Setting up CODESYS OPC UA (SP17 or higher)

Requirements

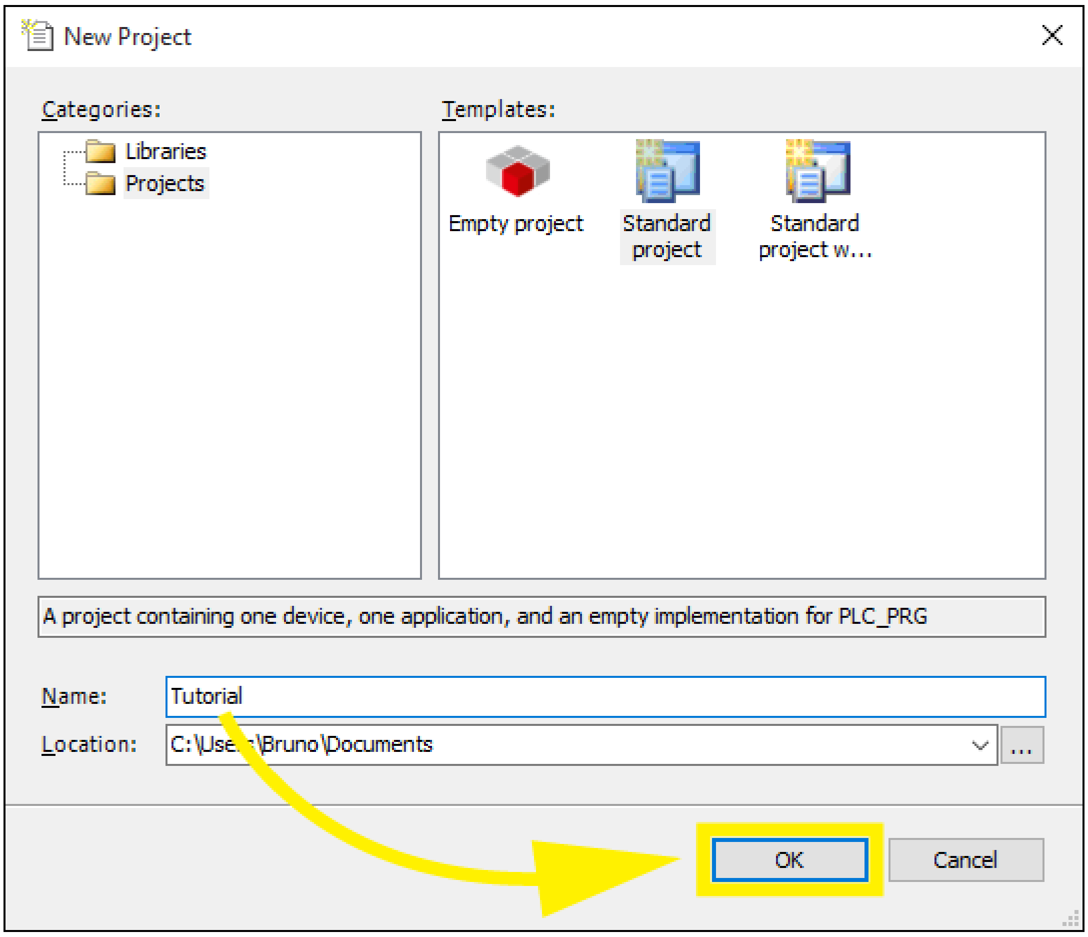
- Modbus & OPC Edition or Ultimate Edition
- CODESYS v3.5 SP17 or higher (works with demo version)

This tutorial shows how to use CODESYS together with Factory I/O through OPC UA. By following this sample you will create a new CODESYS project, configure it to work with Factory I/O and program the CODESYS Control Win (SoftPlc) to control the Sorting by Height (Advanced) scene.

The sample code used in this tutorial is based on the solutions found in the book Industrial Automation Practices.

Creating the Project

1. Start CODESYS and create a new project.

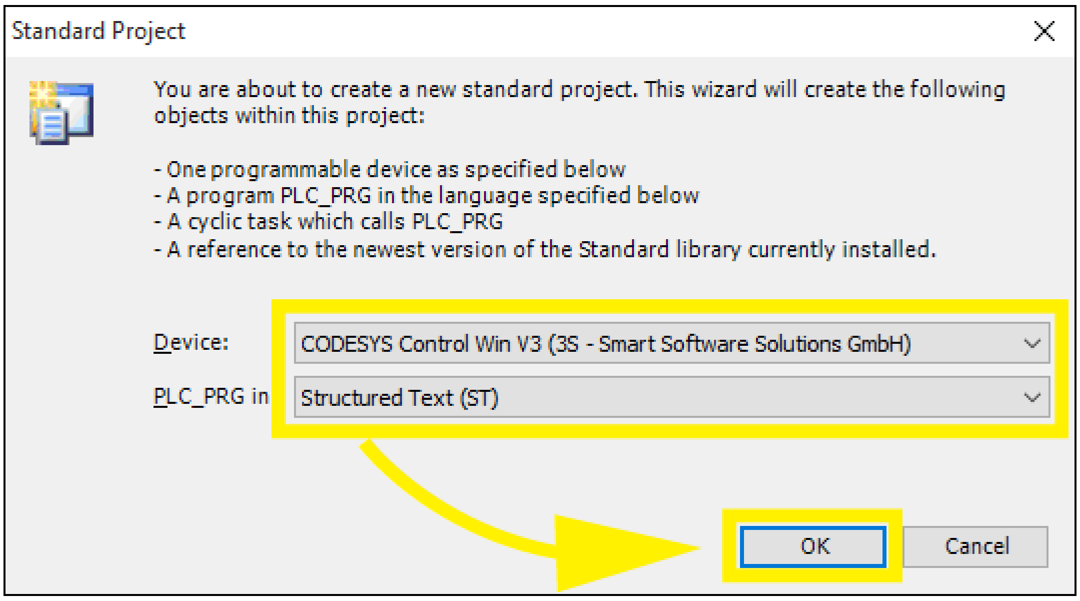


A project containing one device, one application, and an empty implementation for PLC_PRG

Name: Tutorial

Location: C:\Users\Bruno\Documents

2. Select **Standard project** from the Templates list and choose a name for the project (e.g. Tutorial). Click on **OK**.
3. On the Standard Project window select the Device **CODESYS Control Win V3 (3S - Smart Software Solutions GmbH)** and **Structured Text (ST)** for the **PLC_PRG**. Click on **OK**.



