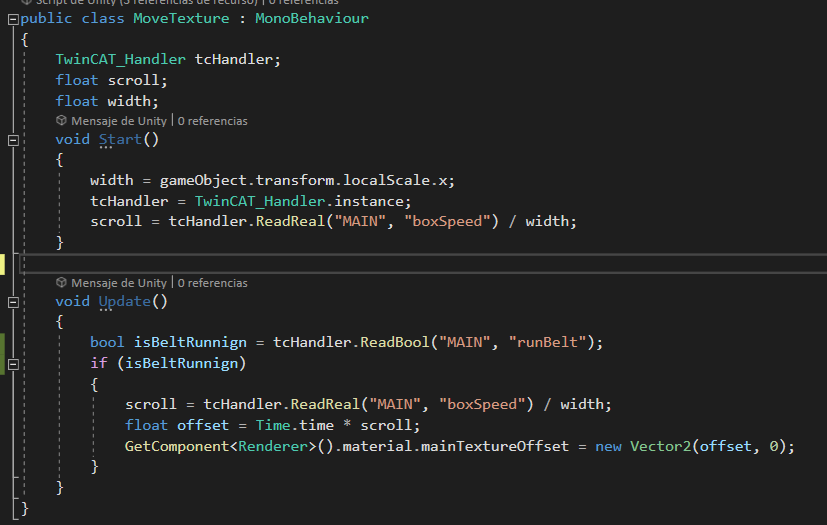
Here´s the script with some methods to read and write variables (if you want to create another method for other variable type, just copy-paste and change the method and variable type):

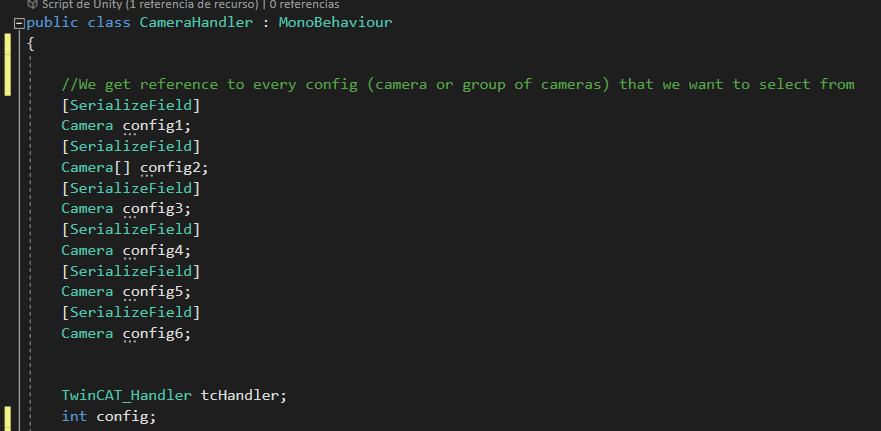


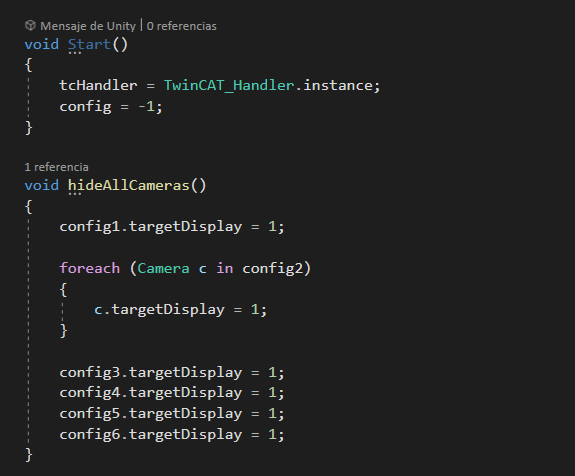
3- MY PROJECT´S SCRIPTS

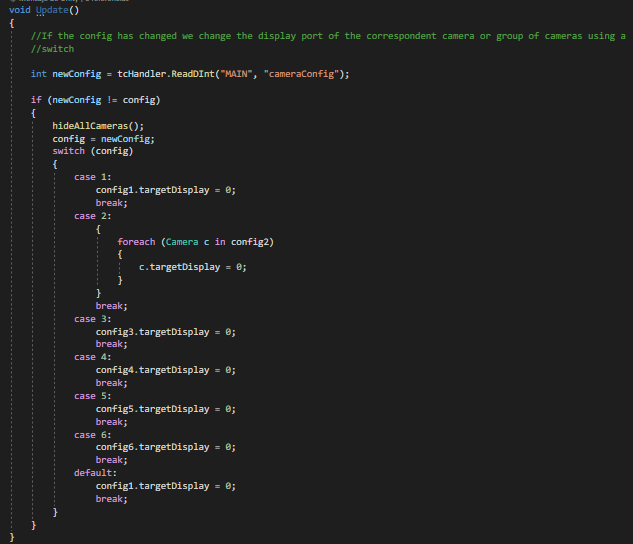
Now we now how to communicate twincat and unity so it´s time to show you some example of my scripts that use PLC variables.