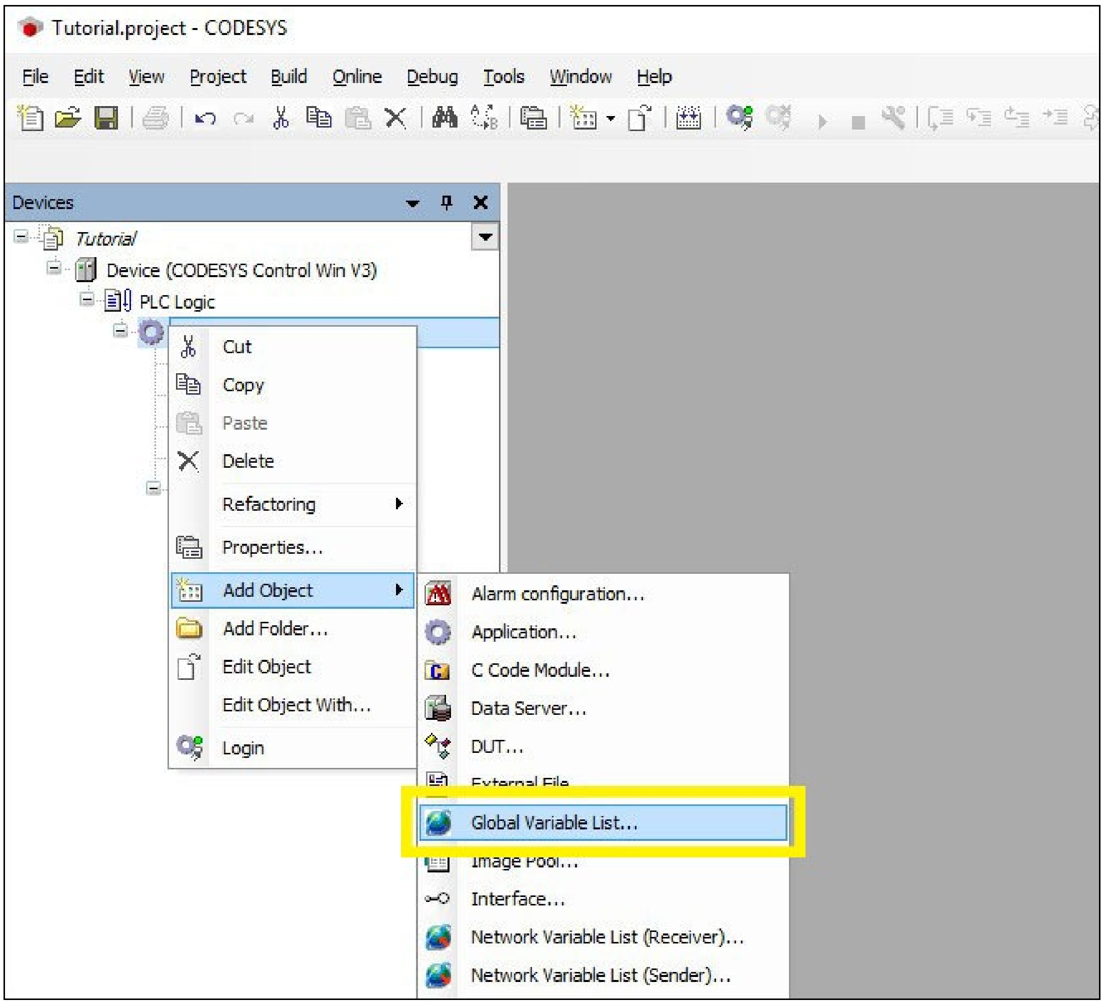
You are about to create a new standard project. This wizard will create the following objects within this project:

- One programmable device as specified below
- A program PLC_PRG in the language specified below
- A cyclic task which calls PLC_PRG
- A reference to the newest version of the Standard library currently installed.

Device: CODESYS Control Win V3 (3S - Smart Software Solutions GmbH)

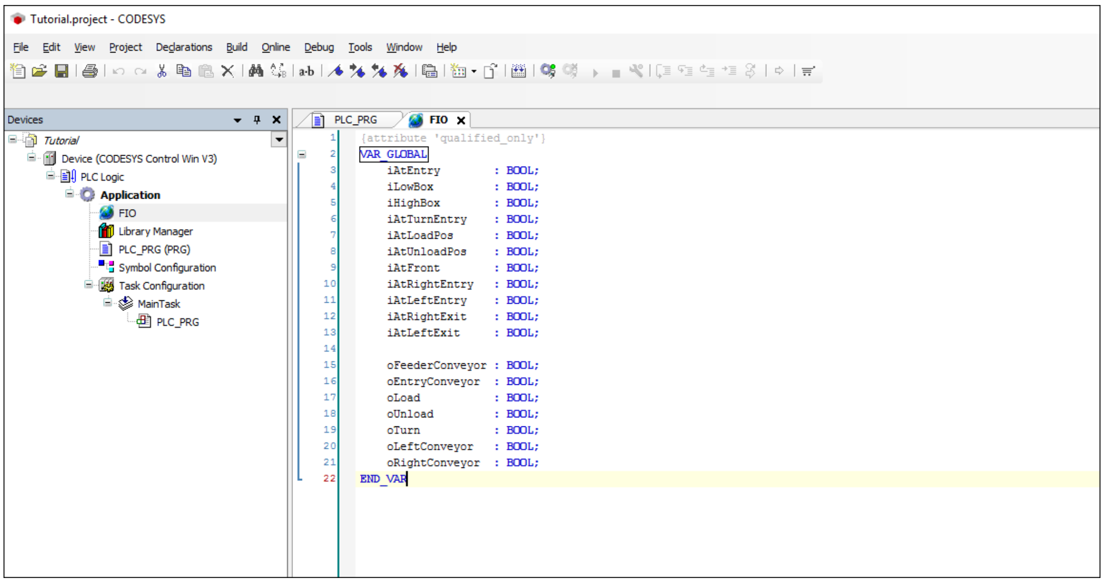
PLC_PRG in: Structured Text (ST)

4. Right-click on **Application** and select **Add Object** > **Global variable List...**. Type **FIO** as the list name, click on **Add**.



5. Open the FIO list by **Double Left-clicking** on it and copy and paste the following global variables. These variables will be used to exchange data between Factory I/O and CODESYS thought OPC UA (these are the I/O points).

```
VAR_GLOBAL
    iAtEntry          : BOOL;
    iLowBox           : BOOL;
    iHighBox          : BOOL;
    iAtTurnEntry      : BOOL;
    iAtLoadPos        : BOOL;
    iAtUnloadPos      : BOOL;
    iAtFront          : BOOL;
    iAtRightEntry     : BOOL;
    iAtLeftEntry      : BOOL;
    iAtRightExit      : BOOL;
    iAtLeftExit       : BOOL;

    oFeederConveyor : BOOL;
    oEntryConveyor  : BOOL;
    oLoad             : BOOL;
    oUnload           : BOOL;
    oTurn             : BOOL;
    oLeftConveyor   : BOOL;
    oRightConveyor  : BOOL;
END_VAR
```

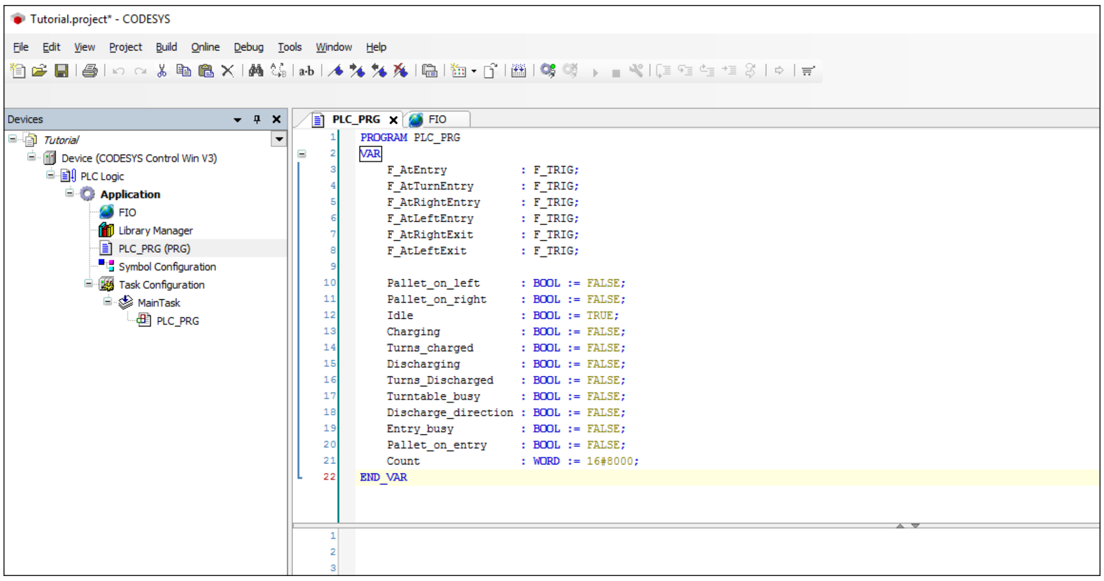


Tutorial.project - CODESYS

```
{attribute 'qualified_only'}
VAR_GLOBAL
    iAtEntry         : BOOL;
    iLowBox          : BOOL;
    iHighBox         : BOOL;
    iAtTurnEntry     : BOOL;
    iAtLoadPos       : BOOL;
    iAtUnloadPos     : BOOL;
    iAtFront         : BOOL;
    iAtRightEntry    : BOOL;
    iAtLeftEntry     : BOOL;
    iAtRightExit     : BOOL;
    iAtLeftExit      : BOOL;

    oFeederConveyor : BOOL;
    oEntryConveyor  : BOOL;
    oLoad           : BOOL;
    oUnload         : BOOL;
    oTurn           : BOOL;
    oLeftConveyor   : BOOL;
    oRightConveyor  : BOOL;
END_VAR
```

6. On the Devices tree **Double Left-click** on **PLC_PRG (PRG)** and copy and paste the following variables. These are the variables you will be using in your program.

```
PROGRAM PLC_PRG
VAR
    F_AtEntry           : F_TRIG;
    F_AtTurnEntry       : F_TRIG;
    F_AtRightEntry      : F_TRIG;
    F_AtLeftEntry       : F_TRIG;
    F_AtRightExit       : F_TRIG;
    F_AtLeftExit        : F_TRIG;

    Pallet_on_left      : BOOL := FALSE;
    Pallet_on_right     : BOOL := FALSE;
    Idle                : BOOL := TRUE;
    Charging            : BOOL := FALSE;
    Turns_charged       : BOOL := FALSE;
    Discharging         : BOOL := FALSE;
    Turns_Discharged    : BOOL := FALSE;
    Turntable_busy      : BOOL := FALSE;
    Discharge_direction : BOOL := FALSE;
    Entry_busy          : BOOL := FALSE;
    Pallet_on_entry     : BOOL := FALSE;
    Count               : WORD := 16#8000;
END_VAR
```



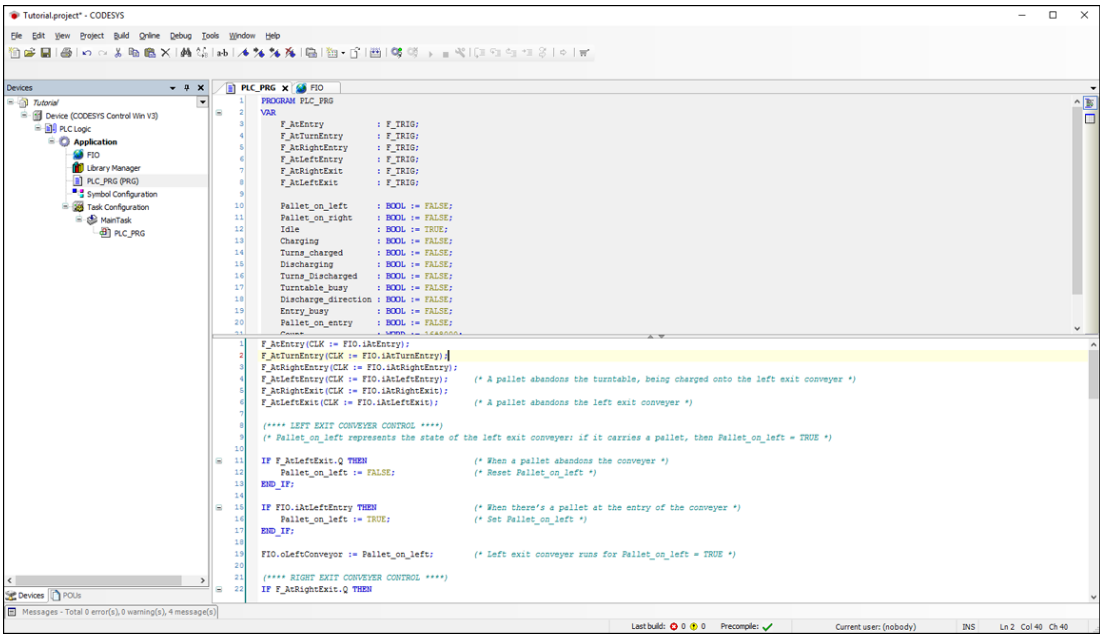
```
PROGRAM PLC_PRG
VAR
    F_AtEntry           : F_TRIG;
    F_AtTurnEntry       : F_TRIG;
    F_AtRightEntry      : F_TRIG;
    F_AtLeftEntry       : F_TRIG;
    F_AtRightExit       : F_TRIG;
    F_AtLeftExit        : F_TRIG;

    Pallet_on_left      : BOOL := FALSE;
    Pallet_on_right     : BOOL := FALSE;
    Idle                : BOOL := TRUE;
    Charging            : BOOL := FALSE;
    Turns_charged       : BOOL := FALSE;
    Discharging         : BOOL := FALSE;
    Turns_Discharged    : BOOL := FALSE;
    Turntable_busy      : BOOL := FALSE;
    Discharge_direction : BOOL := FALSE;
    Entry_busy          : BOOL := FALSE;
    Pallet_on_entry     : BOOL := FALSE;
    Count               : WORD := 16#8000;
END_VAR
```

7. Now, copy and paste the following code. This is the program that will control the **Sorting by Height (Advanced)** scene.

```
F_AtEntry(CLK := FIO.iAtEntry);
F_AtTurnEntry(CLK := FIO.iAtTurnEntry);
F_AtRightEntry(CLK := FIO.iAtRightEntry);
F_AtLeftEntry(CLK := FIO.iAtLeftEntry);      (* A pallet abandons the
turntable, being charged onto the left exit conveyer *)
F_AtRightExit(CLK := FIO.iAtRightExit);
F_AtLeftExit(CLK := FIO.iAtLeftExit);        (* A pallet abandons the left
exit conveyer *)

(**** LEFT EXIT CONVEYER CONTROL ****)
(* Pallet_on_left represents the state of the left exit conveyer: if it
carries a pallet, then Pallet_on_left = TRUE *)

IF F_AtLeftExit.Q THEN                       (* When a pallet abandons the
conveyer *)
    Pallet_on_left := FALSE;                 (* Reset Pallet_on_left *)
END_IF;

IF FIO.iAtLeftEntry THEN                     (* When there's a pallet at the
entry of the conveyer *)
    Pallet_on_left := TRUE;                  (* Set Pallet_on_left *)
END_IF;

FIO.oLeftConveyor := Pallet_on_left;         (* Left exit conveyer runs for
Pallet_on_left = TRUE *)

(**** RIGHT EXIT CONVEYER CONTROL ****)

IF F_AtRightExit.Q THEN
    Pallet_on_right := FALSE;
END_IF;

IF FIO.iAtRightEntry THEN
    Pallet_on_right := TRUE;
END_IF;
```

```
FIO.oRightConveyor := Pallet_on_right;

(**** TURNTABLE CONTROL ****)

IF Idle AND FIO.iAtTurnEntry THEN
    Idle := FALSE;
    Charging := TRUE;
END_IF;

IF Charging AND FIO.iAtFront THEN
    Charging := FALSE;
    Turns_charged := TRUE;
END_IF;

IF Turns_charged AND FIO.iAtUnloadPos THEN
    Turns_charged := FALSE;
    Discharging := TRUE;
END_IF;

(* The turntable discharges now onto both conveyers. Thus: *)

IF Discharging AND (F_AtRightEntry.Q OR F_AtLeftEntry.Q) THEN
    Discharging := FALSE;
    Turns_Discharged := TRUE;
END_IF;

IF Turns_Discharged AND FIO.iAtLoadPos THEN
    Turns_Discharged := FALSE;
    Idle := TRUE;
END_IF;

IF F_AtTurnEntry.Q THEN
    Turntable_busy := TRUE;
END_IF;

IF Idle THEN
    Turntable_busy := FALSE;
END_IF;

(* Computing control outputs according to the current state of the
turntable and the direction of the discharge *)
FIO.oLoad := Charging OR Discharging AND Discharge_direction;

(* oUnload is TRUE if the discharge is onto the left conveyer *)
FIO.oUnload := Discharging AND NOT Discharge_direction;

(* oTurn is TRUE if the discharge is onto the right conveyer *)
FIO.oTurn := Turns_charged OR Discharging;

(**** ENTRY CONVEYER CONTROL ****)
IF F_AtEntry.Q THEN
    Count := ROL (Count,1);
    IF (Count = WORD#16#2) THEN
        Entry_busy := TRUE;
    END_IF;
END_IF;
```

```
IF F_AtTurnEntry.Q THEN
    (** Defining the direction of the discharge: the direction of the
discharge changes every time a pallet is
    discharged from the entry conveyer onto the turntable. If
Discharge_direction = FALSE, then the discharge is onto the right conveyer
**)
    Discharge_direction := NOT Discharge_direction;

    Entry_busy := FALSE;
    Count := ROR(Count,1);

    IF (Count = WORD#16#8000) THEN
        Pallet_on_entry := FALSE;
    END_IF;
END_IF;

IF FIO.iAtEntry THEN
    Pallet_on_entry := TRUE;
END_IF;

FIO.oEntryConveyor := Pallet_on_entry AND (NOT Turntable_busy OR NOT
FIO.iAtTurnEntry);

(**** FEEDING CONVEYER CONTROL ****)
FIO.oFeederConveyor := NOT Entry_busy OR NOT FIO.iAtEntry;
```