MoveTexture: this scripts reads 2 variables from the PLC: runBelt to know if the textura must move, and boxSpeed to know the movement speed of the texture.



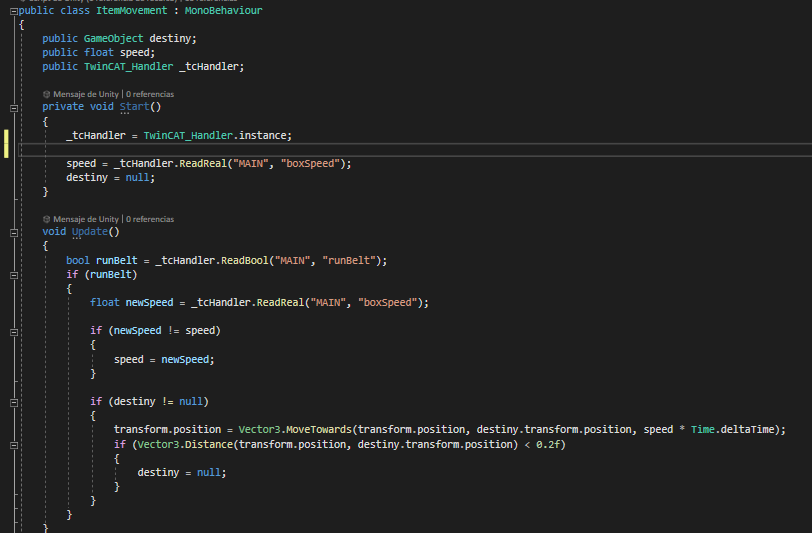
CameraHandler: this scripts reads the cameraConfig variable to know wich config it must use.



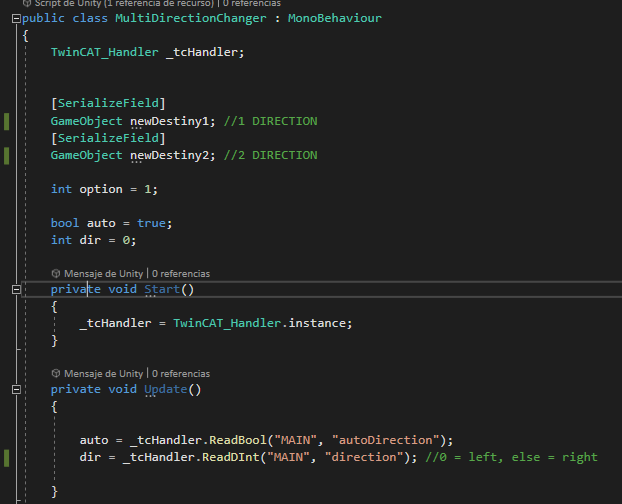


ItemMovement: this scripts reads 2 variables from the PLC: runBelt to know if the box must move, and boxSpeed to know the movement speed of the box.

It also has a reference to a gameObject (destiny) to move towards that point. There are triggers in the scene that changes the direction of the object.

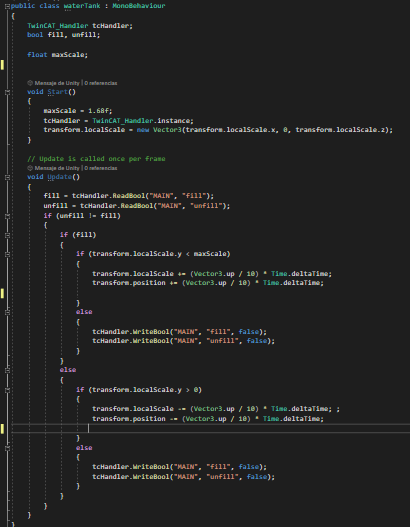


MultiDirectionChanger: this script reads the autoDirection and direction variables from the PLC. If autoDirection is true the script will select the left direction and the right direction alternatively. Otherwise, it will choose the direction given by direction variable.





WaterTank: it reads the unfill and fill variables from the PLC.



Spawner: it reads the spawnRate and runBelt variables from the PLC.



4- EXPLANATIONS AND BIBLIOGRAPHY

In this document I tried to be as   
clear and concise as I could but if you have any doubt about this document or you want to know more about other scripts that I didn´t show here, don´t hesitate to contact me to:

[ivsm18z@gmail.com](mailto:ivsm18z@gmail.com)

Bibliography:

All the information I learn to do this project is found in this website: http://dronefactory.co.uk/?p=43