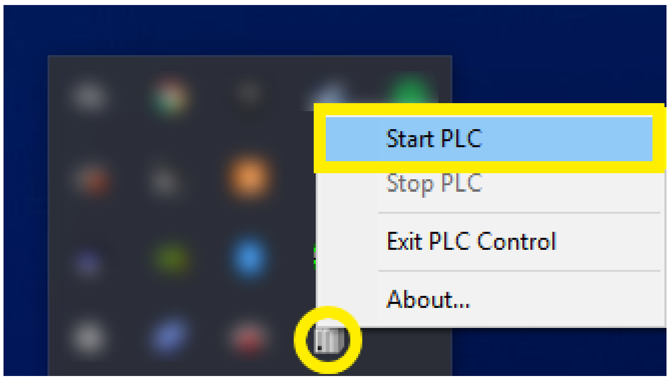
```
PROGRAM PLC_PRG
VAR
    F_AtEntry           : F_TRIG;
    F_AtTurnEntry       : F_TRIG;
    F_AtRightEntry      : F_TRIG;
    F_AtLeftEntry       : F_TRIG;
    F_AtRightExit       : F_TRIG;
    F_AtLeftExit        : F_TRIG;

    Pallet_on_left      : BOOL := FALSE;
    Pallet_on_right     : BOOL := FALSE;
    Idle                : BOOL := TRUE;
    Charging            : BOOL := FALSE;
    Turns_charged       : BOOL := FALSE;
    Discharging         : BOOL := FALSE;
    Turns_Discharged    : BOOL := FALSE;
    Turntable_busy      : BOOL := FALSE;
    Discharge_direction : BOOL := FALSE;
    Entry_busy          : BOOL := FALSE;
    Pallet_on_entry     : BOOL := FALSE;
```

```
F_AtEntry(CLK := FIO.iAtEntry);
F_AtTurnEntry(CLK := FIO.iAtTurnEntry);
F_AtRightEntry(CLK := FIO.iAtRightEntry);
F_AtLeftEntry(CLK := FIO.iAtLeftEntry);     (* A pallet abandons the turntable, being charged onto the left exit conveyer *)
F_AtRightExit(CLK := FIO.iAtRightExit);
F_AtLeftExit(CLK := FIO.iAtLeftExit);       (* A pallet abandons the left exit conveyer *)

(**** LEFT EXIT CONVEYER CONTROL ****)
(* Pallet_on_left represents the state of the left exit conveyer: if it carries a pallet, then Pallet_on_left = TRUE *)

IF F_AtLeftExit.Q THEN                      (* When a pallet abandons the conveyer *)
    Pallet_on_left := FALSE;                (* Reset Pallet_on_left *)
END_IF;

IF FIO.iAtLeftEntry THEN                    (* When there's a pallet at the entry of the conveyer *)
    Pallet_on_left := TRUE;                 (* Set Pallet_on_left *)
END_IF;

FIO.oLeftConveyor := Pallet_on_left;        (* Left exit conveyer runs for Pallet_on_left = TRUE *)

(**** RIGHT EXIT CONVEYER CONTROL ****)
IF F_AtRightExit.Q THEN
```

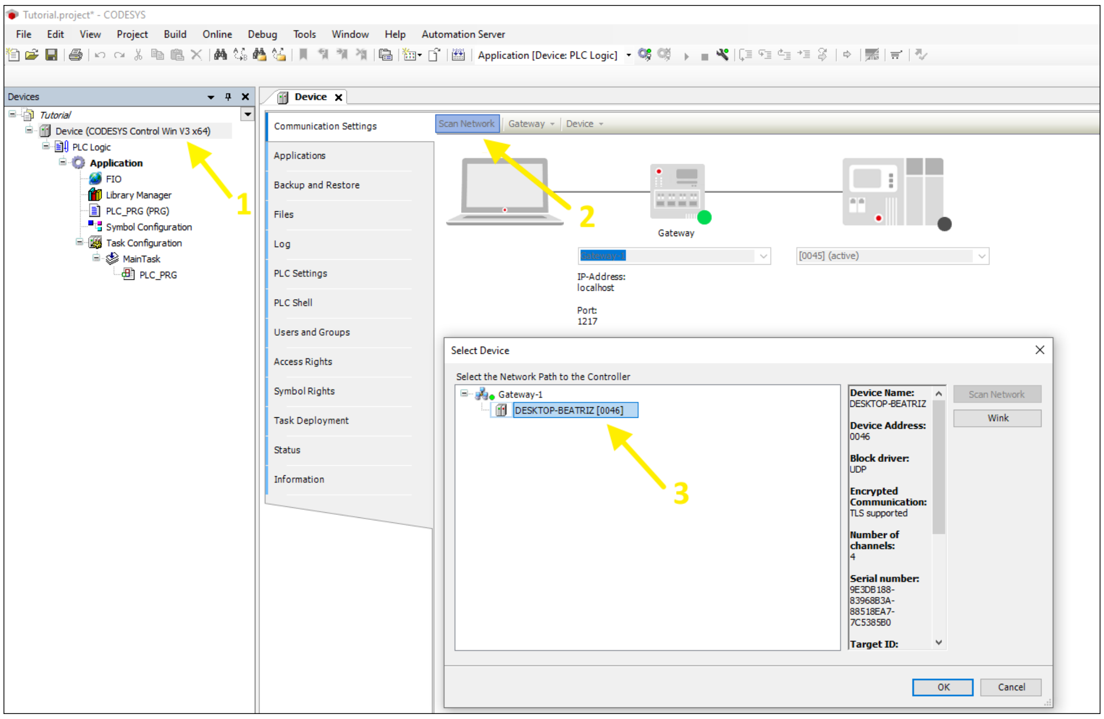
Messages - Total 0 error(s), 0 warning(s), 4 message(s)

Last build: 0 0 Precompile: ✓ Current user: (nobody) INS Ln 2 Col 40 Ch 40

8. **Right-click** on CODESYS Control Win PLC icon (Systray) and select **Start PLC**.



9. Get back to CODESYS and on the project tree, **Double Left-click** on **Device (CODESYS Control Win V3)** and then on **Communication Settings**. Now, click on **Scan network...** Select the network path to the controller and click on **OK**. Note that the device name can be changed by clicking on **Device** > **Rename active device**.



Gateway-1
[0045] (active)

IP-Address:
localhost

Port:
1217

Select the Network Path to the Controller

Gateway-1
DESKTOP-BEATRIZ [0046]

Device Name:
DESKTOP-BEATRIZ

Device Address:
0046

Block driver:
UDP

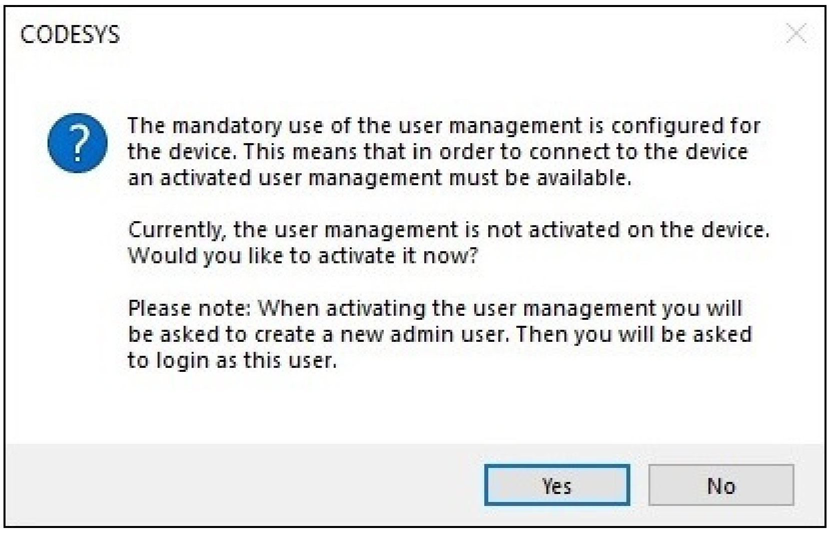
Encrypted Communication:
TLS supported

Number of channels:
4

Serial number:
9E3DB188-83968B3A-88518EA7-7C5385B0

Target ID:

10. After selecting your device, CODESYS will prompt you regarding the mandatory use of the user management, click on **Yes**.

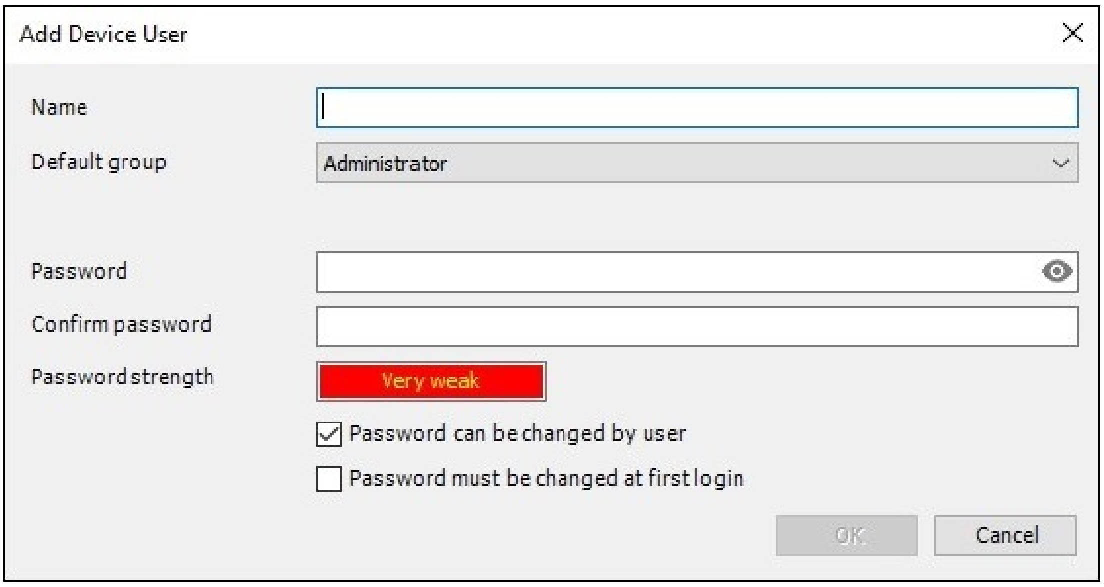

CODESYS

The mandatory use of the user management is configured for the device. This means that in order to connect to the device an activated user management must be available.

Currently, the user management is not activated on the device. Would you like to activate it now?

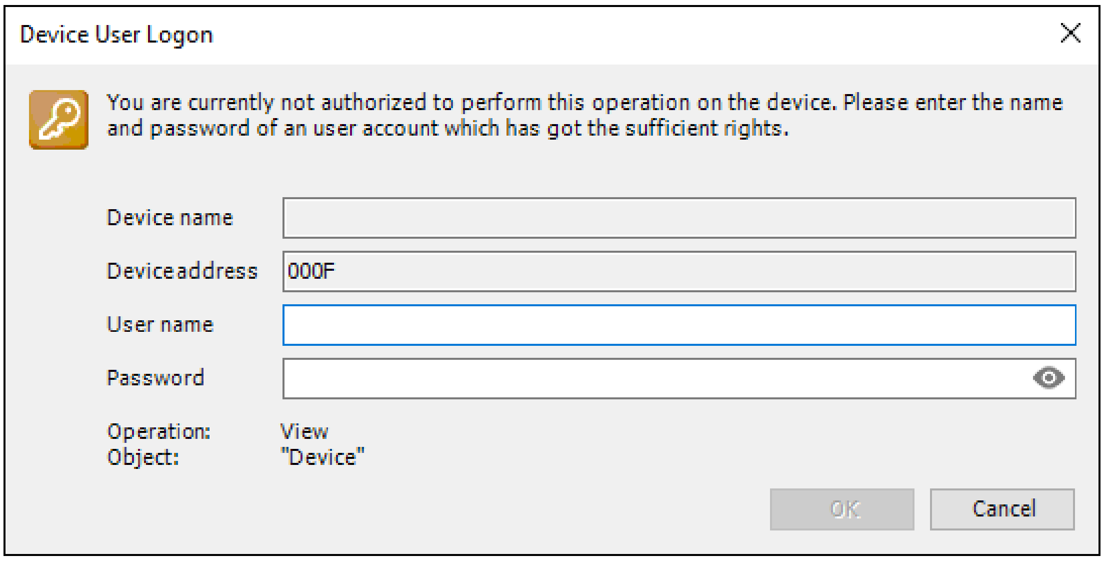
Please note: When activating the user management you will be asked to create a new admin user. Then you will be asked to login as this user.

11. Now, CODESYS will require you to create a Device User. Define a Name and a Password and click **OK**.


Add Device User

Name
Default group: Administrator
Password
Confirm password
Password strength: Very weak
[✓] Password can be changed by user
[ ] Password must be changed at first login

12. Next, you will have to Logon using the credentials defined in the previous step.

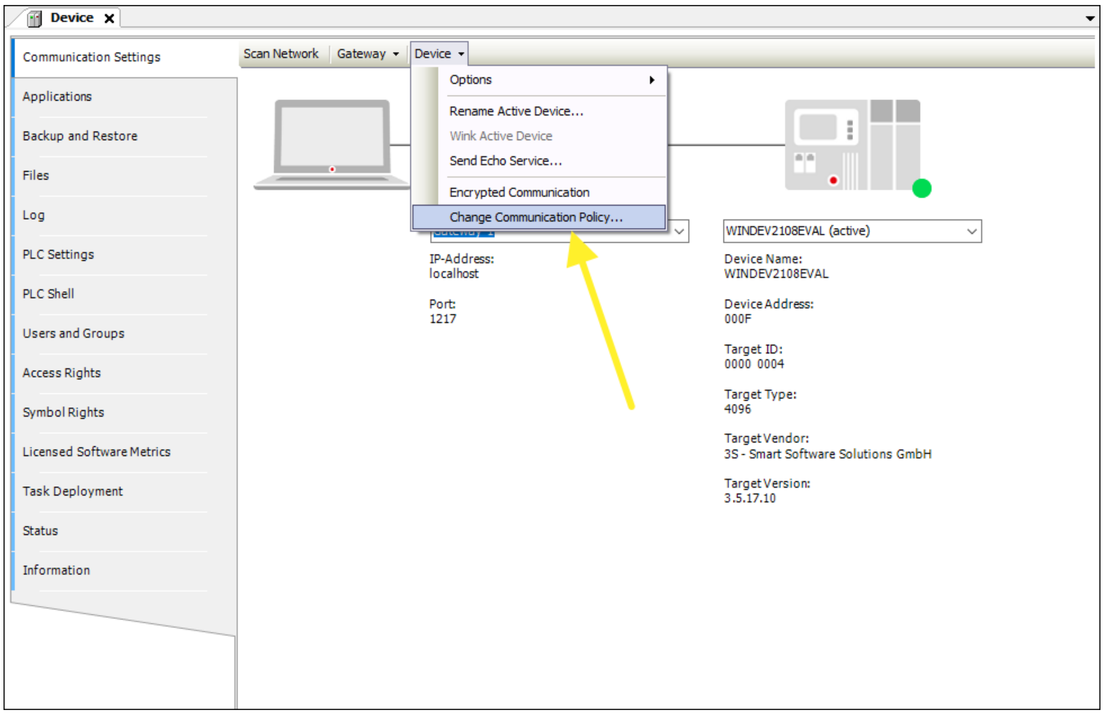

Device User Logon

You are currently not authorized to perform this operation on the device. Please enter the name and password of an user account which has got the sufficient rights.

Device name	
Deviceaddress	000F
User name	
Password	

Operation: View
Object: "Device"

13. In order for CODESYS to communicate with Factory I/O you have to disable the Device User Management. For that, **Double Left-click** on **Device (CODESYS Control Win V3)** and then on **Communication Settings**. Now, click on **Device** > **Change Communication Policy...**



IP-Address:
localhost

Port:
1217

Device Name:
WINDEV2108EVAL

Device Address:
000F

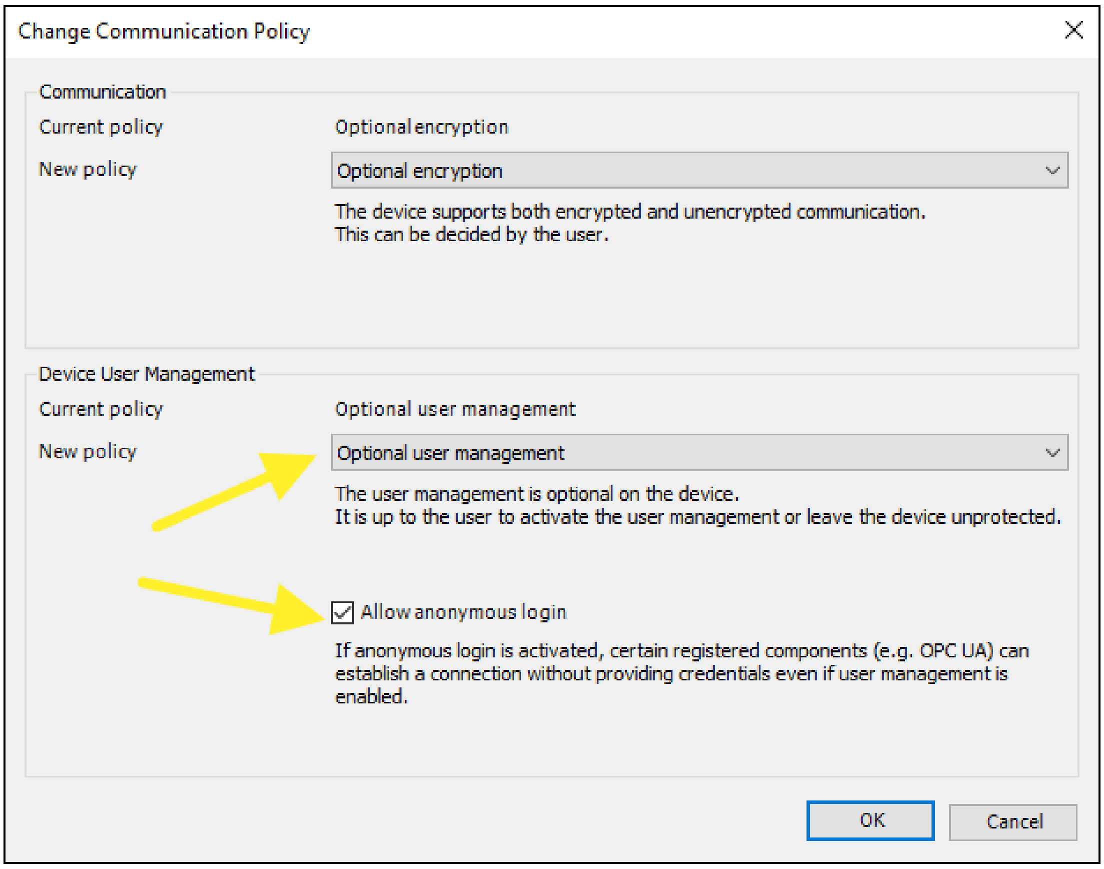
Target ID:
0000 0004

Target Type:
4096

Target Vendor:
3S - Smart Software Solutions GmbH

Target Version:
3.5.17.10

14. In the **Device User Management** section, in the drop-down list, select **Optional user management** and check the **Allow anonymous login** option.



Change Communication Policy

Communication

Current policy: Optional encryption

New policy: Optional encryption

The device supports both encrypted and unencrypted communication.
This can be decided by the user.

Device User Management

Current policy: Optional user management

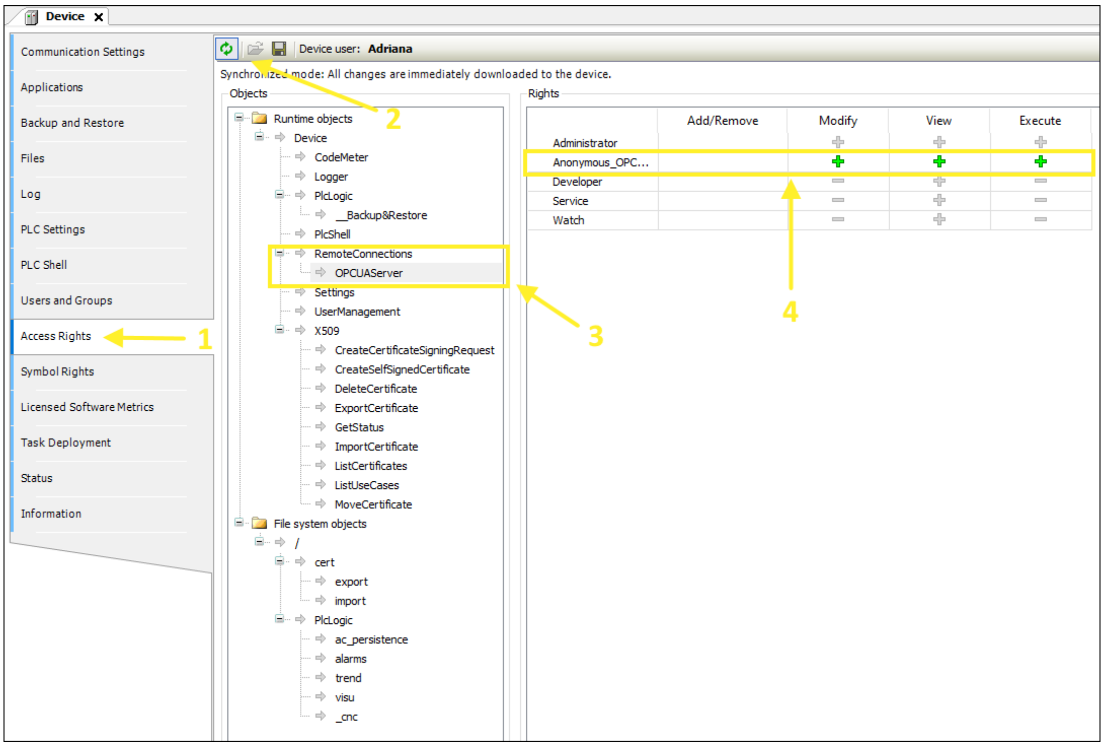
New policy: Optional user management

The user management is optional on the device.
It is up to the user to activate the user management or leave the device unprotected.

☑ Allow anonymous login

If anonymous login is activated, certain registered components (e.g. OPC UA) can establish a connection without providing credentials even if user management is enabled.

15. Click on **Access Rights (1)** and next on the **Synchronization button (2)**. Next, open the **RemoteConnections (3)** and select the **OPCUAServer**. On the **Anonymous_OPCUAServer (4)** allow the **Modify**, **View** and **Execute** by clicking on each one of them.



Device user: **Adriana**

Synchronized mode: All changes are immediately downloaded to the device.

Objects

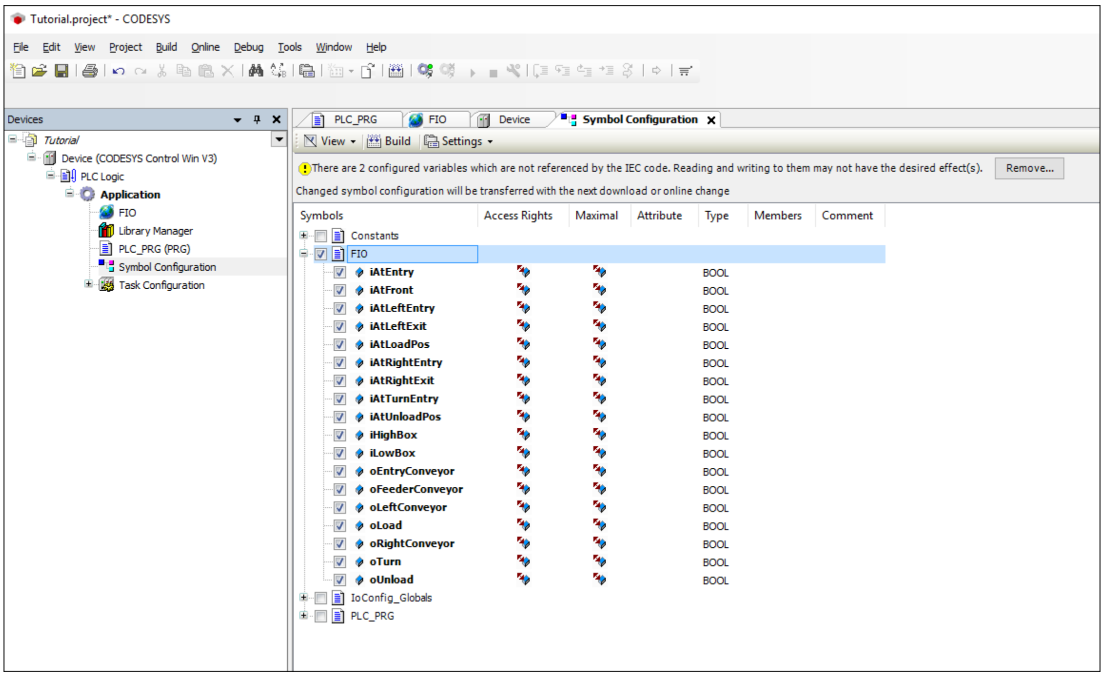
```
Runtime objects
  Device
    CodeMeter
    Logger
    PlcLogic
      __Backup&Restore
    PlcShell
    RemoteConnections
      OPCUAServer
    Settings
    UserManagement
    X509
      CreateCertificateSigningRequest
      CreateSelfSignedCertificate
      DeleteCertificate
      ExportCertificate
      GetStatus
      ImportCertificate
      ListCertificates
      ListUseCases
      MoveCertificate
File system objects
  /
    cert
      export
      import
    PlcLogic
      ac_persistence
      alarms
      trend
      visu
      _cnc
```

Rights

	Add/Remove	Modify	View	Execute
Administrator		+	+	+
Anonymous_OPC...		+	+	+
Developer		−	+	−
Service		−	+	−
Watch		−	+	−

16. In the toolbar click on **Build** > **Build** (F11) to download the program to SoftPLC.

17. Now, **Right-click** on **Application** and select **Add Object** > **Symbol Configuration**. On the **Add Symbol Configuration** window check **Support OPC UA features**, next click on **Add**. Now, check the FIO symbols (see image below) and click on **Build**.



There are 2 configured variables which are not referenced by the IEC code. Reading and writing to them may not have the desired effect(s).

Changed symbol configuration will be transferred with the next download or online change

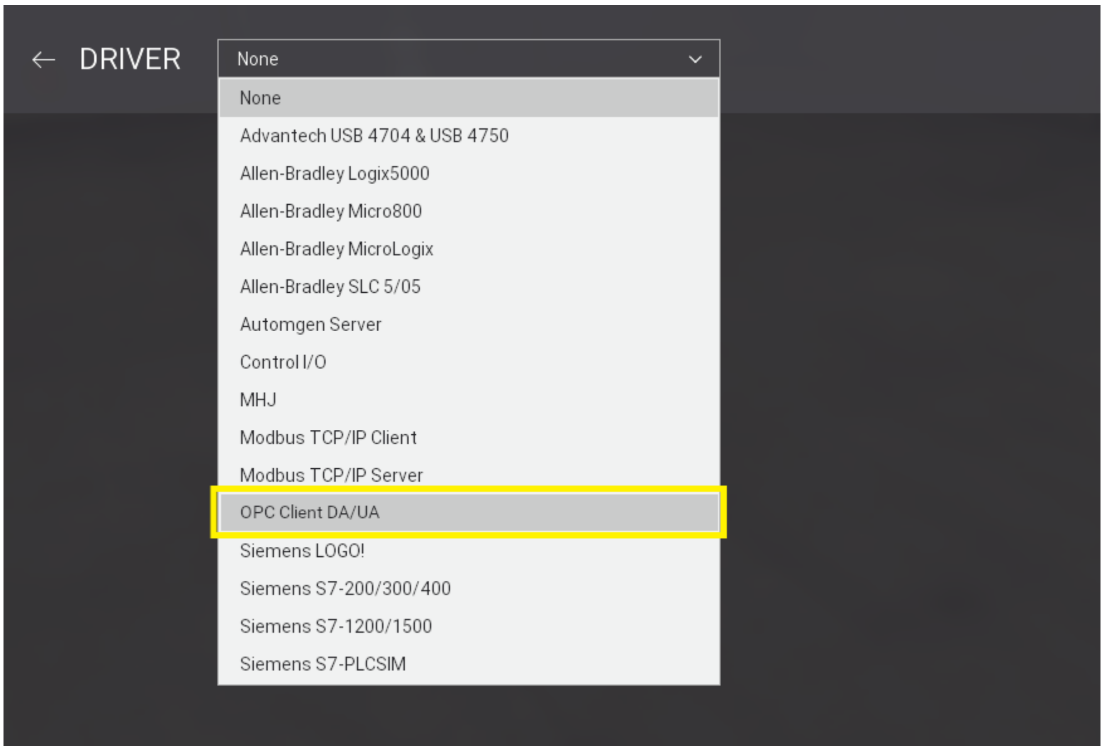
Symbols	Access Rights	Maximal	Attribute	Type	Members	Comment
Constants						
FIO						
iAtEntry				BOOL		
iAtFront				BOOL		
iAtLeftEntry				BOOL		
iAtLeftExit				BOOL		
iAtLoadPos				BOOL		
iAtRightEntry				BOOL		
iAtRightExit				BOOL		
iAtTurnEntry				BOOL		
iAtUnloadPos				BOOL		
iHighBox				BOOL		
iLowBox				BOOL		
oEntryConveyor				BOOL		
oFeederConveyor				BOOL		
oLeftConveyor				BOOL		
oLoad				BOOL		
oRightConveyor				BOOL		
oTurn				BOOL		
oUnload				BOOL		
IoConfig_Globals						
PLC_PRG						

18. Next, click on **Online** > **Login** (Alt + F8). When prompted to download the program to the PLC, click on **Yes**.

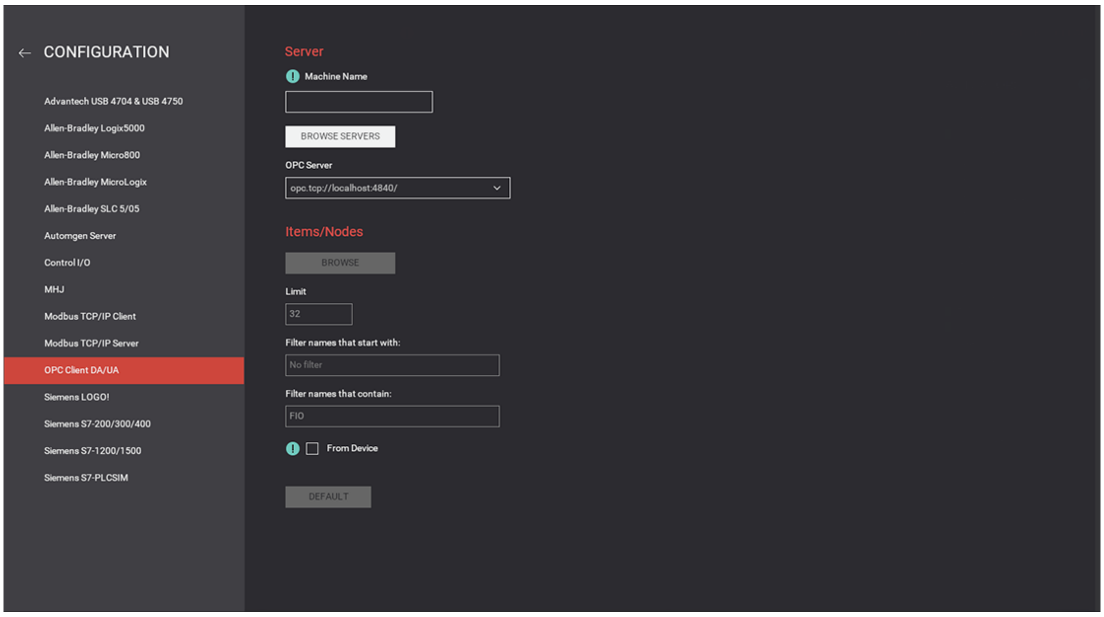
19. Finally, click on **Debug** > **Start** (F5).

Setting up Factory I/O

1. Open the Sorting by Height (Advanced) scene and click on **File** > **Driver Configuration**.
2. Select **OPC Client DA/UA** from the Driver list and click on **CONFIGURATION**.



3. Type **opc.tcp://localhost:4840** on the **OPC Server** input field and press `Enter ⏎` to connect. Set "FIO" as the contain filter (see image below), this will retrieve nodes containing "FIO" (the Global Variable List defined in CODESYS). Next, click on **BROWSE**.



OPC Server: opc.tcp://localhost:4840/

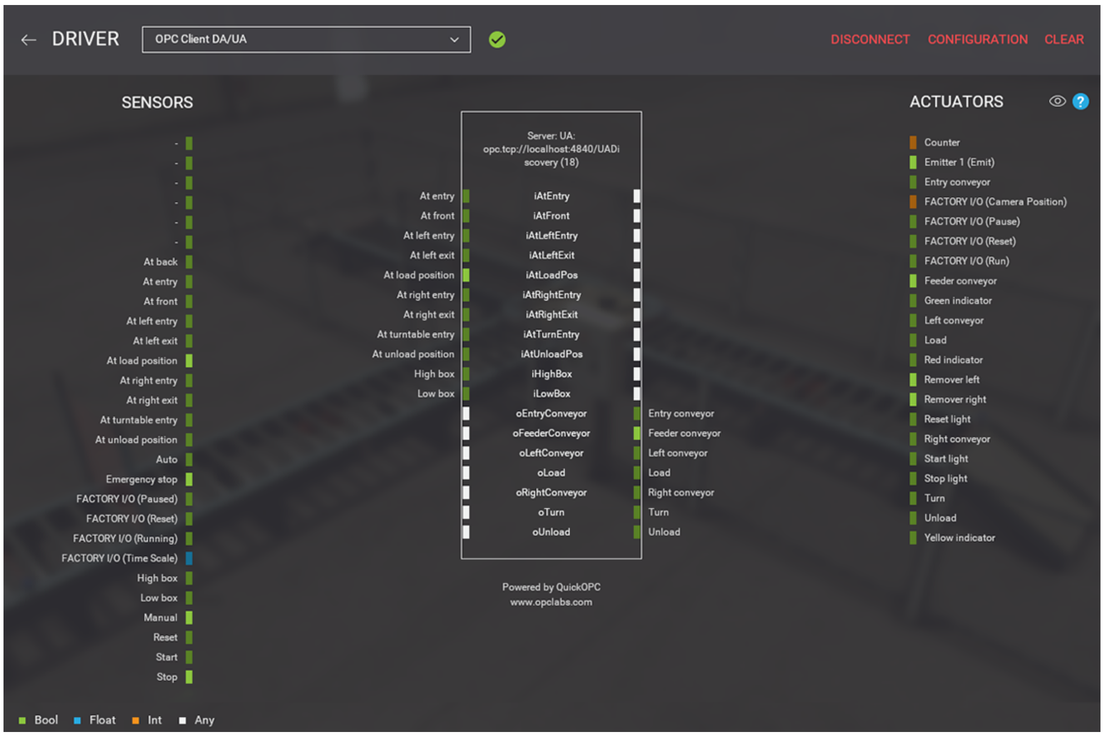
Limit: 32

Filter names that start with: No filter

Filter names that contain: FIO

☐ From Device

4. Map the sensors and actuators tags as shown in the image below.



SENSORS

-
-
-
-
-
-
At back
At entry
At front
At left entry
At left exit
At load position
At right entry
At right exit
At turntable entry
At unload position
Auto
Emergency stop
FACTORY I/O (Paused)
FACTORY I/O (Reset)
FACTORY I/O (Running)
FACTORY I/O (Time Scale)
High box
Low box
Manual
Reset
Start
Stop

Server: UA:
opc.tcp://localhost:4840/UADiscovery (18)

At entry	iAtEntry	
At front	iAtFront	
At left entry	iAtLeftEntry	
At left exit	iAtLeftExit	
At load position	iAtLoadPos	
At right entry	iAtRightEntry	
At right exit	iAtRightExit	
At turntable entry	iAtTurnEntry	
At unload position	iAtUnloadPos	
High box	iHighBox	
Low box	iLowBox	
	oEntryConveyor	Entry conveyor
	oFeederConveyor	Feeder conveyor
	oLeftConveyor	Left conveyor
	oLoad	Load
	oRightConveyor	Right conveyor
	oTurn	Turn
	oUnload	Unload

Powered by QuickOPC
www.opclabs.com

ACTUATORS

Counter
Emitter 1 (Emit)
Entry conveyor
FACTORY I/O (Camera Position)
FACTORY I/O (Pause)
FACTORY I/O (Reset)
FACTORY I/O (Run)
Feeder conveyor
Green indicator
Left conveyor
Load
Red indicator
Remover left
Remover right
Reset light
Right conveyor
Start light
Stop light
Turn
Unload
Yellow indicator

🟩 Bool 🟦 Float 🟧 Int ⬜ Any

5. Finally, close the Driver window and start the simulation (press F5).